ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΑ

Μ.Π.Σ. Προηγμένα Συστήματα Πληροφορικής

Εξόρυξη γνώσης από δεδομένα μεταφορικής αλυσίδας – Πειραματισμός με το εργαλείο Spark MLlib

ΑΔΑΜΟΠΟΥΛΟΣ ΓΕΩΡΓΙΟΣ, ΜΠΣΠ-14001

Εισηγητής: ΓΙΑΝΝΗΣ ΘΕΟΔΩΡΙΔΗΣ

ΕΥΧΑΡΙΣΤΙΕΣ,

στην οικογένεια και στους φίλους μου για την υποστήριξη, σε όλα τα επίπεδα, στην ολοκλήρωση των ακαδημαϊκών μου στόχων και στον κ. Γιάννη Θεοδωρίδη για τις συμβουλές και την καθοδήγησή του ως άμεσος επιβλέπων κατά τη διάρκεια της διπλωματικής μου εργασίας.

ΠΕΡΙΛΗΨΗ

Ένα "Σύστημα Υποστήριξης Αποφάσεων (ΣΥΑ)" είναι ένα είδος πληροφοριακού συστήματος, το οποίο υποστηρίζει τη λήψη αποφάσεων σε επιχειρήσεις και οργανισμούς. Ένα ΣΥΑ έχει σκοπό να βοηθήσει τους αποφασίζοντες να συγκεντρώσουν χρήσιμες πληροφορίες μέσα από ένα συνδυασμό δεδομένων, εγγράφων, προσωπικής γνώσης, ή να βοηθήσει τα επιχειρηματικά μοντέλα να αναγνωρίσουν και να λύσουν προβλήματα και να πάρουν αποφάσεις. Στην εποχή μας ο όγκος των δεδομένων είναι τεράστιος και αυξάνεται με ραγδαίους ρυθμούς. Το γεγονός αυτό έφερε σαν αποτέλεσμα την ανάπτυξη καινούριων τεχνολογιών και εργαλειών για την διαχείρηση τους, όπως η εξόρυξη δεδομένων. Η εξόρυξη δεδομένων συνίσταται στην ανακάλυψη ενδιαφερόντων τάσεων ή προτύπων σχημάτων μέσα σε μεγάλα σύνολα δεδομένων, με σκοπό να καθοδηγήσει αποφάσεις σχετικές με μελλοντικές δραστηριότητες.

Η παρούσα διπλωματική εργασία παρουσιάζει την ανάλυση τέτοιων δεδομένων χρησιμοποιώντας κατανεμημένες τεχνολογίες διαχείρισης δεδομένων και συγκεκριμένα το εργαλείο Spark Mllib (Machine learning library) με στόχο την ορθή λήψη επιχειρηματικών αποφάσεων . Θα αναπτυχθούν σε θεωρητικό επίπεδο έννοιες οπως επιχειρηματική νοημοσύνη (Business intelligence), μεγάλα δεδομένα (Big data), εξόρυξη γνώσης (Data mining), αποθήκη δεδομένων (Data Warehouses) και η διαδικασία ανάλυσης που θα ακολουθηθεί. Θα αντληθούν πραγματικά δεδομένα απο κάποια εταιρεία στα οποία, αφού γίνει καθαρισμός, κατάλληλη προ-επεξεργασία, ελάττωση τους και μετασχηματισμοί, θα εφαρμοσθούν αλγόριθμοι κατηγοριοποίησης (classification) και ομαδοποίησης (clustering). Σκοπός αυτής της διαδικασίας είναι η ανακάλυψη γνώσης μέσα από την μελέτη διάφορων προτύπων (patterns) που θα προκύψουν από την αναλυσή μας. Στην συνέχεια θα γίνουν προτάσεις εφαρμογής αυτής της γνώσης και θα αναλυθούν τα συμπεράσματα-αποτελέσματα.

Table of Contents

Κεφάλαιο 1 - Εισαγωγή

Σήμερα, οι οργανισμοί και οι επιχειρήσεις συγκεντρώνουν τεράστιο όγκο δεδομένων. Έτσι, η μετατροπή αυτής της τεράστιας ποσότητας δεδομένων σε γνώση παραμένει μία πρόκληση για τους περισσότερους οργανισμούς και επιχειρήσεις.

Μέσα από αυτόν τον τεράστιο όγκο ακατέργαστων δεδομένων, πρέπει να είναι σε θέση μία επιχείρηση να μπορεί να επιλύει περίπλοκα προβλήματα και να χρησιμοποιεί την κατάλληλη στρατηγική για να καταλήξει σε σωστές αποφάσεις. Άρα θα πρέπει να εξάγει σημαντική γνώση και να εκμεταλλευτεί όλες τις πληροφορίες που είναι αποθηκευμένες στις βάσεις δεδομένων και σε άλλες παρόμοιες πηγές.

Η εξόρυξη δεδομένων καλύπτει όλες τις παραπάνω ανάγκες με την ανακάλυψη και αξιοποίηση προτύπων, δομών, μοντέλων, τάσεων και συσχετίσεων. Έτσι η ανακάλυψη γνώσης, έχει ως αποτέλεσμα το βέλτιστο σχεδιασμό, τη λήψη των βέλτιστων αποφάσεων και την αύξηση της παραγωγικότητας σε στρατηγικά και επιχειρησιακά επίπεδα και κατά συνέπεια την αύξηση της ανταγωνιστικότητας έναντι άλλων επιχειρήσεων.

1.1 Συστήματα υποστήριξης αποφάσεων και επιχειρήσεις

1.1.1 Ορισμός

Ένα "Σύστημα Υποστήριξης Αποφάσεων (ΣΥΑ)" είναι ένα είδος πληροφοριακού συστήματος, το οποίο υποστηρίζει τη λήψη αποφάσεων σε επιχειρήσεις και οργανισμούς. Ένα σωστά σχεδιασμένο ΣΥΑ είναι ένα διαπροσωπικό σύστημα λογισμικού, το οποίο έχει σκοπό να βοηθήσουν τους αποφασίζοντες να συγκεντρώσουν χρήσιμες πληροφορίες μέσα από ένα συνδυασμό δεδομένων, εγγράφων, προσωπικής γνώσης, ή να βοηθήσει τα επιχειρηματικά μοντέλα να αναγνωρίσουν και να λύσουν προβλήματα και να πάρουν αποφάσεις.

Οι τυπικές πληροφορίες που συγκεντρώνει και παρουσιάζει μια εφαρμογή υποστήριξης αποφάσεων είναι:

- περιεχόμενα όλων των τωρινών πληροφοριών (συμπεριλαμβανομένων σχετικών πηγών πληροφόρησης, cubes, αποθήκες δεδομένων, και marts δεδομένα),
- συγκριτικά στοιχεία πωλήσεων μεταξύ μιας εβδομάδας και της επόμενης,

- προβλεπόμενα ποσά εσόδων βασισμένα σε προβλέψεις πωλήσεων του νέου προϊόντος.

1.1.2 Δομή

Οι τρεις βασικές συνιστώσες της δομής ενός ΣΥΑ είναι:

1. Η βάση δεδομένων (ή βάση γνώσης),
2. Το μοντέλο (δηλαδή, το περιβάλλον απόφασης και τα κριτήρια του χρήστη), και
3. Η αλληλεπίδραση μεταξύ των χρηστών.

Οι ίδιοι οι χρήστες αποτελούν επίσης μια σημαντική συνιστώσα της δομής.

1.2 Εξόρυξη δεδομένων – Μηχανική Μάθηση

1.2.1 Ορισμός

Ο όρος εξόρυξη δεδομένων είναι μία έννοια που συνήθως παραπέμπει σε κάθε είδος φόρμας με μεγάλη ποσότητα δεδομένων ή επεξεργασία δεδομένων (συλλογή, εξαγωγή δεδομένων, warehouse, ανάλυση δεδομένων και στατιστικής) αλλά επίσης γενικεύεται σε κάθε είδος συστήματος υποστήριξης αποφάσεων συμπεριλαμβανομένου της τεχνητής νοημοσύνης, της εκμάθησης μηχανής και της επιχειρηματικής ευφυΐας. Στην ορθή χρήση του όρου η λέξη κλειδί είναι η ανακάλυψη, που ορίζεται ως η ανίχνευση κάτι καινούριου.

1.2.2 Στόχος

Ο πραγματικός στόχος της εξόρυξης δεδομένων είναι η αυτόματη ή ημιαυτόματη ανάλυση μεγάλων ποσοτήτων δεδομένων για την εξαγωγή κάποιου ενδιαφέροντος προτύπου που ήταν άγνωστο μέχρι εκείνη τη στιγμή, όπως ομάδες από εγγραφές δεδομένων (συσταδοποίηση), ασυνήθιστες εγγραφές (anomaly detection) και εξαρτήσεις (κανόνες συσχετίσεων). Αυτό συνήθως συμπεριλαμβάνει τη χρήση βάσης δεδομένων όπως χωρικά ευρετήρια. Αυτά τα πρότυπα ύστερα μπορούν να θεωρηθούν ως μία περιγραφή των δεδομένων εισαγωγής και να χρησιμοποιηθούν για περαιτέρω ανάλυση ή για παράδειγμα στην εκμάθηση μηχανής και στην προγνωστική ανάλυση. Για παράδειγμα, η εξόρυξη δεδομένων θα μπορούσε να προσδιορίσει πολλαπλά σύνολα στα δεδομένα, τα οποία μπορούν να χρησιμοποιηθούν μετά για να εξασφαλίσουν περισσότερο ακριβή αποτελέσματα από ένα σύστημα υποστήριξης αποφάσεων. Παρότι η συλλογή δεδομένων και η προετοιμασία δεδομένων, αλλά και η ερμηνεία των αποτελεσμάτων και εκθέσεων

δεν αποτελούν μέρος της εξόρυξης δεδομένων, παρ' όλα αυτά ανήκουν στην ανακάλυψη γνώσης από βάσεις δεδομένων σαν κάποια επιπρόσθετα βήματα.

Άλλοι σχετικοί όροι της εξόρυξης δεδομένων είναι οι data dredging, data fishing και data snooping, που αναφέρονται στην χρήση μεθόδων της εξόρυξης δεδομένων για να πάρουν δείγματα από μεγαλύτερη συλλογή δεδομένων που είναι (ή μπορεί να είναι) πολύ μικρά για αξιόπιστα στατιστικά συμπεράσματα που έγιναν σχετικά με τη εγκυρότητα των προτύπων που ανακαλύφθηκαν. Αυτές οι μέθοδοι, επίσης, μπορούν να χρησιμοποιηθούν για την δημιουργία νέων υποθέσεων προς εξέταση έναντι μεγαλύτερων συλλογών δεδομένων.

1.2.3 Διαδικασία KDD

Η διαδικασία ανακάλυψης γνώσης από βάσεις δεδομένων (KDD) συνήθως ορίζεται από τα εξής στάδια:

1. Συλλογή
2. Προεπεξεργασία
3. Μετασχηματισμός
4. Εξόρυξη δεδομένων
5. Ερμηνεία/Αξιολόγηση.

Εικόνα 1: Η σκάλα της διαδικασίας KDD

1.2.4 Μηχανική Μάθηση

Η μηχανική μάθηση είναι ένα υπό-πεδίο της επιστήμης των υπολογιστών, που εξελίχθηκε κατά την μελέτη της αναγνώρισης προτύπων και της θεωρίας της υπολογιστικής μάθησης στον τομέα της τεχνητής νοημοσύνης. Η μηχανική μάθηση διερευνά τη μελέτη και κατασκευή αλγορίθμων που μπορούν να μάθουν και να κάνουν προβλέψεις από τα δεδομένα. Οι εν λόγω αλγόριθμοι λειτουργούν με την δημιουργία ενός μοντέλου ως παράδειγμα εισόδου, προκειμένου να κάνει προβλέψεις σε δεδομένα ή αποφάσεις που εκφράζονται ως έξοδοι.

Εντός του τομέα των analytics δεδομένων, η μηχανική μάθηση είναι μια μέθοδος που χρησιμοποιείται για να σχεδιάσει πολύπλοκα μοντέλα και αλγορίθμους που προσφέρονται για την πρόβλεψη. Αυτά τα αναλυτικά μοντέλα επιτρέπουν στους ερευνητές, επιστήμονες δεδομένων, μηχανικούς, και στους αναλυτές την παραγωγή «αξιόπιστων, επαναλαμβανόμενων αποφάσεων και αποτελεσμάτων» και την αποκαλύψη «κρυφών γνώσεων» μέσω της μάθησης από τις ιστορικές σχέσεις και τις τάσεις από τα δεδομένα.

Παρακάτω αναφέρονται κάποια παραδείγματα χρήσης:

- Ανίχνευση απάτης: Εντοπισμός συναλλαγών με πιστωτική κάρτα, οι οποίες περιέχουν δολιοφθορά.
- Πρόγνωση καιρού.
- Αναγνώριση προσώπου.
- φιλτράρισμα Spam: αναφέρονται τα μηνύματα e-mail ως spam ή μη-spam.
- Κατηγοριοποίηση πελατών: για παράδειγμα μία πρόβλεψη, κατά την οποία ποιοι πελάτες θα ανταποκριθούν σε μια συγκεκριμένη προσφορά.
- Robotics.
- Ιατρική διάγνωση: διάγνωση ενός ασθενή ως πάσχων ή μη πάσχοντα από κάποια νόσο.

1.2.5 Τύποι αλγορίθμων μηχανικής μάθησης

Για να εκπαιδεύσουμε το σύστημά μας έχουμε τρεις διαφορετικές μεθόδους:

- Εποπτευόμενη μάθηση: Αυτή η μέθοδος χρησιμοποιεί μία βάση δεδομένων για την εκπαίδευση του συστήματός μας.
- Μη εποπτευόμενη μάθηση: Σε αντίθεση με την εποπτευόμενη μάθηση παίρνει αποφάσεις χωρίς να χρησιμοποιεί κάποια βάση δεδομένων για εκπαίδευση.

- Ενισχυτική μάθηση: Δεδομένα εισόδου παρέχονται ως κίνητρο για το μοντέλο από ένα περιβάλλον, το οποίο πρέπει να ανταποκρίνεται και να αντιδρά. Υπάρχει ανατροφοδότηση του μοντέλου αλλά όχι από μια διαδικασία διδασκαλίας και εποπτευόμενης μάθησης, αλλά από τιμωρίες και ανταμοιβές από το περιβάλλον. [1]

Τέλος, οι πιο σχετικοί αλγόριθμοι ανά μέθοδο εκμάθησης παρατίθενται στον παρακάτω πίνακα:

Supervised	KNN
	Linear regression
	Logistic regression
	Naive Bayes
	Decision trees
	Random Forests
	Neural networks
	SVM
Unsupervised	K-means
	Hierarchical clustering
	Fuzzy clustering
	Gaussian mixture models
	Self-organizing maps
Reinforcement	Temporal Differences
	Sarsa
	Q-learning

Εικόνα 2: Οι πιο σχετικοί αλγόριθμοι ανά μέθοδο εκμάθησης

1.3 Big Data

1.3.1 Ορισμός

Τα τελευταία χρόνια υπάρχει μία τάση που πολλοί έχουν ακούσει, την έννοια των Big Data. Αλλά τι είναι τα Big Data; Σε γενικές γραμμές μπορούμε να πούμε ότι είναι η τάση της ανεπτυγμένης τεχνολογίας, η οποία έχει ανοίξει μία καινούρια προσέγγιση ως προς την κατανόηση και την υποστήριξη αποφάσεων των δεδομένων. Τα Big Data χρησιμοποιούνται για να περιγράψουν ποσότητες δεδομένων, οι οποίες είναι πολύ μεγάλες και το κόστος αποθήκευσης και διαχείρισης τους είναι πολύ μεγάλο για να τις εισάγεις σε μία σχεσιακή βάση για ανάλυση. Για αυτό το λόγο τα Big Data εφαρμόζονται σε δεδομένα που δεν μπορούν να αναλυθούν με τα παραδοσιακά εργαλεία ανάλυσης.

[1] https://en.wikipedia.org/?title=Machine learning

1.3.2 Τα τέσσερα V's των Big Data

Σήμερα πολλές σημαντικές εφαρμογές χρειάζονται εργαλεία Big Data, όπως Internet search, business informatics, social networks, social media, genomics, streaming services, meteorology, οι οποίες έχουν να κάνουν με τεράστιες ποσότητες δεδομένων. Αυτό είναι ένα πρόβλημα που πρέπει να αντιμετωπιστεί.

Το ερώτημα που γεννιέται τώρα είναι: Πότε θεωρούμε κάτι ως Big Data; Σε πολλές εταιρείες αρέσει να χωρίζουν τα Big Data σε τέσσερα V's[2]:

1. Volume: Κάθε μέρα όλο και περισσότερες συσκευές και ακόμα πιο πολύπλοκα πληροφοριακά συστήματα εμφανίζονται. Αυτό το γεγονός έχει ως αποτέλεσμα όλο και περισσότεροι άνθρωποι να έχουν πρόσβαση σε αυτά με αποτέλεσμα να δημιουργούνται τεράστιες ποσότητες δεδομένων, οι οποίες αυξάνονται εκθετικά. Αυτό είναι το πρώτο Big Data πρόβλημα.

2. Velocity: Όπως προαναφέρθηκε νωρίτερα οι συσκευές γίνονται όλο και πιο πολύπλοκες. Οι λειτουργίες παρακολούθησης σε πραγματικό χρόνο αυξάνονται και πρέπει να αντιμετωπιστούν. Η διαχείριση τεράστιων ποσοτήτων δεδομένων συνεχούς ροής είναι μία πρόκληση και τα Big Data μπορούν να την υλοποιήσουν επιτυχώς.

3. Variety: Υπάρχουν άπειροι διαφορετικοί τύποι δεδομένων και πρέπει να διαχειριστούν. Π.χ. Κοινωνικά δίκτυα, υγεία, streaming video κ.α.

4. Veracity: Αυτό το V μπορεί να μην είναι τόσο ευδιάκριτο, αλλά είναι πολύ σημαντικό. Οι περισσότεροι ηγέτες των επιχειρήσεων δεν εμπιστεύονται τις πληροφορίες που χρησιμοποιούν για να πάρουν αποφάσεις.

1.4 Εργαλείο διαχείρισης Spark

1.4.1 Ιστορική αναδρομή του εργαλείου διαχείρισης δεδομένων Spark

Το εργαλείο διαχείρισης δεδομένων Spark είναι ένα open source εργαλείο, το οποίο δημιουργήθηκε από μία ομάδα προγραμματιστών. Ξεκινάει το 2009 στο εργαστήριο RAD του πανεπιστημίου του Berkeley, αλλά τελικά γίνεται μέρος του εργαστηρίου AMP. Το εργαλείο διαχείρισης δεδομένων Spark δημιουργείται για να

[2] http://nutch.apache.org/

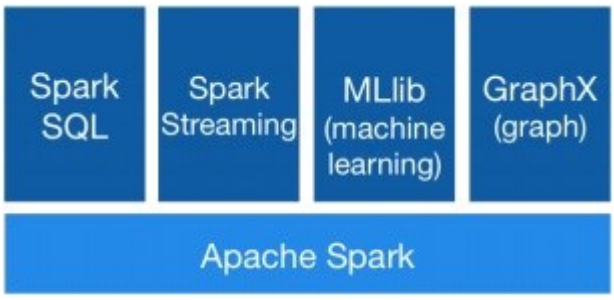
μπορέσουν να παρατηρήσουν την αναποτελεσματικότητα του MapReduce (εργαλείο BIG DATA) στην επανάληψιμότητα και διαδραστικότητα σε διάφορες υπολογιστικές εργασίες. Για αυτό το λόγο ακόμα και από το ξεκινημά του, δημιουργήθηκε να είναι γρήγορο σε συγκεκριμένες υπολογιστικές εργασίες. Επίσης έφερε ιδέες, όπως υποστήριξη για την εσωτερική μνήμη και αποτελεσματική ανάκτηση σφάλματος. Με αποτέλεσμα να είναι 10-20 φορές πιο γρήγορο από το MapReduce για συγκεκριμένες υπολογιστικές εργασίες.[3]

Στην αρχή οι χρήστες ήταν ομάδες μέσα από το πανεπιστήμιο του Berkeley, αλλά σε πολύ σύντομο χρονικό διάστημα πάνω από 50 οργανισμοί άρχισαν να χρησιμοποιούν το Spark. Οι πιο σημαντικοί παράγοντες για την δημιουργία του σημερινού Spark είναι το πανεπιστήμιο Berkeley, η Databricks, η Yahoo! και η Intel.

1.4.2 Εισαγωγή στο εργαλείο διαχείρισης Spark

Το Apache Spark δημιουργήθηκε με την γλώσσα προγραμματισμού scala. Είναι μία γρήγορη και γενικής χρήσεως μηχανή επεξεργασίας μεγάλης κλίμακας παράλληλων δεδομένων, η οποία επεκτείνει το δημοφιλές μοντέλο MapReduce. Μπορούμε να πούμε ότι ένα από τα πιο σημαντικά χαρακτηριστικά του Spark είναι ότι λειτουργεί στην μνήμη. Το γεγονός αυτό, έχει ως αποτέλεσμα να είναι πιο αποτελεσματικό σε υπολογισμούς, λόγω της αποφυγής του δίσκου ανάγνωσης / εγγραφής συμφόρησης.

Το έργο Spark περιέχει πολλές συνιστώσες: Spark SQL, Spark Streaming, MlLib και GraphX. Αυτές οι συνιστώσες έχουν σχεδιαστεί για να εργαστούν από κοινού, όποτε το θελήσει κανείς. Έτσι, μπορούν να συνδυαστούν σε ένα έργο λογισμικού, όπου το Spark αποτελεί τον πυρήνα τους. Τότε το SPARK είναι υπεύθυνο για τον προγραμματισμό, τη διανομή, και τις εφαρμογές παρακολούθησης ενός συμπλέγματος(cluster).



[3] http://oracle4ryou.blogspot.gr/2014/09/hadoop-history.html

Εικόνα 3: Συνιστώσες Spark[4]

Τέλος, πριν προχωρήσουμε πιο βαθιά, αξίζει να αναφέρουμε ότι το Spark είναι επίσης πολύ προσιτό. Προσφέρει μερικά ενδιαφέροντα APIs σε Python, Java, scala, SQL, και ενσωματωμένες βιβλιοθήκες . Επίσης, μπορεί να τρέξει πάνω στο Hadoop, πράγμα που σημαίνει ότι μπορεί να λειτουργεί σε συνεργασία και με άλλα εργαλεία Big Data.

1.4.3 Βασικές ιδέες

Για να κατανοήσουμε τη λειτουργία του Spark είναι απαραίτητο να κατανοήσουμε τις ακόλουθες λέξεις:

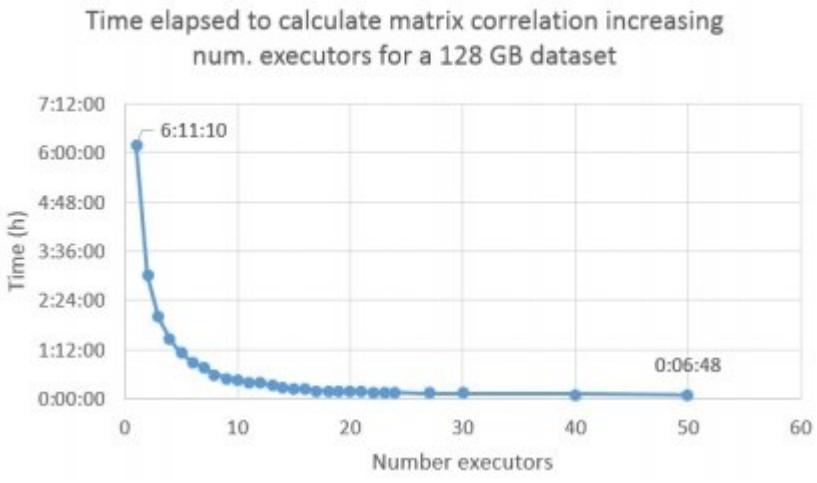
• job: Είναι μια παράλληλη εργασία υπολογισμού, η οποία περιέχει κάποια στοιχεία από HDFS, HBase[5], Cassandra[6] κ.α. και εκτελεί κάποιον υπολογισμό για αυτά τα δεδομένα (π.χ. συλλογής).

• tasks: Κάθε στάδιο έχει κάποιες εργασίες (tasks), κάθε εργασία εκτελείται σε ξεχωριστό διαμέρισμα (partition) από ένα μηχάνημα εκτέλεσης (executor).

• Executor: Η διαδικασία που είναι υπεύθυνη για την εκτέλεση μιας εργασίας.

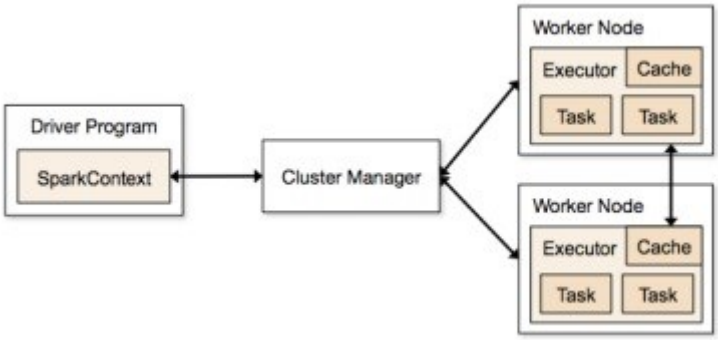
Ο αριθμός των εκτελεστών που χρησιμοποιούνται στα προγράμματα έχουν άμεση σχέση με τον χρόνο εκτέλεσης μια εργασίας. Η εικόνα παρακάτω μας δείχνει πως χρησιμοποιώντας ένα σωστό αριθμό εκτελεστών μειώνεται εκθετικά ο χρόνος εκτέλεσης των εργασιών μας. Αν και μπορούμε επίσης να εκτιμήσουμε, ότι υπάρχει ένα όριο όπου η αύξηση του αριθμού των εκτελεστών δεν μειώνει το χρόνο εκτέλεσης αλλά τον αυξάνει.

[4] Extracted from the book: Learning Spark, Lighting-fast data analysis
[5] https://hbase.apache.org/
[6]http://cassandra.apache.org/



Εικόνα 4: Σχέση χρόνου με αριθμό εκτελεστών

Το παρακάτω σχήμα δείχνει πως ένα πρόγραμμα οδήγησης στέλνει μία εργασία (task) στους εκτελεστές (executors) για να εκτελεστεί.[7]



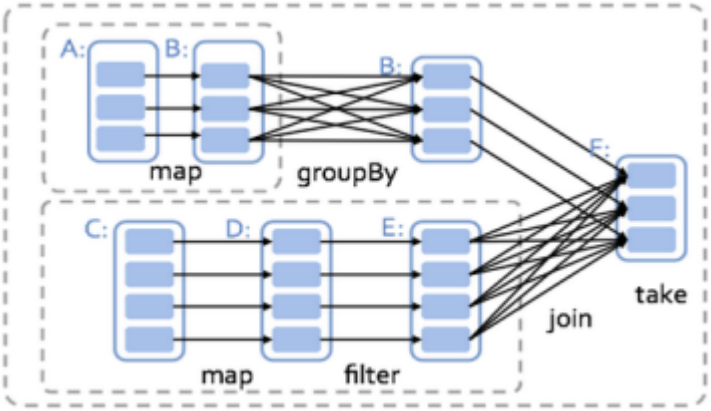
Εικόνα 5: Διαδικασία προγράμματος οδήγησης

- Driver: το πρόγραμμα / διαδικασία που είναι υπεύθυνη για τη λειτουργία της εργασίας πάνω στη μηχανή Spark.
- Master: Η μηχανή στην οποία εκτελείται το πρόγραμμα οδήγησης.
- Slave: Η μηχανή στην οποία εκτελείται το πρόγραμμα εκτέλεσης.
- Stages: Οι θέσεις εργασίας χωρίζονται σε στάδια. Τα στάδια ταξινομούνται σε έναν χάρτη ή μειώνονται και χωρίζονται με βάση τα υπολογιστικά όρια.

[7] http://spark.apache.org/docs/1.3.0/cluster-overview.html

Spark DAG (directed acyclic graph)

Το μοντέλο εκτέλεσης DAG είναι ουσιαστικά μια γενίκευση του μοντέλου MapReduce. Ενώ το μοντέλο MapReduce έχει 2 είδους βήματα υπολογισμού (ένα βήμα χάρτη (map) και ένα βήμα μείωσης (reduce)), το Spark μπορεί να περιέχει οποιοδήποτε αριθμό σταδίων. Αυτό έχει σαν αποτέλεσμα κάποιες εργασίες να ολοκληρώνονται πιο γρήγορα από ότι θα έκαναν με το MapReduce. Για παράδειγμα, αν έχουμε υποβάλει ένα job στο Spark, όπως στο δεύτερο στάδιο της εικόνας 5, η οποία περιέχει μία λειτουργία χάρτη που ακολουθείται από μία λειτουργία φίλτρου, το μοντέλο εκτέλεσης DAG θα βελτιστοποιήσει και θα αναδιατάξει τη σειρά αυτών των λειτουργιών και θα τις αναπρογραμματίσει. Αυτό σημαίνει ότι, σε περιπτώσεις που το MapReduce πρέπει να γράψει σε ενδιάμεσα αποτελέσματα στο κατανεμημένο σύστημα, το Spark μπορεί να προσπεράσει κατευθείαν στο επόμενο βήμα στον αγωγό (pipeline).



Εικόνα 6: Παράδειγμα για το πώς το Spark υπολογίζει τα στάδια εργασίας.

Τα κουτιά που είναι με συμπαγείς γραμμές είναι τα RDDs. Τα partitions είναι τα σκιασμένα ορθογώνια. Στο πρώτο στάδιο εκτελείται μια λειτουργία χάρτη. Το δεύτερο στάδιο εφαρμόζει μία λειτουργία χάρτη και φιλτράρεται. Τέλος, τα αποτελέσματα και των δύο σταδίων εντάσσονται στο τρίτο στάδιο (αποτελέσματα).[8]

RDD (resilient distributed dataset)

Αυτή είναι η έννοια του πυρήνα στο Spark. Επισήμως, η RDD είναι μόνο για ανάγνωση, κατανομή και συλλογή των αντικειμένων, τα οποία μπορούν να αποθηκευτούν είτε σε μνήμη ή σε δίσκο. RDDS μπορούν να δημιουργηθούν με 2

[8] http://blog.cloudera.com/blog/2014/05/apache-spark-resource-management-andyarn-app-models/

τρόπους: με τη φόρτωση ενός εξωτερικού συνόλου δεδομένων ή συλλέγοντας κατανεμημένα αντικείμενα χρησιμοποιώντας την εντολή parallelize.

Τα κύρια χαρακτηριστικά της χρήσης των RDDs είναι[9]:

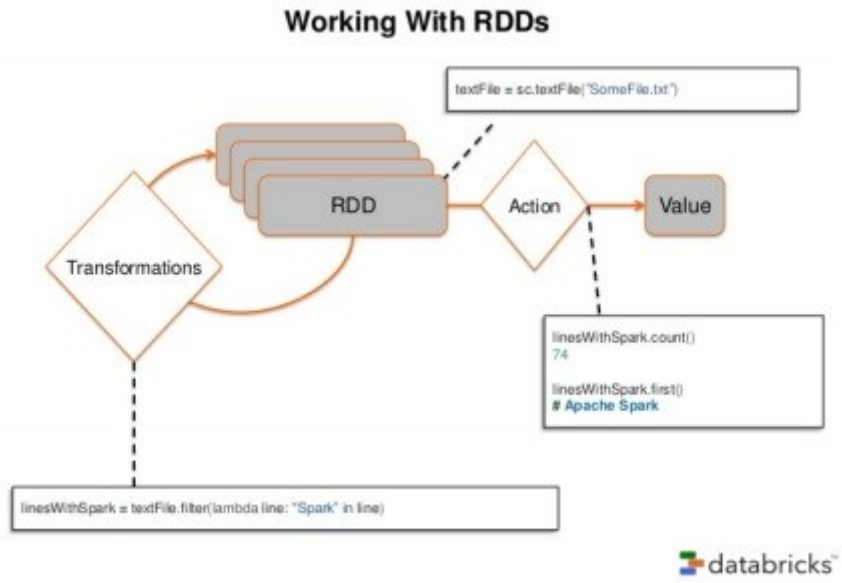
- Το σφάλμα ανεκτικότητας (fault tolerant) χρησιμοποιεί coarse-grained μετασχηματισμούς. Αυτού του είδους μετασχηματισμοί, θα πρέπει να εφαρμόζονται σε όλα τα σετ δεδομένων, αλλά όχι ως μεμονωμένα στοιχεία. Με αυτή την λειτουργία χάνουμε την ευελιξία, αλλά από την άλλη καθίσταται δυνατή η αποθήκευση των εργασιών που πραγματοποιούνται στα δεδομένα μας στο DAG.
  Επίσης σε περίπτωση που ένα partition μιας RDD χαθεί, η RDD έχει αρκετές πληροφορίες σχετικά με το πώς προήλθε από άλλες RDDs και μπορεί να ξανά-υπολογιστεί το συγκεκριμένο partition. Άρα τα δεδομένα μπορούν να ανακτηθούν χωρίς να απαιτείται δαπανηρή αντιγραφή.

- Χρησιμοποιεί lazy evaluation. Lazy evaluation σημαίνει, ότι όταν ζητάμε έναν μετασχηματισμό από μία RDD, η λειτουργία αυτή δεν εκτελείται αμέσως. Αντί αυτού καταγράφει δεδομένα για να δείξει, ότι έχει ζητηθεί αυτή η λειτουργία. Αυτή η ιδιότητα χρησιμοποιείται για να μειώσει τον αριθμό των περασμάτων στα δεδομένα που έχει αναλάβει η Spark, ομαδοποιώντας τις λειτουργίες.

- Όπως είπαμε πριν, DAG είναι ένα σύνολο δεδομένων το οποίο έχει χωριστεί σε partitions. Κάθε partition μπορεί να είναι παρόν στη μνήμη ή το δίσκο διαφορετικών μηχανών. Έτσι, αν επεξεργαστούμε μια RDD, στη συνέχεια το Spark θα πρέπει να ξεκινήσει μία εργασία ανά partition της RDD.

- Η RDDs δέχεται δύο τύπους δραστηριοτήτων: μετασχηματισμούς και δράσεις.

[9] https://es.slideshare.net/tsliwowicz/reversim2014



Εικόνα 7: Φόρτωση αρχείου κειμένου σε RDD

Στην εικόνα 6 έχουμε δημιουργήσει το πρώτο μας RDD (textfile) φορτώνοντας ένα αρχείο κειμένου. Μόλις έχει δημιουργηθεί το RDD εφαρμόζουμε ένα μετασχηματισμό ταξινόμησης που θα μας επιστρέψει ένα νέο RDD (LinesWithSpark) με το σύνολο δεδομένων φιλτραρισμένο από τις γραμμές που περιέχονται στο Spark. Τέλος εκτελούνται δύο δράσεις για το linesWithSpark RDD που θα μας επιστρέψουν τις επιθυμητές τιμές.

Caching

Το Apache Spark δίνει την δυνατότητα να αποθηκεύονται κατά τη διάρκεια εκτέλεσης ενός job ενδιάμεσα αποτελέσματα, (που είναι και αυτά με τη σειρά τους RDD) στην κύρια μνήμη των κόμβων του cluster, κάτι που ονομάζεται caching.

Παραδοσιακά υπολογιστικά μοντέλα όπως το MapReduce, που ακολουθούν παρόμοια φιλοσοφία εκτέλεσης μπορούνε μόνο να διατηρήσουν δεδομένα στη κύρια μνήμη στα πλαίσια που αυτά επεξεργάζονται από ένα συγκεκριμένο task κάποια δεδομένη στιγμή. Σε περίπτωση που τα ίδια δεδομένα χρειάζονται και από κάποιο επόμενο task που θα εκτελεστεί στον ίδιο κόμβο τότε αυτά παρέχονται από το δίσκο. Όπως είναι προφανές αυτό δημιουργεί ένα αδύναμο σημείο στην εκτέλεση των task καθώς η ταχύτητα προσπέλασης δεδομένων στη κύρια μνήμη σε σύγκριση με τον δίσκο είναι υποπολλαπλάσια. Πολύ σημαντικά οφέλη από το caching αποκομίζουν τα jobs εκείνα που χρησιμοποιούν επαναληπτικούς αλγόριθμους (iterative algorithms), που επεξεργάζονται σε κάθε επανάληψη το dataset με το οποίο τροφοδοτούνται.

Παρότι το caching είναι ισχυρός σύμμαχος της αποδοτικότητας ενός job αυτό δεν σημαίνει ότι το σύστημα δεν χρησιμοποιεί το δίσκο του κάθε κόμβου ως μέσο αποθήκευσης. Όταν τα απαιτούμενα δεδομένα που επιλέγονται για caching ξεπερνούν τον διαθέσιμο χώρο που παρέχεται από τη storage region τότε αναπόφευκτα το Spark επιλέγει partitions τα οποία γράφει στο δίσκο. Ακόμα μπορεί να υπάρξει η ανάγκη για deserialization δεδομένων στην unroll region που να διεκδικήσει χώρο από τα αποθηκευμένα partitions όπου θα πρέπει να πράξει ανάλογα. Σε αυτές τις περιπτώσεις το Spark χρησιμοποιεί την πολιτική Least Recently Used (LRU) για να διαλέξει ποια partitions θα γράψει στο δίσκο. Σύμφωνα με αυτή τη πολιτική το πιο παλαιό χρονικά partition που έχει δεσμεύσει cache μνήμη διαλέγεται για αποχώρηση ώστε να αποθηκευτεί στη θέση του κάποιο νεότερο.

1.4.4 Machine Learning Library (MLlib)

Ένα από τα πιο σημαντικά εργαλεία στην Spark είναι η βιβλιοθήκη μηχανικής μάθησης . Αυτή η βιβλιοθήκη είναι μέρος του πυρήνα Spark, ως εκ τούτου μπορεί να χρησιμοποιηθεί από οποιαδήποτε άλλη από τις άλλες βιβλιοθήκες. Ο σχεδιασμός και η φιλοσοφία Mllib είναι απλή: επιτρέπει να επικαλούνται διάφοροι αλγόριθμοι σε κατανεμημένα σύνολα δεδομένων, που αντιπροσωπεύουν όλα τα δεδομένα σε RDDs. Η Mllib εισάγει μερικούς τύπους δεδομένων (π.χ., επισημασμένα σημεία και φορείς), αλλά στο τέλος της ημέρας, είναι απλά ένα σύνολο λειτουργιών που καλεί RDDs.[10]

Ο αλγόριθμοι που είναι διαθέσιμοι είναι:

[10] https://en.wikipedia.org/?title=Machine learning

Basic statistics	summary statistics
	correlations
	stratified sampling
	hypothesis testing
	random data generation
Classification and regression	linear models (SVMs, logistic regression, linear regression)
	naive Bayes
	decision trees
	ensembles of trees (Random Forests and Gradient-Boosted Trees)
	isotonic regression
Collaborative filtering	alternating least squares (ALS)
Clustering	k-means
	Gaussian mixture
	power iteration clustering (PIC)
	latent Dirichlet allocation (LDA)
	streaming k-means
Dimensionality reduction	singular value decomposition (SVD)
	principal component analysis (PCA)
Frequent pattern mining	FP-growth
Optimization (developer)	stochastic gradient descent
	limited-memory BFGS (L-BFGS)

Εικόνα 8: Αλγόριθμοι της βιβλιοθήκης MLlib

Κεφάλαιο 2

Ενότητα 1: Θεωρητικό Υπόβαθρο

Σε αυτήν την ενότητα θα παρουσιάσουμε το θεωρητικό και τεχνολογικό υπόβαθρο που απαιτείται για την κατανόηση των αλγορίθμων και των τεχνικών που θα χρησιμοποιηθούν στην επίλυση των προβλημάτων μίας ελληνικής εταιρείας μεταφορών. Συγκεκριμένα θα αναλυθούν οι αλγόριθμοι συσταδοποίησης και οι αλγόριθμοι κατηγοριοποίησης. Στους αλγόριθμους συσταδοποίησης αναλύεται ιδιαιτέρως ο αλγόριθμος kmeans ενώ στους αλγόριθμους κατηγοριοποίησης αναλύονται οι αλγόριθμοι δένδρου ταξινόμησης , δένδρου παλινδρόμησης και ο Naïve Bayes.

2.1 Αλγόριθμοι και τεχνικές Datamining

2.1.1. Αλγόριθμοι συσταδοποίησης

Η έννοια της συσταδοποίησης προκύπτει στα προβλήματα εκείνα κατά τα οποία δεδομένου ενός συνόλου αντικειμένων μας ενδιαφέρει να δημιουργήσουμε επιμέρους ομάδες μέσα στις οποίες τοποθετούμε αντικείμενα που ικανοποιούν μια έννοια ομοιότητας. Αποτελεί μία από τις πιο συνηθισμένες μορφές μη επιβλεπόμενης μάθησης, όπου απουσία επίβλεψης εννοείται η μη πρότερη γνώση ύπαρξης σαφώς ορισμένων ομάδων μεταξύ των υπό εξέταση αντικειμένων. Συχνά η συσταδοποίηση είναι μία προκαταρτική διαδικασία που μας προσφέρει χρήσιμη γνώση για επόμενα στάδια ανάλυσης σε χώρους αντικειμένων με μικρή πρότερη εμπειρία.

Παρότι η κατηγοριοποίηση των αντικειμένων του φυσικού κόσμου από τους ανθρώπους είναι μια υποσυνείδητη και αυτοματοποιημένη διαδικασία, στον κόσμο των υπολογιστών και της μηχανικής μάθησης οι έννοιες αντικείμενα και ομοιότητα χρειάζονται μαθηματική μοντελοποίηση. Αυτή αφορά συνήθως στην αναπαράσταση των αντικειμένων που μας ενδιαφέρουν σε διανύσματα στο ν-διάστατο ευκλείδειο χώρο όπου το πλήθος των διαστάσεων του διανύσματος αντικατοπτρίζει τον αριθμό των χαρακτηριστικών ιδιοτήτων που περιγράφουν το εκάστοτε αντικείμενο, ενώ οι αντίστοιχες τιμές αυτών τα ποσοτικοποιούν σε μια επιλεγμένη κλίμακα. Καθώς λοιπόν τα αντικείμενα αποτελούν διανύσματα στο χώρο η έννοια της ομοιότητας αντιστοιχίζεται εύλογα με κάποια νόρμα του ευκλείδειου χώρου. Ανάμεσα στις

πολλές επιλογές οι συνηθέστερες αφορούν την ευκλείδεια απόσταση, την απόσταση manhattan και την απόσταση συνημίτονου.

Ανεξάρτητα από την μοντελοποίηση και την επιλογή συνάρτησης απόστασης, η ίδια η συσταδοποίηση κατηγοριοποιείται σε επίπεδη και ιεραρχική ανάλογα με το τελικό αποτέλεσμα. Συγκεκριμένα η επίπεδη συσταδοποίηση αναφέρεται σε πλήρως ανεξάρτητες συστάδες αντικειμένων με μηδενική αλληλεξάρτηση που στόχο έχει μόνο την ανάδειξη ομοιοτήτων αντικειμένων-μελών μόνο της ίδιας συστάδας. Η δεύτερη κατηγορία συσταδοποίησης στοχεύει στην εμφώλευση συστάδων σε δενδρική μορφή, όπου στη ρίζα έχουμε συστάδες που περιγράφουν γενικευμένες ομαδοποιήσεις ενώ οι ενδιάμεσοι κόμβοι αποτελούν πιο εξειδικευμένες ομαδοποιήσεις. Η τελευταία κατηγορία οπτικοποιείται ιδανικά με τα λεγόμενα δενδρογράμματα.

Επιπρόσθετα οι συσταδοποιήσεις μπορούν να χαρακτηριστούν ολικές ή μερικές. Στις ολικές συσταδοποιήσεις ο τελικός σκοπός είναι κάθε ένα από τα αντικείμενα εισόδου της διαδικασίας να αποτελούν μέλος μιας συστάδας. Αντίθετα στις μερικές μπορεί να υπάρξουν και αντικείμενα τα οποία δεν εμπεριέχονται σε κάποια συστάδα μετά το πέρας της συσταδοποίησης. Η τελευταία κατηγορία αποτελεί λογική επιλογή όταν θέλουμε να απομονώσουμε αντικείμενα του χώρου που είναι αισθητά διαχωρίσιμα από τα υπόλοιπα και λειτουργούν σαν ένα είδος θορύβου.

Ακόμα ένα στοιχείο που κατηγοριοποιεί την διαδικασία, είναι ο τρόπος που αντιλαμβανόμαστε την έννοια μιας συστάδας. Προς αυτή τη κατεύθυνση έχουμε συστάδες όπου εκπροσωπούνται από αντιπροσώπους (prototype-based), οι οποίες αποτελούνται από αντικείμενα που βρίσκονται πιο κοντά στο συγκεκριμένο αντιπρόσωπο σε σχέση με αυτούς των άλλων συστάδων. Επίσης έχουμε συστάδες που δημιουργούνται από αντικείμενα που βρίσκονται πιο κοντά σε τουλάχιστον ένα αντικείμενο της ίδιας συστάδας σε σχέση με οποιοδήποτε άλλο σε έταιρη συστάδα (continuity-based). Επιπρόσθετα μπορεί να έχουμε συστάδες που βασίζονται στη πυκνότητα αντικειμένων στο χώρο που εξετάζουμε (densitybased), σε αυτή τη τελευταία περίπτωση, συστάδα αντικειμένων θεωρούμε μια περιοχή με υψηλή περιεκτικότητα και εκτός αυτής αισθητά χαμηλότερη περιεκτικότητα.

Από τα παραπάνω φαίνεται πως η συσταδοποίηση έχει στενή εξάρτηση με το είδος και τα ιδιαίτερα χαρακτηριστικά του συνόλου των αντικειμένων στο οποίο θα εφαρμοστεί. Απόρροια αυτών είναι και η ύπαρξη μιας πληθώρας αλγορίθμων, με πολύ διαφορετικές πολυπλοκότητες, που την υλοποιούν. Η γενικότερη αξία και αναγκαιότητα της διαδικασίας πάντως είναι και αυτή που την καθιστά χρήσιμη σε ένα μεγάλο εύρος επιστημονικών πεδίων όπως η μηχανική εκμάθηση, η αναγνώριση προτύπων, η ανάλυση εικόνας και βίντεο, η εξόρυξη δεδομένων, η βιοπληροφορική και πολλά άλλα.

2.1.2. Αλγόριθμος K-means

Ο συγκεκριμένος αλγόριθμος είναι από τους πιο πολυεφαρμοσμένους και είναι η ρίζα για πολλούς άλλους. Ανήκει στην κατηγορία της επίπεδης συσταδοποίησης διότι παράγει ένα σύνολο συσταδοποιήσεων οι οποίες δεν έχουν κάποια ιδιαίτερη δομή-σχέση μεταξύ τους. Ο αλγόριθμος έχει ως στόχο τη βελτιστοποιήση μίας συνάρτησης – της συνάρτησης κόστους.[11]

Αρχικά έχουμε k-ομάδες, με την κάθε ομάδα να αντιπροσωπεύεται από το μέσο διάνυσμα. Ο αλγόριθμος λειτουργεί ως εξής:

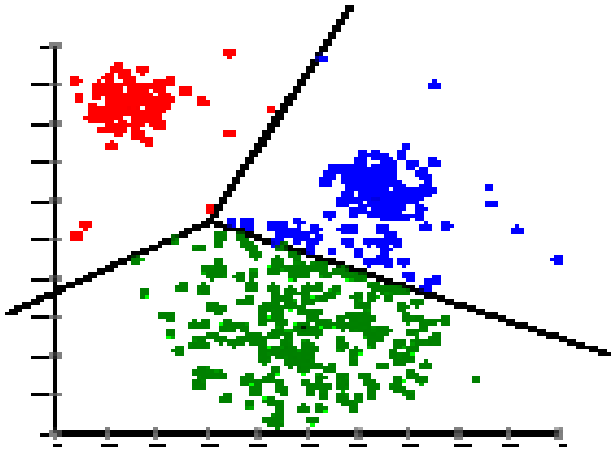
1. Κατά τη φάση της διαμέρισης γίνεται προσπέλαση των διανυσμάτων και για κάθε διάνυσμα βρίσκουμε την απόστασή του από (ανομοιότητα με) τις υπάρχουσες ομάδες. Η απόσταση μεταξύ ενός διανύσματος και μίας ομάδας είναι η ευκλείδεια απόσταση από το μέσο (mp) διάνυσμα.
2. Για i=1,2...N υπολογίζουμε τις αποστάσεις ως εξής:

   d(x_{i},C_{j})=\|x_{i}-m_{j}\|^{2}}
   ∀j=1,2,3...k. Προσδιορίζονται N αριθμοί q, έτσι ώστε να ισχύει: d(xi,Cq) ≤ d(xi,Cj) ∀j=1,2,3...k. Κατά την ολοκλήρωση αυτού του βήματος σχηματίζονται k-σύνολα διανυσμάτων, ένα σύνολο για κάθε ομάδα.

3. Κατά τη φάση της ενημέρωσης, γίνονται οι κατάλληλες τροποποιήσεις στα μέσα διανύσματα, δηλαδή ∀j=1,2...m: υπολογίζονται ξανά τα μέσα διανύσματα mi για i=1,2...κ. Στον υπολογισμό του νέου mi συνεισφέρει το αντίστοιχο σύνολο διανυσμάτων που υπολογίστηκε κατά το παραπάνω βήμα.

Ο αλγόριθμος ολοκληρώνεται όταν οι ενημερώσεις που γίνονται σε κάθε mi είναι αμελητέες. Σημαντικό σημείο του αλγορίθμου είναι η αρχικοποίηση των k-διανυσμάτων. Αν αντί για το μέσο διάνυσμα χρησιμοποιήσουμε ένα διάνυσμα xj, με xj∈X, τότε έχουμε τον αλγόριθμο k-εσωτερικών αντιπροσώπων. Η χρονική πολυπλοκότητα του αλγόριθμου είναι O(Nkq) όπου q ο αριθμός των επαναλήψεων που πρέπει να εκτελέσει ο αλγόριθμος για να τερματίσει.

[11] https://en.wikipedia.org/wiki/K-means_clustering



Εικόνα 9: Συσταδοποίηση από τον K-means

2.1.3. Αλγόριθμοι κατηγοριοποίησης

Η κατηγοριοποίηση (classification) είναι μία τεχνική της εξόρυξης δεδομένων, κατά την οποία ένα στοιχείο ανατίθεται σε ένα προκαθορισμένο σύνολο κατηγοριών. Ο όρος κατηγοριοποίηση συναντάται στην βιβλιογραφία και ως ταξινόμηση. Γενικότερα, ο στόχος της διαδικασίας αυτής είναι η ανάπτυξη ενός μοντέλου, το οποίο αργότερα θα μπορεί να χρησιμοποιηθεί για την κατηγοριοποίηση μελλοντικών δεδομένων. Τέτοια παραδείγματα είναι ο διαχωρισμός των emails με βάση την επικεφαλίδα τους ή το περιεχόμενό τους, η πρόβλεψη καρκινικών κυττάρων χαρακτηρίζοντας τα ως καλοήθη ή κακοήθη, η κατηγοριοποίηση πελατών μιας τράπεζας ανάλογα με την πιστωτική τους ικανότητα κ.α.

Η κατηγοριοποίηση μπορεί να περιγραφεί ως μία διαδικασία δύο βημάτων:

Εκμάθηση (Learning):

Στο πρώτο βήμα της διαδικασίας δημιουργείται/προσδιορίζεται το μοντέλο με βάση ένα σύνολο προκατηγοριοποιημένων παραδειγμάτων, που ονομάζεται δεδομένα εκπαίδευσης (training data).Τα δεδομένα εκπαίδευσης αναλύονται από ένα αλγόριθμο κατηγοριοποίησης, προκειμένου να σχηματιστεί το μοντέλο. Λόγω του ότι τα δεδομένα εκπαίδευσης ανήκουν σε μία προκαθορισμένη κατηγορία, η οποία είναι γνωστή, η κατηγοριοποίηση αποτελεί μέθοδος εποπτευομένης μάθησης (supervised learning). Το μοντέλο, που λέγεται και αλλιώς κατηγοριοποιητής (classifier), αναπαρίσταται με τη μορφή κανόνων κατηγοριοποίησης (classification rules), δέντρων απόφασης (decision trees) ή μαθηματικών τύπων.

Κατηγοριοποίηση (Classification):

Μετά την δημιουργία του μοντέλου, το επόμενο βήμα είναι η αξιολόγησή του. Για να επιτευχθεί αυτό, χρησιμοποιούμε τα δοκιμαστικά δεδομένα (test data) για να υπολογίσουν την ακρίβεια του μοντέλου. Το μοντέλο κατηγοριοποιεί τα δοκιμαστικά δεδομένα. Έπειτα, η κατηγορία που σχηματίστηκε με βάση τα δοκιμαστικά δεδομένα συγκρίνρται με την πρόβλεψη που έγινε για τα δεδομένα εκπαίδευσης, τα οποία είναι ανεξάρτητα από αυτά της δοκιμής. Η ακρίβεια του μοντέλου υπολογίζεται από το ποσοστό των δειγμάτων δοκιμής που κατηγοριοποιήθηκαν σωστά σε σχέση με το υπό εκπαίδευση μοντέλο.

Στην περίπτωση που το μοντέλο κριθεί αποδεκτό, τότε μπορεί να χρησιμοποιηθεί για την κατηγοριοποίηση μελλοντικών δειγμάτων δεδομένων, των οποίων η κατηγοριοποίηση είναι άγνωστη.

2.1.4. Κατηγορίες μεθόδων κατηγοριοποίησης

Σε αυτήν την ενότητα θα δούμε τις πιο σημαντικές κατηγορίες μεθόδων κατηγοριοποίησης που χρησιμοποιούνται σε τέτοιου είδους αναλύσεις και θα μας βοηθήσουν στην επίλυση του παραπάνω προβλήματος.

Bayesian

Η Bayesian κατηγοριοποίηση αποτελεί μία κατηγορία μεθόδων της κατηγοριοποίησης και βασίζεται στη στατιστική θεωρία κατηγοριοποίησης του Bayes. Αυτο σημαίνει ότι πραγματοποιείται μια πιθανοτική πρόβλεψη, δηλαδή προβλέπει την πιθανότητα ένα δείγμα X να ανήκει σε κάποια κατηγορία. Ο απλούστερος Bayesian κατηγοριοποιητής είναι ο Naïve Bayesian. Αυτός υποθέτει ότι η επίδραση ενός γνωρίσματος σε μία κατηγορία είναι ανεξάρτητη από τις τιμές των υπόλοιπων γνωρισμάτων. Ο λόγος που γίνεται αυτό είναι για να αποφεύγονται οι πολύπλοκοι υπολογισμοί κατά τη συνθήκη ανεξαρτησίας της κατηγορίας.

Δέντρα Απόφασης στη Μηχανική Μάθηση

Η μηχανική μάθηση με δέντρα απόφασης χρησιμοποιεί ένα δέντρο απόφασης ως προγνωστικό μοντέλο το οποίο χαρτογραφεί παρατηρήσεις σχετικά με ένα στοιχείο (που αντιπροσωπεύεται στα κλαδιά) σε συμπεράσματα σχετικά με την τιμή στόχο του στοιχείου (που αντιπροσωπεύεται στα φύλλα). Είναι μία από τις προσεγγίσεις προγνωστικής μοντελοποίησης που χρησιμοποιούνται στις στατιστικές, εξόρυξη δεδομένων και μηχανική μάθηση. Τα μοντέλα δέντρων όπου η μεταβλητή στόχος μπορεί να πάρει ένα πεπερασμένο σύνολο τιμών ονομάζονται δέντρα ταξινόμησης. Σε αυτές τις δομές δέντρων, τα φύλλα αντιπροσωπεύουν τις

ετικέτες τάξης / κατηγορίας και τα κλαδιά αντιπροσωπεύουν τους συνδέσμους των χαρακτηριστικών που οδηγούν σε αυτές τις ετικέτες κατηγορίας. Τα δέντρα απόφασης, όπου η μεταβλητή στόχος μπορεί να πάρει συνεχείς τιμές (συνήθως πραγματικοί αριθμοί) ονομάζονται δέντρα παλινδρόμησης.[12]

Στην ανάλυση αποφάσεων, ένα δέντρο απόφασης μπορεί να χρησιμοποιηθεί για να αντιπροσωπεύσει ρητά τις αποφάσεις και τη λήψη αποφάσεων. Στην εξόρυξη δεδομένων, ένα δέντρο απόφασης περιγράφει τα δεδομένα (αλλά το προκύπτον δέντρο ταξινόμησης μπορεί να είναι μια εισροή για τη λήψη αποφάσεων).

Το δέντρο απόφασης στη μηχανική μάθηση είναι μια μέθοδος που χρησιμοποιείται ευρέως στον τομέα της εξόρυξης δεδομένων. Ο στόχος είναι να δημιουργηθεί ένα μοντέλο που προβλέπει την τιμή μιας μεταβλητής στόχου βασισμένο σε διάφορες μεταβλητές εισόδου.

Ένα δέντρο απόφασης είναι μια απλή απεικόνιση για την ταξινόμηση παραδειγμάτων. Για το τμήμα αυτό, ας υποθέσουμε ότι όλα τα χαρακτηριστικά εισόδου έχουν πεπερασμένη διακριτά πεδία, και υπάρχει ένα μόνο χαρακτηριστικό στόχου που ονομάζεται ταξινόμηση. Κάθε στοιχείο του τομέα της ταξινόμησης ονομάζεται τάξη. Ένα δέντρο απόφασης ή ένα δέντρο ταξινόμησης είναι ένα δέντρο στο οποίο κάθε εσωτερικός (μη-φύλλο) κόμβος είναι επισημασμένος με ένα χαρακτηριστικό εισόδου. Τα τόξα που έρχονται από έναν κόμβο σημασμένο με ένα χαρακτηριστικό εισόδου επισημαίνονται με κάθε μία από τις πιθανές τιμές του χαρακτηριστικού στόχου ή εξόδου ή το τόξο οδηγεί σε υποδεέστερη κόμβο αποφάσεων σε ένα διαφορετικό χαρακτηριστικό εισόδου. Κάθε φύλλο του δέντρου είναι επισημασμένο με μια τάξη ή μια κατανομή πιθανότητας πάνω από τις τάξεις.

Ένα δέντρο μπορεί να "μάθει" από τη διάσπαση της πηγής σε υποσύνολα βασιζόμενο σε μία δοκιμή με χαρακτηριστικά με συγκεκριμένη αξία. Αυτή η διαδικασία επαναλαμβάνεται σε κάθε υποσύνολο με επαναληπτικό τρόπο που ονομάζεται επαναλαμβανόμενη διαμέριση. Η αναδρομή ολοκληρώνεται όταν το υποσύνολο σε έναν κόμβο έχει την ίδια τιμή της μεταβλητής στόχου, ή όταν ο διαχωρισμός δεν προσθέτει πλέον αξία στις προβλέψεις. Αυτή η διαδικασία της top-down επαγωγής των δέντρων απόφασης είναι ένα παράδειγμα ενός άπληστου αλγόριθμου, και είναι μακράν η πιο κοινή στρατηγική για την εκμάθηση δέντρων απόφασης από δεδομένα.

Τα δεδομένα που καταγράφονται είναι της μορφής:

$(x, Y) = (x_1, x_2, x_3, ..., x_k, Y)$

[12] https://en.wikipedia.org/wiki/Classification

Η εξαρτημένη μεταβλητή, Y, είναι η μεταβλητή στόχος που προσπαθούμε να καταλάβουμε, ταξινομίσουμε ή γενικεύουμε. Το διάνυσμα x αποτελείται από τις μεταβλητές εισόδου, x_1, x_2, x_3 κ.λπ., που χρησιμοποιούνται για την εργασία αυτή.

Τύποι Δέντρων Απόφασης

Τα δέντρα απόφασης που χρησιμοποιούνται στην εξόρυξη δεδομένων χωρίζονται σε δύο βασικά είδη:

- **Ανάλυση Δέντρου Ταξινόμησης** είναι όταν η προβλεπόμενη έκβαση είναι η τάξη στην οποία τα δεδομένα ανήκουν.
- **Ανάλυση Δέντρου Παλινδρόμησης** είναι όταν η προβλεπόμενη έκβαση μπορεί να θεωρηθεί ένας πραγματικός αριθμός

Ο όρος Ανάλυση Δέντρου Ταξινόμησης και παλινδρόμησης (CART) είναι ένας γενικός όρος που χρησιμοποιείται για να αναφερθούμε και στις δύο από τις παραπάνω διαδικασίες, και παρουσιάστηκε για πρώτη φορά από τον Breiman et al. Τα δέντρα που χρησιμοποιούνται για την παλινδρόμηση και τα δέντρα που χρησιμοποιούνται για την ταξινόμηση έχουν κάποιες ομοιότητες - αλλά και κάποιες διαφορές, όπως είναι η διαδικασία που χρησιμοποιείται για να προσδιοριστεί το πού θα γίνει ο διαχωρισμός.

Αλγόριθμος Naive Bayes

Στη μηχανική μάθηση, οι ταξινομητές Naive Bayes είναι μια οικογένεια απλών πιθανολογικών ταξινομητών που βασίζονται στην εφαρμογή του θεωρήματος του Bayes με υποθέσεις ισχυρής ανεξαρτησίας μεταξύ των χαρακτηριστικών.[13]

Ο Naive Bayes έχει μελετηθεί εκτενώς από το 1950. Εισήχθη με διαφορετικό όνομα στην κοινότητα ανάκτησης κειμένου στις αρχές της δεκαετίας του 1960 και παραμένει μια δημοφιλής μέθοδος για την κατηγοριοποίηση κειμένου, για το πρόβλημα του να κρίνουμε έγγραφα που ανήκουν στη μία κατηγορία ή την άλλη (όπως spam ή νόμιμα, αθλητικά ή πολιτικά, κλπ) με συχνότητα λέξεων, όπως τα χαρακτηριστικά. Με την κατάλληλη προ-επεξεργασία, είναι ανταγωνιστική αυτή η μέθοδος σε αυτόν τον τομέα με πιο προηγμένες μεθόδους, συμπεριλαμβανομένων των μηχανών διανυσμάτων υποστήριξης. Βρίσκει επίσης εφαρμογή στην αυτόματη ιατρική διάγνωση.

Οι Naive Bayes ταξινομητές είναι εξαιρετικά επεκτάσιμοι, που απαιτούν μια σειρά από παραμέτρους γραμμική του αριθμού των μεταβλητών (χαρακτηριστικά /

[13] https://en.wikipedia.org/wiki/Naive_Bayes_classifier

προγνωστικοί παράγοντες) σε ένα πρόβλημα μάθησης. Η εκπαίδευση μέγιστης πιθανόφάνειας μπορεί να γίνει με την αξιολόγηση μιας έκφρασης κλειστής μορφής, η οποία λαμβάνει γραμμικό χρόνο, παρά και όχι με ακριβή επαναληπτική προσέγγιση, όπως χρησιμοποιείται για πολλούς άλλους τύπους ταξινομητών.

Στη λογοτεχνία της στατιστικής και της επιστήμης των υπολογιστών, τα μοντέλα Naive Bayes είναι γνωστά με διάφορες ονομασίες, όπως είναι το απλό Bayes μοντέλο και το μοντέλο ανεξαρτησία του Bayes. Όλα αυτά τα ονόματα αναφορέρουν τη χρήση του θεωρήματος του Bayes στον κανόνα απόφασης του ταξινομητή, αλλά η Naive Bayes δεν είναι (απαραιτήτως) μία Bayesian μέθοδος.

Η Naive Bayes είναι μια απλή τεχνική για την κατασκευή ταξινομητών: μοντέλα που αποδίδουν ετικέτες κατηγορίας στις περιπτώσεις προβλημάτων, αντιπροσωπεύονται ως διανύσματα χαρακτηριστικών αξιών, όπου οι ετικέτες κατηγορίας προέρχονται από κάποιο πεπερασμένο σύνολο. Δεν είναι ένας απλός αλγόριθμος για την κατάρτιση αυτών των ταξινομητών, αλλά μια οικογένεια αλγορίθμων που βασίζεται σε μια κοινή αρχή: όλοι οι ταξινομητές Naïve Bayes υποθέτουν ότι η τιμή ενός συγκεκριμένου χαρακτηριστικού είναι ανεξάρτητη από την τιμή κάθε άλλου χαρακτηριστικού, δεδομένης της μεταβλητής κλάσης. Για παράδειγμα, ένα φρούτο μπορεί να θεωρηθεί ότι είναι ένα μήλο αν είναι κόκκινο, στρογγυλό, και περίπου 10 cm σε διάμετρο. Ένας ταξινομητής Naïve Bayes θεωρεί κάθε ένα από αυτά τα χαρακτηριστικά να συμβάλουν ανεξάρτητα στην πιθανότητα ότι αυτό το φρούτο είναι ένα μήλο, ανεξάρτητα από τυχόν συσχετίσεις μεταξύ των χαρακτηριστικών χρώματος, στρογγυλάδας, και διαμέτρου.

Για ορισμένους τύπους μοντέλων πιθανοτήτων, αφελείς ταξινομητές Bayes μπορεί να εκπαιδευτεί πολύ αποτελεσματικά σε ελεγχόμενο περιβάλλον μάθησης. Σε πολλές πρακτικές εφαρμογές, εκτίμηση παραμέτρων για αφελείς μοντέλα Bayes χρησιμοποιεί τη μέθοδο της μέγιστης πιθανοφάνειας? Με άλλα λόγια, μπορεί κανείς να συνεργαστεί με την αφελή μοντέλο Bayes, χωρίς την αποδοχή Μπεϋζιανή πιθανότητα ή τη χρήση οποιωνδήποτε Bayesian μεθόδους.

Για ορισμένους τύπους μοντέλων πιθανοτήτων, οι Naive Bayes ταξινομητές μπορεί να εκπαιδευτούν πολύ αποτελεσματικά σε ελεγχόμενο περιβάλλον μάθησης. Σε πολλές πρακτικές εφαρμογές, η εκτίμηση παραμέτρων για Naive Bayes μοντέλα χρησιμοποιεί τη μέθοδο της μέγιστης πιθανοφάνειας. Με άλλα λόγια, μπορεί κανείς να δουλέψει με Naïve Bayes μοντέλα, χωρίς την αποδοχή της Μπεϋζιανής πιθανότητας ή τη χρήση οποιωνδήποτε μεθόδων Bayesian.

Παρά τον απλό σχεδιασμό τους και τις υπεραπλουστευμένες υποθέσεις, οι Naïve Bayes ταξινομητές έχουν εργαστεί αρκετά καλά σε πολλές περίπλοκες πραγματικές καταστάσεις. Το 2004, μια ανάλυση του προβλήματος Bayesian ταξινόμησης έδειξε ότι υπάρχουν βάσιμοι θεωρητικοί λόγοι για τη φαινομενικά

απίθανη αποτελεσματικότητα των Bayes ταξινομητών. Ακόμα, μια ολοκληρωμένη σύγκριση με άλλους αλγορίθμους ταξινόμησης το 2006, έδειξε ότι η ταξινόμηση Bayes έχει πολύ καλύτερες επιδόσεις από άλλες προσεγγίσεις, όπως τα ενισχυμένα δέντρα ή τα τυχαία δάση.

Ένα πλεονέκτημα της Naive Bayes τεχνικής είναι ότι απαιτεί μόνο ένα μικρό αριθμό των δεδομένων για την εκτίμηση των παραμέτρων που είναι αναγκαίες για την ταξινόμηση.

Πολυωνυμικά μοντέλα Naive Bayes

Με ένα πολυωνυμικό μοντέλο, τα δείγματα (διανύσματα χαρακτηριστικών) αντιπροσωπεύουν τις συχνότητες με τις οποίες ορισμένα γεγονότα έχουν δημιουργηθεί από ένα πολυώνυμο (p_1, p_2, p_3, …, p_n) όπου p_i είναι η πιθανότητα ότι το γεγονός i συμβαίνει. Ένα διάνυσμα χαρακτηριστικών x = (x_1, …, x_n) είναι έτσι ένα ιστόγραμμα, όπου x_i μετρά πόσες φορές το γεγονός i παρατηρήθηκε σε μία συγκεκριμένη περίπτωση. Αυτό είναι το μοντέλο που τυπικά χρησιμοποιείται για την κατηγοριοποίηση εγγράφων, με τα γεγονότα να αντιπροσωπεύουν την εμφάνιση μιας λέξης σε ένα ενιαίο έγγραφο. Η πιθανότητα παρατήρησης ενός ιστογράμματος x δίνεται από τη σχέση:

$$P(x|\ C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i {p_{ki}}^{x_i}$$

Ο πολυωνυμικός ταξινομητής Naïve Byes γίνεται γραμμικός όταν εκφράζεται σε λογάριθμο.

Εάν μια δεδομένη τάξη και τιμή χαρακτηριστικού ποτέ δεν εμφανίζονται μαζί με τα δεδομένα, τότε η εκτίμηση πιθανότητας συχνότητας θα είναι μηδέν. Αυτό είναι προβληματικό, διότι θα εξαλείψει όλες τις πληροφορίες στις άλλες πιθανότητες, όταν πολλαπλασιάζονται. Ως εκ τούτου, είναι συχνά επιθυμητή η ενσωμάτωση μιας μικρής διόρθωσης του δείγματος, που ονομάζεται ψευδοϋπολογισμός, σε όλες τις εκτιμήσεις της πιθανότητας έτσι ώστε να μην είναι ποτέ ακριβώς μηδέν. Αυτός ο τρόπος τακτοποίησης του Naive Bayes ονομάζεται Laplace εξομάλυνση όταν ο ψευδοϋπολογισμός είναι ένας, και Lidstone εξομάλυνση στη γενική περίπτωση.

Ο Rennie et al. συζητά προβλήματα με την πολυωνυμική υπόθεση στο πλαίσιο της ταξινόμησης εγγράφων και τους πιθανούς τρόπους για την αντιμετώπιση των προβλημάτων αυτών, συμπεριλαμβανομένης της χρήσης των TF-IDF βαρών αντί των συχνοτήτων πρώτου όρου και την εξομάλυνση του μήκους του κειμένου, για την παραγωγή ενός Naïve Bayes ταξινομητή που είναι ανταγωνιστικός σε σχέση με μηχανές διανυσμάτων υποστήριξης.

Νευρωνικά δίκτυα

Μια άλλη τεχνική κατηγοριοποίησης που χρησιμοποιείται σε εφαρμογές εξόρυξης γνώσης για πρόβλεψη και κατηγοριοποίηση στηρίζεται στα νευρωνικά δίκτυα. Τα βήματα αυτής της διαδικασίας χονδρικά είναι:

- Η αναγνώριση των χαρακτηριστικών εισόδου και εξόδου
- Δημιουργία ενός δικτύου με την κατάλληλη τοπολογία
- Επιλογή του συνόλου εκπαίδευσης (train data)
- Εφαρμογή του δικτύου με ένα αντιπροσωπευτικό σύνολο δεδομένων, ώστε να μεγιστοποιείται η δυνατότητα του δικτύου να αναγνωρίζει τα πρότυπα
- Επαλήθευση-αξιολόγηση του δικτύου με την χρήση ενός σύνολο ελέγχου (test data).

Παραγωγή κανόνων κατηγοριοποίησης

Η γνώση που αποκτούμε κατά την διαδικασία της κατηγοριοποίησης μπορεί να αναπαρασταθεί και με τη χρήση κανόνων. Οι κανόνες κατηγοριοποίησης, σε σχέση με τα δέντρα απόφασης, γίνονται ευκολότερα κατανοητοί όταν το δέντρο που παράχθηκε είναι μεγάλο. Έτσι μπορούμε να μετατρέψουμε ένα δέντρο απόφασης σε ένα σύνολο κανόνων κατηγοριοποίησης. Αυτό μπορεί να επιτευχθεί εάν θεωρήσουμε ότι κάθε κανόνας αντιστοιχεί σε ένα μονοπάτι του δέντρου από τη ρίζα μέχρι ένα κόμβο φύλλο. Άρα κάθε φύλλο παράγει ένα κανόνα. Οι συνθήκες που θα μας οδηγήσουν στο φύλλο (υπόθεση) αποτελούν το αριστερό μέρος του κανόνα, ενώ το φύλλο (αποτέλεσμα) αντιστοιχεί στο δεξιό μέρος του κανόνα.

Ενότητα 2: Τεχνολογικό Υπόβαθρο

Σε αυτήν την ενότητα θα αναλυθούν οι αλγόριθμοι που θα χρησιμοποιηθούν στο εργαλείο Spark για τα πειράματά μας, καθώς και οι παράμετροι που θα εισαχθούν σε κάθε ένα από τους αλγορίθμους.

2.2 Ανάλυση αλγορίθμων

Ξεκινώντας την ανάλυση των αλγορίθμων, θα αναφερθούμε στον αλγόριθμο k-means (αλγόριθμος συσταδοποίησης) και στους αλγορίθμους decision tree classification, decision tree regression και naïve bayes (αλγόριθμοι κατηγοριοποίησης).

2.2.1 Spark (MLlib) - kmeans

Όπως προαναφέρθηκε η MLlib είναι ένα από τα υποσυστήματα που εμπεριέχεται στη στοίβα λογισμικού που αποτελούν το Apache Spark. Ουσιαστικά παρέχει μια συλλογή αλγορίθμων machine learning και data science ειδικά σχεδιασμένων για την εκτέλεσή τους στο κατανεμημένο περιβάλλον πάνω από το Spark Core υποσύστημα. Εκτός αυτού ορίζεται μια συλλογή τύπων δεδομένων (data structures) όπως είναι τα labeled points και τα vectors που είναι ειδικά σχεδιασμένοι για τις ανάγκες αυτών των αλγορίθμων. Η ευκολία που παρέχει η MLlib είναι ότι όλα τα παραπάνω υλοποιούνται ως προγραμματιστικές διεπαφές των RDD, δηλαδή από τη προγραμματιστική πλευρά αρκεί η κλήση μιας μεθόδου σε ένα RDD για να γίνει χρήση των δυνατοτήτων της MLlib.

Όσο αφορά στον αλγόριθμο k-means , για την εκτέλεσή του με την Mllib αρκεί να ορίσουμε ως RDD το dataset που θέλουμε να χρησιμοποιήσουμε για clustering και να καλέσουμε μία μέθοδο train() που δέχεται τα ορίσματα initializationMode, maxIterations και runs. Η επιλογή του initializationMode είναι αυτή που κάνει χρήση παραλλαγών του αλγορίθμου όπου γίνεται αρχική επιλογή κεντροϊδών με βελτιστοποιημένο τρόπο, αλλά παρέχεται και η δυνατότητα τυχαίας επιλογής με την παράμετρο random. Η παράμετρος maxIterations αφορά στο μέγιστο αριθμό επαναλήψεων που επιτρέπουμε στον αλγόριθμο να εκτελέσει σε περίπτωση που δεν επιτύχει σύγκλιση νωρίτερα. Τέλος η παράμετρος runs παρέχει τη δυνατότητα να εκτελεστούν παράλληλα περισσότερες της μίας εκτελέσεις του αλγορίθμου ξεκινώντας με διαφορετικά κεντροειδή, έτσι ώστε τελικό αποτέλεσμα επιλέγεται εκείνο που επέτυχε και το καλύτερο clustering.

2.2.2 Decision Tree Classification - Regression

Το δέντρο απόφασης είναι ένας αλγόριθμος που εκτελεί μια αναδρομική δυαδική διαμέριση του χώρου χαρακτηριστικών, επιλέγοντας ένα μόνο στοιχείο από την καλύτερη διάσπαση του συνόλου, όπου κάθε στοιχείο του συνόλου μεγιστοποιεί την πληροφορία που έχει αποκτηθεί σε ένα κόμβο του δένδρου.

Η πρόσμιξη του κόμβου (*node impurity*) είναι ένα μέτρο της ομοιογένειας των ετικετών στον κόμβο. Με άλλα λόγια αντιπροσωπεύει πόσο καλά τα δέντρα απόφασης διαχωρίζουν τα δεδομένα. Η τρέχουσα εφαρμογή παρέχει δύο μέτρα *node impurity* για την ταξινόμηση (Gini impurity και Entropy) και ένα μέτρο *node impurity* για παλινδρόμηση (Variance).

Impurity	Task	Formula	Description
Gini impurity	Classification	$\sum_{i=1}^{M} f_i(1-f_i)$	f_i is the frequency of label i at a node and M is the number of unique labels.
Entropy	Classification	$\sum_{i=1}^{M} -f_i log(f_i)$	f_i is the frequency of label i at a node and M is the number of unique labels.
Variance	Regression	$\frac{1}{n}\sum_{i=1}^{N}(x_i-\mu)^2$	y_i is label for an instance, N is the number of instances and μ is the mean given by $\frac{1}{N}\sum_{i=1}^{n} x_i$.

Εικόνα 10: Πίνακας παραμέτρων Node impurity (Πηγή: http://spark.apache.org)

Η πληροφόρια που έχει αποκτηθεί (gain) είναι η διαφορά του node impurity στον μητρικό κόμβο και στο σταθμισμένο άθροισμα των δύο node impurities των κόμβων παιδιών. Η αναδρομική κατασκευή δέντρο σταματά σε έναν κόμβο, όταν μία από τις δύο παρακάτω προϋποθέσεις ισχύουν:

- Το βάθος του κόμβου είναι ίσο με την παράμετρο της κατάρτισης MaxDepth

- Η μη διάσπαση των δεδομένων (no split candidate) οδηγεί σε μία αποκτηθήσα πληροφορία του κόμβου.

2.2.3 Naïve Bayes

Ο Naive Bayes είναι ένας απλός multiclass αλγόριθμος ταξινόμησης με την υπόθεση της ανεξαρτησίας μεταξύ κάθε ζεύγους χαρακτηριστικών. Ο Naive Bayes μπορεί να εκπαιδευτεί πολύ αποτελεσματικά. Μέσα σε ένα μόνο πέρασμα στα δεδομένα εκπαίδευσης, υπολογίζει τη δεσμευμένη κατανομή πιθανότητας για κάθε χαρακτηριστικό δεδομένης ετικέτας, και στη συνέχεια εφαρμόζεται το θεώρημα του Bayes για τον υπολογισμό της υπό όρους κατανομής πιθανότητας της ετικέτας δεδομένης μιας παρατήρησης η οποία χρησιμοποιείται για την πρόβλεψη.

Όσον αφορά τη βιβλιοθήκη MLlib, υλοποιήσαμε multinomial naive Bayes, που συνήθως χρησιμοποιείται για την ταξινόμηση του εγγράφου. Μέσα σε αυτό το πλαίσιο, κάθε παρατήρηση είναι ένα έγγραφο, το καθένα χαρακτηριστικό αντιπροσωπεύει έναν όρο, των οποίων η αξία είναι η συχνότητα του όρου. Το προσθετικό εξομάλυνσης (additive smoothing) μπορεί να χρησιμοποιηθεί από τον καθορισμό της λ παραμέτρου (προεπιλογή έως 1.0).

Κεφάλαιο 3

Όπως προαναφέρθηκε στο πρώτο κεφάλαιο, η ανάγκη για διαχείριση τεράστιου όγκου δεδομένων έχει ως αποτέλεσμα την δημιουργία κατάλληλων εργαλείων όπως το spark το οποίο θα χρησιμοποιηθεί σε αυτή τη διπλωματική για την επίλυση συγκεκριμένων προβλημάτων που συναντάμε στις σύγχρονες επιχειρήσεις. Σε αυτό το κεφάλαιο θα παρουσιαστεί η μελέτη περίπτωσης μιας ελληνικής εταιρίας στον κλάδο των μεταφορών. Στόχος του συγκεκριμένου κεφαλαίου είναι να δείξει τον τρόπο με τον οποίο το συγκεκριμένο εργαλείο μπορεί να επιλύσει σημαντικά προβλήματα αυτής της εταιρείας, Τέτοια προβλήματα είναι η ομαδοποίηση των καταστημάτων της με βάση την αποδοτικότητα τους, η πρόβλεψη του κέρδους που μπορεί να επιφέρει ένας πελάτης.

3.1 Εφαρμογή σε δεδομένα μεταφορικής αλυσίδας

3.1.1 Συλλογή δεδομένων για το πρόβλημα της συσταδοποίησης των καταστημάτων

Τα δεδομένα αντλούνται από τις βάσεις δεδομένων της εταιρείας χρησιμοποιώντας ερωτήματα SQL και εξάγονται σε csv αρχείο.

Τα δεδομένα τα οποία συλλέχθηκαν είναι τα εξής:

α/α	Attributes	Επεξήγηση των attributes
1	**Γεωγραφική Θέση**	Περιλαμβάνει πληροφορίες σχετικές με την τοποθεσία των καταστημάτων.
2	**Παράπονο**	Περιλαμβάνει πληροφορίες σχετικά με τα παράπονα ανά αποστολή.
3	**Αριθμός πελατών**	Περιλαμβάνει πληροφορίες σχετικά το αριθμό των πελατών ανά κατάστημα.
4	**Σταθερότητα πελατών**	Περιλαμβάνει πληροφορίες σχετικά με την σταθερότητα του πελατολογίου ανά κατάστημα.
5	**Είδος πελάτη**	Πίνακας χαρακτηρισμού πελατών σε «Μετρητοίς» και «Με πίστωση».
6	**Ποσοστό παραδόσεων**	Περιλαμβάνει πληροφορίες σχετικά με το ποσοστό παράδοσης ανά κατάστημα.
7	**Λάθη άφιξης**	Περιλαμβάνει πληροφορίες για τις λάθος αφίξεις.
8	**Κατηγορία βάρους**	Περιλαμβάνει τις κατηγορίες βάρους των αποστολών.
9	**Δρομολόγιο**	Περιλαμβάνει τα Εντός Πόλης, Εντός Νομού, εντός περιοχής, Χερσαίους, Νησιωτικούς, Δυσπρόσιτους προορισμούς
10	**Διανομείς**	Περιλαμβάνει όλους τους διανομείς του δικτύου της εταιρίας.
11	**Loyalty**	Πελάτες με κάρτα Member
12	**Κατάστημα**	Κωδικός καταστήματος

Εικόνα 11: Πίνακας δεδομένων

Αυτά είναι τα χαρακτηριστικά των δεδομένων τα οποία αφορούν τα καταστήματα της εταιρείας και στα οποία θα γίνει επεξεργασία και ανάλυση.

3.1.2 Συλλογή δεδομένων για το πρόβλημα της πρόβλεψης του κέρδους που μπορεί να επιφέρει ένας πελάτης.

Τα δεδομένα και σε αυτό το πρόβλημα αντλούνται από τις βάσεις δεδομένων της εταιρείας χρησιμοποιώντας ερωτήματα SQL και εξάγονται σε csv αρχείο.

Τα δεδομένα τα οποία συλλέχθηκαν (28.419 εγγραφές) περιέχουν τα εξής χαρακτηριστικά - στήλες:

α/α	Attributes	Επεξήγηση των attributes
1	**ΑΦΜ πελάτη**	Περιλαμβάνει το ΑΦΜ του πελάτη
2	**Περιοχή Παραλαβής**	Περιλαμβάνει την περιοχή παραλαβής του κάθε πελάτη
3	**Κάρτα Μέλους**	Περιλαμβάνει την πληροφορία αν ο πελάτης έχει κάρτα μέλους (σημαντικές εκπτώσεις) ή όχι
4	**Ενεργός Πελάτης**	Περιλαμβάνει πληροφορίες σχετικά με το αν ένας πελάτης είναι ενεργός (συχνές αποστολές) ή μη ενεργός (έχει χρησιμοποιήσει τις υπηρεσίες της εταιρείας ελάχιστες φορές).
5	**Όροι πλρωμής**	Πίνακας χαρακτηρισμού πελατών σε «Μετρητοίς» και «Με πίστωση».
6	**Αριθμός Αποστολών Τιμολόγησης**	Περιλαμβάνει το σύνολο των αποστολών που έχουν τιμολογηθεί για κάθε πελάτη
7	**Βάρος τιμολόγησης**	Περιλαμβάνει το σύνολο του βάρους των αποστολών που έχουν τιμολογηθεί για κάθε πελάτη
8	**Μέσο Βάρος**	Περιλαμβάνει το μέσο βάρος, το οποίο είναι το βάρος των αποστολών του πελάτη σε σχέση με τον αριθμό των αποστολών με χρέωση
9	**Μέσος ημερήσιος τζίρος**	Περιλαμβάνει το μέσο ημερήσιο έσοδο ανά πελάτη
10	**Αξία Παραγωγής**	Περιλαμβάνει το συνολικό έσοδο από κάθε πελάτη

Εικόνα 12: Σύνολο δεδομένων

Αυτά είναι τα χαρακτηριστικά των δεδομένων τα οποία αφορούν τους πελάτες της εταιρείας και στα οποία θα γίνει επεξεργασία και ανάλυση.

3.2. Καθαρισμός και φόρτωση στο spark

Πολύ συχνά παρατηρείται ότι οι τιμές των χαρακτηριστικών (attributes) των δεδομένων που θέλουμε να αναλύσουμε είναι λανθασμένες (θόρυβο), ελλιπείς (λείπουν τιμές) και ασυνεπείς. Ο θόρυβος συνήθως προέρχεται από ανθρώπινα λάθη και σπανίως από λάθη του υπολογιστή. Τα ελλιπή δεδομένα συνήθως δημιουργούνται από τη μη εισαγωγή κάποιων στοιχείων επειδή τη στιγμή της εισαγωγής δεν έχουν αξία. Τα ασυνεπή δεδομένα εμφανίζονται εξαιτίας των ενοποιήσεων των δεδομένων από διαφορετικές βάσεις όπου τα ίδια στοιχεία μπορεί να έχουν διαφορετικό όνομα στην κάθε βάση.

3.2.1 Διαδικασία Καθαρίσματος Δεδομένων (Data Cleaning)

Τρόποι καθαρισμού δεδομένων:

- Αγνόηση τιμής κάποιου χαρακτηριστικού

- Αντικατάσταση των τιμών που λείπουν με κάποια συγκεκριμένη σταθερά

- Τοποθέτηση τιμών με το χέρι (δεν προτείνεται σε μεγάλο όγκο δεδομένων γιατί είναι υπορβολικά χρονοβόρα μέθοδος)

- Υπολογισμός του μέσου όρου ενός χαρακτηριστικού και αντικατάσταση των ελλιπών τιμών με την τιμή του μέσο όρου.

- Μπορούν να χρησιμοποιηθούν αλγόριθμοι κατηγοριοποίησης (regression) και αλγόριθμοι συσταδοποίησης για να βρεθούν οι λανθασμένες τιμές και να αφαιρεθούν από τα δεδομένα μας

3.2.2 Εφαρμογή καθαρισμού δεδομένων για το πρόβλημα της συσταδοποίησης των καταστημάτων

Για να μπορέσουμε να εφαρμόσουμε τον αλγόριθμο K-means θα πρέπει να φέρουμε σε κατάλληλη μορφή τα δεδομένα μας (data cleaning), όπως για παράδειγμα να αφαιρέσουμε τυχόν λανθασμένες εγγραφές, να μετασχηματίσουμε κάποια attributes εφόσον είναι αναγκαίο, να αφαιρέσουμε περιγραφικά attributes καθώς ο K-means δεν δέχεται τίποτα άλλο εκτός από αριθμούς, να αλλάξουμε ή να αφαιρέσουμε τυχόν λανθασμένες τιμές π.χ. χρόνος παράδοσης: -10 λεπτά (αρνητική τιμή για το χρόνο).

Μετά την εφαρμογή της παραπάνω διαδικασίας καταλήξαμε στις κατηγορίες δεδομένων του παρακάτω πίνακα όπου σχημάτισαν 550 εγγραφές.

Στα δεδομένα εφαρμόστηκαν κανόνες ανωνυμίας για την διασφάλιση των προσωπικών δεδομένων της εταιρείας και των πελατών της.

Δεδομένα (CSV): 550 εγγραφές

α/α	Χαρακτηριστικά (Attributes)	Περιγραφή των attributes
1	**ID**	Μοναδικό ID για κάθε record
2	**Γεωγραφική Θέση (Περιοχή)**	Περιλαμβάνει πληροφορίες σχετικές με την τοποθεσία των καταστημάτων.
3	**Κατάστημα**	Κωδικός καταστήματος
5	**Ταχύτητα παράδοσης**	Περιλαμβάνει πληροφορίες σχετικά με τον χρόνο παράδοσης.
6	**Παραγωγή**	Αριθμός αποστολών
7	**Παραδόσεις**	Αριθμός Παραδόσεων
8	**Επιστροφές**	Αριθμός επιστροφών
9	**Ποσοστό παραδόσεων**	Περιλαμβάνει πληροφορίες σχετικά με το ποσοστό παράδοσης ανά κατάστημα.

Εικόνα 13: Πίνακας δεδομένων μετά την εκκαθάριση

3.2.3 Εφαρμογή καθαρισμού δεδομένων για το πρόβλημα της κατηγοριοποίησης των πελατών

Επαναλαμβάνοντας την ίδια διαδικασία με το προηγούμενο πρόβλημα και για να εφαρμόσουμε τους αλγορίθμους κατηγοριοποίησης θα πρέπει να φέρουμε σε κατάλληλη μορφή τα δεδομένα μας (data cleaning), όπως για παράδειγμα να αφαιρέσουμε τυχόν λανθασμένες τιμές, να μετασχηματίσουμε τα attributes αν είναι υποχρεωτικό.

Για διευκόλυνση της διαδικασίας της ανάλυσης κωδικοποιήσαμε τα εξής χαρακτηριστικά των πελατών:

- Στην κατηγορία Κάρτα Μέλους οι πελάτες που έχουν κάρτα μέλους έχουν την τιμή 1 και αυτοί που δεν έχουν την τιμή 0.
- Στην κατηγορία Ενεργός Πελάτης οι πελάτες που είναι ενεργοί την τιμή 1 και οι πελάτες που δεν είναι ενεργοί την τιμή 0.
- Στην κατηγορία Όροι Πληρωμής οι πελάτες που είναι με πίστωση πήραν την τιμή 1, ενώ οι υπόλοιποι που είναι στην κατηγορία μετρητοίς πήραν την τιμή 0.

Μετά την εφαρμογή της παραπάνω διαδικασίας καταλήξαμε στις κατηγορίες δεδομένων του παρακάτω πίνακα όπου σχημάτισαν 28.419 εγγραφές.

Στα δεδομένα εφαρμόστηκαν και εδώ κανόνες ανωνυμίας για την διασφάλιση των προσωπικών δεδομένων της εταιρείας και των πελατών της.

Δεδομένα (CSV): 28.419 εγγραφές

α/α	Attributes	Επεξήγηση των attributes
1	**Κάρτα Μέλους**	Περιλαμβάνει την πληροφορία αν ο πελάτης έχει κάρτα μέλους (σημαντικές εκπτώσεις) ή όχι
2	**Ενεργός Πελάτης**	Περιλαμβάνει πληροφορίες σχετικά με το αν ένας πελάτης είναι ενεργός (συχνές αποστολές) ή μη ενεργός (έχει χρησιμοποιήσει τις υπηρεσίες της εταιρείας ελάχιστες φορές).
3	**Όροι Πληρωμής**	Πίνακας χαρακτηρισμού πελατών σε «Μετρητοίς» και «Με πίστωση».
4	**Αριθμός Αποστολών Τιμολόγησης**	Περιλαμβάνει το σύνολο των αποστολών που έχουν τιμολογηθεί για κάθε πελάτη

α/α	Attributes	Επεξήγηση των attributes
5	**Βάρος Τιμολόγησης**	Περιλαμβάνει το σύνολο του βάρους των αποστολών που έχουν τιμολογηθεί για κάθε πελάτη
6	**Μέσο Βάρος**	Περιλαμβάνει το μέσο βάρος, το οποίο είναι το βάρος των αποστολών του πελάτη σε σχέση με τον αριθμό των αποστολών με χρέωση
7	**Μέσος Ημερήσιος Τζίρος**	Περιλαμβάνει το μέσο ημερήσιο έσοδο ανά πελάτη
8	**Αξία Παραγωγής**	Περιλαμβάνει το συνολικό έσοδο από κάθε πελάτη

Εικόνα 14: Σύνολο δεδομένων μετά από εκκαθάριση

3.3 Πρόβλημα συσταδοποίησης

3.3.1 Το πρόγραμμα ανάλυσης για τον kmeans

Στο ακόλουθο πείραμα, μετά τη φόρτωση και ανάλυση των δεδομένων, χρησιμοποιούμε τον KMeans για να δημιουργήσουμε τις συστάδες των δεδομένων. Ο αριθμός των επιθυμητών συστάδων περνιέται στον αλγόριθμο. Στη συνέχεια υπολογίζουμε το συνολικό άθροισμα των τετραγώνων σφαλμάτων (WSSSE). Αυξομειώνοντας την παράμετρο k μπορούμε να μειώσουμε αυτό το σφάλμα. [14]

Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η scala[15] (1.4.1) σε περιβάλλον εκτέλεσης spark-shell.

Στη συνέχεια και σύμφωνα με τα παραπάνω ακολουθούμε την παρακάτω διαδικασία:

```
// Φορτώνουμε τις απαιτούμενες βιβλιοθήκες

import org.apache.spark.mllib.linalg.Vectors

import org.apache.spark.mllib.clustering.KMeans

import org.apache.spark.sql.functions._

//Φόρτωση του αρχείου και αφαίρεση των επικεφαλίδων

val data = sc.textFile("C:/Users/adamopoulos/Desktop/thesis/Fdata.txt")

val header = data.first

val rows = data.filter(l => l != header)

// Ορισμός κλάσης
```

[14] https://spark.apache.org/docs/1.4.1/mllib-clustering.html
[15] http://www.artima.com/weblogs/viewpost.jsp?thread=163733

case class CC1(ID: String, PERIOXI: String, KATASTIMA_PROORISMOU_KOD: String, DIAKINISI_SE: Double, ARITHMOS_APOSTOLON: Double, PARADOSEIS: Double, EPISTROFES: Double, POSOSTO_PARADOSEON: Double)

// Ορίζουμε με πιο σύμβολο γίνεται ο διαχωρισμός στα δεδομένα

val allSplit = rows.map(line => line.split(","))

// Ορίζουμε τα δεδομένα μας με βάση την κλάση μας σε χάρτη (map)

val allData = allSplit.map(p => CC1(p(0).toString, p(1).toString, p(2).toString, p(3).trim.toDouble, p(4).trim.toDouble, p(5).trim.toDouble, p(6).trim.toDouble, p(7).trim.toDouble))

// Μετασχηματισμός των δεδομένων μας σε dataframe

val allDF = allData.toDF()

// Μετασχηματισμός των δεδομένων πάλι σε rdd και τα εισάγουμε σε μνήμη cache

val rowsRDD = allDF.rdd.map(r => (r.getString(0), r.getString(1), r.getString(2), r.getDouble(3), r.getDouble(4), r.getDouble(5), r.getDouble(6), r.getDouble(7)))

rowsRDD.cache()

// Μετατροπή των δεδομένων σε RDD που θα εισαχθούν στον αλγόριθμο kmeans και τα εισάγουμε στην μνήμη cache.

val vectors = allDF.rdd.map(r => Vectors.dense(r.getDouble(3), r.getDouble(4), r.getDouble(5), r.getDouble(6), r.getDouble(7)))

vectors.cache()

// Μοντέλο kmeans με 3 clusters και 20 επαναλήψεις

val kMeansModel = KMeans.train(vectors, 3, 20)

// Εμφάνιση των κέντρων των clusters

kMeansModel.clusterCenters.foreach(println)

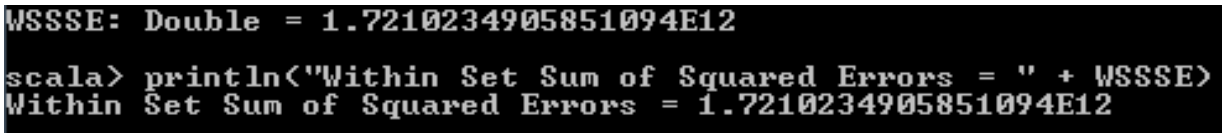
```
scala> kMeansModel.clusterCenters.foreach(println)
[2.4710920770877944,10688.241970021414,10053.71948608137,634.5224839400429,0.6044164882226981]
[1.0,442260.25,442087.9,172.35000000000002,0.9996450000000001]
[1.129032258064516,160836.09677419355,160383.51612903224,452.5806451612903,0.9963129032258063]
```

Εικόνα 15: Κέντρα των clusters

// Εμφάνιση του αθροίσματος των τετραγωνικών λαθών

val WSSSE = kMeansModel.computeCost(vectors)

println("Within Set Sum of Squared Errors = " + WSSSE)



Εικόνα 16: Άθρισμα τετραγωνικών λαθών

// Παίρνουμε την πρόβλεψη από το μοντέλο με το ID ώστε να μπορούμε να συνδέσουμε τον πίνακά μας με άλλους πίνακες

val predictions = rowsRDD.map{r => (r._1, kMeansModel.predict(Vectors.dense(r._4, r._5, r._6, r._7, r._8)))}

// Μετασχηματισμός των RDD σε dataframe

val predDF = predictions.toDF("ID","CLUSTER")

Εξήγηση του κώδικα:
Ο κώδικας εισάγει κάποιες μεθόδους για την Vector, KMeans και SQL απαραίτητες για το πρόγραμμα. Στη συνέχεια φορτώνει το αρχείο .csv(data) από το δίσκο και καταργεί την κεφαλίδα που περιέχει περιγραφές της κάθε στήλης . Στη συνέχεια ορίζεται μια κλάση (class) ,γίνεται τη διάσπαση των στηλών με κόμμα και χαρτογραφούνται (map) τα δεδομένα με βάση την κλάση που έχει οριστεί. Μετατρέπονται τα RDD σε dataframe . Χαρτογραφήσει τα dataframe σε RDD και προσωρινά αποθηκεύονται τα δεδομένα στην μνήμη cache. Δημιουργείται RDD για τις 5 στήλες που θέλουμε να περάσει ο αλγόριθμος KMeans και αποθηκεύονται τα δεδομένα στην μνήμη cache . Η προσωρινή αποθήκευση συμβάλλει στην επιτάχυνση των επιδόσεων . Τρέχουμε το kMeansModel θέτοντας τρεις συστάδες και 20 επαναλήψεις . Εκτυπώνονται τα κέντρα για όλες τις συστάδες. Τώρα που το μοντέλο έχει δημιουργηθεί , παίρνουμε τις προβλέψεις μας για τις συστάδες με ένα ID (αναγνωριστικό) , έτσι ώστε να υπάρχει η δυνατότητα να προσδιοριστεί μονοσήμαντα κάθε περίπτωση με κάθε σύμπλεγμα που θα ανατεθεί . Τέλος μετατρέπεται πάλι σε μια dataframe για να αναλυθεί.

Στην παρακάτω εικόνα 16: βλέπετε ένα υποσύνολο του allDF dataframe με τα δεδομένα.

allDF.show()

ID	PERIOXI	KATASTIMA_PROORISMOU_KOD	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON
1	ACS	ATH	1.0	894828.0	894667.0	161.0	0.9998
2	ACS	ATH	2.0	91962.0	89985.0	1977.0	0.9785
3	ACS	ATH	3.0	384.0	235.0	149.0	0.612
4	ACS	ATH	4.0	14.0	9.0	5.0	0.6429
5	ACS	THS	1.0	23360.0	23332.0	28.0	0.9988
6	ACS	THS	2.0	365.0	56.0	309.0	0.1534
7	ACS	THS	3.0	56.0	2.0	54.0	0.0357
8	ACS	KY	1.0	700.0	700.0	0.0	1.0
9	ACS	KY	2.0	30.0	30.0	0.0	1.0
10	ATTIKIS	AA	1.0	163272.0	163249.0	23.0	0.9999
11	ATTIKIS	AA	2.0	103405.0	102188.0	1217.0	0.9882
12	ATTIKIS	AA	3.0	3056.0	2854.0	202.0	0.9339
13	ATTIKIS	AA	4.0	3.0	0.0	3.0	0.0
14	ATTIKIS	AB	1.0	326844.0	326834.0	10.0	1.0
15	ATTIKIS	AB	2.0	49042.0	47187.0	1855.0	0.9622
16	ATTIKIS	AB	3.0	4806.0	4536.0	270.0	0.9438
17	ATTIKIS	AB	4.0	9.0	0.0	9.0	0.0
18	ATTIKIS	AD	1.0	521783.0	521716.0	67.0	0.9999
19	ATTIKIS	AD	2.0	18555.0	14521.0	4034.0	0.7826
20	ATTIKIS	AD	3.0	276.0	32.0	244.0	0.1159

Εικόνα 17: Υποσύνολο δεδομένων

Στην εικόνα 17 βλέπετε ένα υποσύνολο του predDF dataframe με το ID (αναγνωριστικό) και το σύμπλεγμα . Έχουμε τώρα ένα μοναδικό ID (αναγνωριστικό) και σε ποιο cluster (0, 1, 2) ανήκει.

predDF.show()

ID	CLUSTER
1	1
2	2
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	2
11	2
12	0
13	0
14	1
15	0
16	0
17	0
18	1
19	0
20	0

Εικόνα 18: Υποσύνολο δεδομένων με το ID

Επειδή allDF και predDF dataframes έχουν μία κοινή στήλη (ID) γίνεται join των πινάκων για να γίνει περεταίρω ανάλυση.

//Κάνουμε join τα dataframes με το ID

```
val t = allDF.join(predDF,"ID")
```

Τώρα όλα τα δεδομένα μας είναι συνδεδεμένα με τα αντίστοιχα clusters. Στην εικόνα 18 βλέπετε ένα υποσύνολο των δεδομένων και σε ποιο cluster ανήκει.

```
t.show()
```

ID	PERIOXI	KATASTIMA_PROORISMOU_KOD	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
286	THESSALONIKIS	THK	3.0	91.0	31.0	60.0	0.3407	0
330	KENTRIKI ELLADA	EB	2.0	1889.0	1735.0	154.0	0.9185	0
448	NISIA	LI	3.0	46.0	4.0	42.0	0.087	0
287	THESSALONIKIS	THK	4.0	1.0	0.0	1.0	0.0	0
331	KENTRIKI ELLADA	EB	3.0	24.0	10.0	14.0	0.4167	0
449	NISIA	LI	4.0	2.0	0.0	2.0	0.0	0
170	BOREIA ELLADA	DR	1.0	172404.0	172387.0	17.0	0.9999	2
288	THESSALONIKIS	THL	1.0	18100.0	18092.0	8.0	0.9996	0
332	KENTRIKI ELLADA	EN	1.0	29716.0	29712.0	4.0	0.9999	0
171	BOREIA ELLADA	DR	2.0	16911.0	14044.0	2867.0	0.8305	0
289	THESSALONIKIS	THL	2.0	911.0	862.0	49.0	0.9462	0
333	KENTRIKI ELLADA	EN	2.0	15470.0	14886.0	584.0	0.9622	0
172	BOREIA ELLADA	DR	3.0	75.0	36.0	39.0	0.48	0
334	KENTRIKI ELLADA	EN	3.0	2408.0	2059.0	349.0	0.8551	0
173	BOREIA ELLADA	THA	1.0	21723.0	21721.0	2.0	0.9999	0
335	KENTRIKI ELLADA	EN	4.0	2.0	2.0	0.0	1.0	0
174	BOREIA ELLADA	THA	2.0	541.0	278.0	263.0	0.5139	0
336	KENTRIKI ELLADA	THB	1.0	48907.0	48905.0	2.0	1.0	0
175	BOREIA ELLADA	THA	3.0	12.0	1.0	11.0	0.0833	0
337	KENTRIKI ELLADA	THB	2.0	5772.0	4511.0	1261.0	0.7815	0

Εικόνα 19: Συστάδες στις οποίες ανήκουν τα δεδομένα

// Εμφάνιση κάθε cluster ξεχωριστά

```
t.filter("CLUSTER = 0").show()
```

ID	PERIOXI	KATASTIMA_PROORISMOU_KOD	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
286	THESSALONIKIS	THK	3.0	91.0	31.0	60.0	0.3407	0
330	KENTRIKI ELLADA	EB	2.0	1889.0	1735.0	154.0	0.9185	0
448	NISIA	LI	3.0	46.0	4.0	42.0	0.087	0
287	THESSALONIKIS	THK	4.0	1.0	0.0	1.0	0.0	0
331	KENTRIKI ELLADA	EB	3.0	24.0	10.0	14.0	0.4167	0
449	NISIA	LI	4.0	2.0	0.0	2.0	0.0	0
288	THESSALONIKIS	THL	1.0	18100.0	18092.0	8.0	0.9996	0
332	KENTRIKI ELLADA	EN	1.0	29716.0	29712.0	4.0	0.9999	0
171	BOREIA ELLADA	DR	2.0	16911.0	14044.0	2867.0	0.8305	0
289	THESSALONIKIS	THL	2.0	911.0	862.0	49.0	0.9462	0
333	KENTRIKI ELLADA	EN	2.0	15470.0	14886.0	584.0	0.9622	0
172	BOREIA ELLADA	DR	3.0	75.0	36.0	39.0	0.48	0
334	KENTRIKI ELLADA	EN	3.0	2408.0	2059.0	349.0	0.8551	0
173	BOREIA ELLADA	THA	1.0	21723.0	21721.0	2.0	0.9999	0
335	KENTRIKI ELLADA	EN	4.0	2.0	2.0	0.0	1.0	0
174	BOREIA ELLADA	THA	2.0	541.0	278.0	263.0	0.5139	0
336	KENTRIKI ELLADA	THB	1.0	48907.0	48905.0	2.0	1.0	0
175	BOREIA ELLADA	THA	3.0	12.0	1.0	11.0	0.0833	0
337	KENTRIKI ELLADA	THB	2.0	5772.0	4511.0	1261.0	0.7815	0
220	BOREIA ELLADA	SR	4.0	5.0	0.0	5.0	0.0	0

Εικόνα 20: Υποσύνολο δεδομένων συστάδας 0

t.filter("CLUSTER = 1").show()

ID	PERIOXI	KATASTIMA_PROORISMOU_KOD	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
1	ACS	ATH	1.0	894828.0	894667.0	161.0	0.9998	1
63	ATTIKIS	BE	1.0	521931.0	521904.0	27.0	0.9999	1
74	ATTIKIS	BTH	1.0	484076.0	484063.0	13.0	1.0	1
78	ATTIKIS	BI	1.0	406042.0	405652.0	390.0	0.999	1
18	ATTIKIS	AD	1.0	521783.0	521716.0	67.0	0.9999	1
26	ATTIKIS	AZ	1.0	479900.0	478977.0	923.0	0.9981	1
93	ATTIKIS	BX	1.0	717158.0	717000.0	158.0	0.9998	1
97	ATTIKIS	BS	1.0	550061.0	549996.0	65.0	0.9999	1
46	ATTIKIS	AX	1.0	599715.0	598591.0	1124.0	0.9981	1

Εικόνα 21: Υποσύνολο δεδομένων συστάδας 1

t.filter("CLUSTER = 2").show()

ID	PERIOXI	KATASTIMA_PROORISMOU_KOD	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
170	BOREIA ELLADA	DR	1.0	172404.0	172387.0	17.0	0.9999	2
176	BOREIA ELLADA	KG	1.0	252038.0	251800.0	238.0	0.9991	2
339	KENTRIKI ELLADA	KA	1.0	144586.0	144580.0	6.0	1.0	2
225	DYTIKI ELLADA	AG	1.0	227149.0	227129.0	20.0	0.9999	2
53	ATTIKIS	BA	1.0	175196.0	175185.0	11.0	0.9999	2
54	ATTIKIS	BA	2.0	137436.0	132846.0	4590.0	0.9666	2
291	THESSALONIKIS	THM	1.0	326520.0	326496.0	24.0	0.9999	2
56	ATTIKIS	BB	1.0	318546.0	318542.0	4.0	1.0	2
57	ATTIKIS	BB	2.0	120585.0	117599.0	2986.0	0.9752	2
294	THESSALONIKIS	THN	1.0	134522.0	134514.0	8.0	0.9999	2
187	BOREIA ELLADA	KM	1.0	151305.0	151291.0	14.0	0.9999	2
349	KENTRIKI ELLADA	LM	1.0	173642.0	173635.0	7.0	1.0	2
235	DYTIKI ELLADA	IB	1.0	196098.0	196078.0	20.0	0.9999	2
121	ATTIKIS	GL	1.0	315618.0	315567.0	51.0	0.9998	2
239	DYTIKI ELLADA	IN	1.0	189483.0	189212.0	271.0	0.9986	2
469	NISIA	RD	1.0	247222.0	247172.0	50.0	0.9998	2
514	PELOPONNISOS	KL	1.0	292975.0	292967.0	8.0	1.0	2
517	PELOPONNISOS	KO	1.0	260916.0	260891.0	25.0	0.9999	2
194	BOREIA ELLADA	KT	1.0	185236.0	185228.0	8.0	1.0	2
356	KENTRIKI ELLADA	LR	1.0	243386.0	243264.0	122.0	0.9995	2

Εικόνα 22: Υποσύνολο δεδομένων συστάδας 2

Παρακάτω εμφανίζονται τα στατιστικά στοιχεία για κάθε μία από τις συστάδες (clusters). Ο κώδικας μας δίνει την πλήθος, την μέση τιμή , την τυπική απόκλιση , το ελάχιστο και το μέγιστο για όλες τις αριθμητικές τιμές του dataframe.

// Υπολογισμός στατιστικών στοιχείων για κάθε συστάδα

```
t.filter("CLUSTER = 0").describe().show()
```

summary	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
count	488	488	488	488	488	488
mean	2.415983606557377	14499.25	13865.78893442623	633.4610655737705	0.621172540983607	0.0
stddev	0.9819637065812211	25889.97817370731	25669.133340999113	1384.9832700506445	0.36860096558198135	0.0
min	1.0	1.0	0.0	0.0	0.0	0
max	4.0	116918.0	116900.0	11398.0	1.0	0

Εικόνα 23: Στατιστικά στοιχεία συστάδας 0

```
t.filter("CLUSTER = 1").describe().show()
```

summary	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
count	9	9	9	9	9	9
mean	1.0	575054.8888888889	574729.5555555555	325.3333333333333	0.9993888888888888	1.0
stddev	0.0	139538.11178666446	139586.3481856394	390.91431286152726	7.430334889524386E-4	0.0
min	1.0	406042.0	405652.0	13.0	0.9981	1
max	1.0	894828.0	894667.0	1124.0	1.0	1

Εικόνα 24: Στατιστικά στοιχεία συστάδας 1

t.filter("CLUSTER = 2").describe().show()

summary	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
count	52	52	52	52	52	52
mean	1.0769230769230769	222256.23076923078	221952.92307692306	303.3076923076923	0.9980788461538459	2.0
stddev	0.26646935501059676	73592.92317738773	73809.33597474343	914.0741751264698	0.006360918430921094	0.0
min	1.0	120585.0	117599.0	1.0	0.9666	2
max	2.0	364853.0	364826.0	4590.0	1.0	2

Εικόνα 25: Στατιστικά στοιχεία συστάδας 2

Έτσι λοιπόν από την παραπάνω ανάλυση, παρατηρείται ότι έχουν δημιουργηθεί τρεις συστάδες, όπου η συστάδα 0 είναι η μεγαλύτερη σε πλήθος 488, η συστάδα 1 είναι η μικρότερη με πλήθος μόνο 9 και τέλος η συστάδα 2 με πλήθος 52. Επίσης διακρίνεται, ότι στην συστάδα 1 ο μέσος όρος στο ποσοστό των παραδόσεων είναι 0.99 σχεδόν 100%, με μέσο όρο αριθμό αποστολών 575054.88 και με αριθμό επιστροφών 325.33. Στην συστάδα 2 βλέπουμε, ότι το ποσοστό των παραδόσεων είναι και εδώ πολύ υψηλό 0.99 αλλά με μέσο όρο αριθμό αποστολών στις 222256.23 και αριθμό επιστροφών 303.30. Τέλος στην συστάδα 0 παρατηρείται ότι μέσος όρος το ποσοστό παράδοσης είναι στο 0.62 με μέσο όρο αριθμών αποστολών 14499.25 πολύ χαμηλό αριθμό σε σχέση με τις προηγούμενες συστάδες, σχεδόν με τον διπλάσιο αριθμό επιστροφών 633.46, όπως επίσης και μεγαλύτερο αριθμό ημερών διακίνησης 2.4 σε σχέση με τις υπόλοιπες συστάδες που έχουν μέσο όρο 1 και 1.07.

Στην συνέχεια εφόσον έγινε κανονικοποίηση(nolmalization) των δεδομένων μας εφαρμόσαμε την ίδια διαδικασία ανάλυσης, χρησιμοποιώντας και εδώ τον αλγόριθμο kmeans για 3 clusters και 20 επαναλήψεις.

Τα στατιστικά στοιχεία των τριών συστάδων που παράχθηκαν είναι:

summary	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
count	318	318	318	318	318	318
mean	1.5	0.08325335849056606	0.08220829559748428	0.08314420754716988	0.885899883647799	0.0
stddev	0.5	0.13536884836410829	0.13570943278228195	0.14639564134830255	0.18890861812351042	0.0
min	1.0	3.0E-6	3.0E-6	0.0	0.006353	0
max	2.0	1.0	1.0	1.0	1.0	0

Εικόνα 26: Στατιστικά στοιχεία συστάδας 1

summary	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
count	74	74	74	74	74	74
mean	4.0	2.2E-5	4.635135135135136...	0.001397986486486...	0.1520666351351351	1.0
stddev	0.0	4.268552511161195E-5	2.103765884899276E-5	0.002591419414316...	0.30670491613264766	0.0
min	4.0	0.0	0.0	0.0	0.0	1
max	4.0	2.45E-4	1.74E-4	0.014477	1.0	1

Εικόνα 27: Στατιστικά στοιχεία συστάδας 2

summary	DIAKINISI_SE	ARITHMOS_APOSTOLON	PARADOSEIS	EPISTROFES	POSOSTO_PARADOSEON	CLUSTER
count	157	157	157	157	157	157
mean	3.0	8.276942675159233E-4	6.500000000000001E-4	0.013884496815286616	0.4525961656050955	2.0
stddev	0.0	0.001939682253387721	0.0018218495991116 94	0.018515611668540176	0.2947361571127047	0.0
min	3.0	0.0	0.0	0.0	0.0	2
max	3.0	0.0139	0.013407	0.092568	1.0	2

Εικόνα 28: Στατιστικά στοιχεία συστάδας 2

Έτσι βλέπουμε την παραγωγή τριών συστάδων, με τελείως διαφορετική συσταδοποίηση σε σχέση με το την πρώτη ανάλυση στην οποία δεν είχαμε κάνει κανονικοποίηση στα δεδομένα. Η μετατροπή των δεδομένων μας στην κατάλληλη μορφή, για να βγουν όσο πιο σωστά αλλά και αποδοτικά τα αποτελέσματα, είναι ένα πολύ σημαντικό κομμάτι πρωτού εφαρμόσουμε τους αλγοριθμους εξόρυξης.

3.4 Πρόβλημα πρόβλεψης του κέρδους των πελατών

3.4.1 Εισαγωγή

Στη συνέχεια, θα αναλύσουμε το πρόβλημα της πρόβλεψης του κέρδους που μπορεί να επιφέρει ένας πελάτης. Θα χρησιμοποιήσουμε τη ίδια μεθοδολογία με το προηγούμενο πρόβλημα, όσον αφορά τη συλλογή δεδομένων, τη διαδικασία καθαρισμού των δεδομένων (data cleaning) και εφαρμογή αλγορίθμων κατηγοριοποίησης.[16]

3.4.2 Το πρόγραμμα ανάλυσης Decision Tree (Classification)

Το παρακάτω παράδειγμα δείχνει πώς να φορτώσουμε ένα αρχείο CSV, να το αναλύσουμε ως RDD του LabeledPoint και, στη συνέχεια, να εκτελέσουμε την ταξινόμηση, χρησιμοποιώντας ένα δέντρο απόφασης με χρήση Gini node impurity και μέγιστο βάθος δέντρου (maxdepth) 5. Το σφάλμα εκπαίδευσης υπολογίζεται για τη μέτρηση της ακρίβειας του αλγορίθμου.

```
// Φορτώνουμε τις απαιτούμενες βιβλιοθήκες
import org.apache.spark.SparkContext
import org.apache.spark.mllib.tree.DecisionTree
import org.apache.spark.mllib.regression.LabeledPoint
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.tree.configuration.Algo._
import org.apache.spark.mllib.tree.impurity.Gini
//Φόρτωση του αρχείου
val data = sc.textFile("C:/Users/adamopoulos/Desktop/thesis/dataClas.txt")

val parsedData = data.map { line =>
val parts = line.split(' ').map(_.toDouble)
LabeledPoint(parts(0), Vectors.dense(parts.tail))
}
```

[16] https://spark.apache.org/docs/1.4.1/mllib-clustering.html

// Τρέχουμε το μοντέλο decisiontree (classification) με maxDepth = 5

val maxDepth = 5

val model = DecisionTree.train(parsedData, Classification, Gini, maxDepth)

```
17/01/31 10:39:19 INFO RandomForest:   init: 1.30607605
  total: 2.821231731
  findSplitsBins: 0.614550349
  findBestSplits: 1.503087619
  chooseSplits: 1.498098601
maxDepth: Int = 5
```

Εικόνα 29: Μοντέλο Δένδρου Απόφασης

// Αξιολόγηση του μοντέλου που δημιουργήθηκε

val labelAndPreds = parsedData.map { point =>

val prediction = model.predict(point.features)

(point.label, prediction)

}

val trainErr = labelAndPreds.filter(r => r._1 != r._2).count.toDouble / parsedData.count

println("Training Error = " + trainErr)

```
17/01/31 10:39:56 INFO DAGScheduler: Job 9 finished: count at <console>:39, took 0,053302 s
Training Error = 0.14694394595165206
trainErr: Double = 0.14694394595165206
```

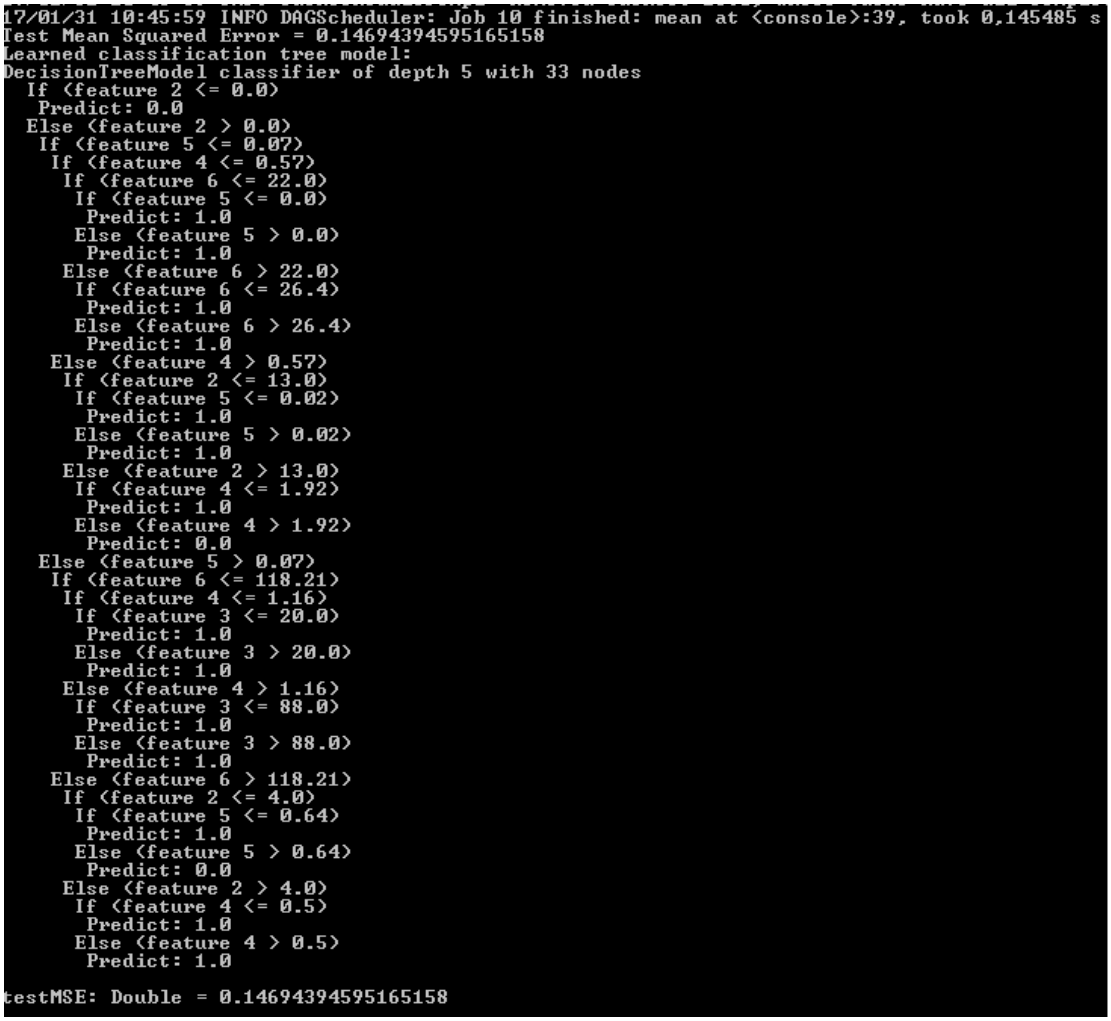
Εικόνα 30: Εμφάνιση ποσοστού λαθών

// Εμφάνιση του μοντέλου

val testMSE = labelAndPreds.map{ case (v, p) => math.pow(v - p, 2) }.mean()

println("Test Mean Squared Error = " + testMSE)

println("Learned regression tree model:\n" + model.toDebugString)



Εικόνα 31: Εμφάνιση μοντέλου δένδρου απόφασης

Το πρόγραμμα ανάλυσης Decision Tree (Regression)

```
// Φορτώνουμε τις απαιτούμενες βιβλιοθήκες
import org.apache.spark.SparkContext
import org.apache.spark.mllib.tree.DecisionTree
import org.apache.spark.mllib.regression.LabeledPoint
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.tree.configuration.Algo._
import org.apache.spark.mllib.tree.impurity.Variance

// Φόρτωση του αρχείου
val data = sc.textFile("C:/Users/adamopoulos/Desktop/thesis/dataClas.txt")
val parsedData = data.map { line =>
val parts = line.split(' ').map(_.toDouble)
```

```
LabeledPoint(parts(0), Vectors.dense(parts.tail))
}

// Τρέχουμε το μοντέλο decisiontree (Regression) με maxDepth = 5
val maxDepth = 5
val model = DecisionTree.train(parsedData, Regression, Variance, maxDepth)
```

```
17/01/31 10:47:54 INFO RandomForest:   init: 0.258727202
  total: 1.025771454
  findSplitsBins: 0.156340243
  findBestSplits: 0.753516276
  chooseSplits: 0.753088939
maxDepth: Int = 5
```

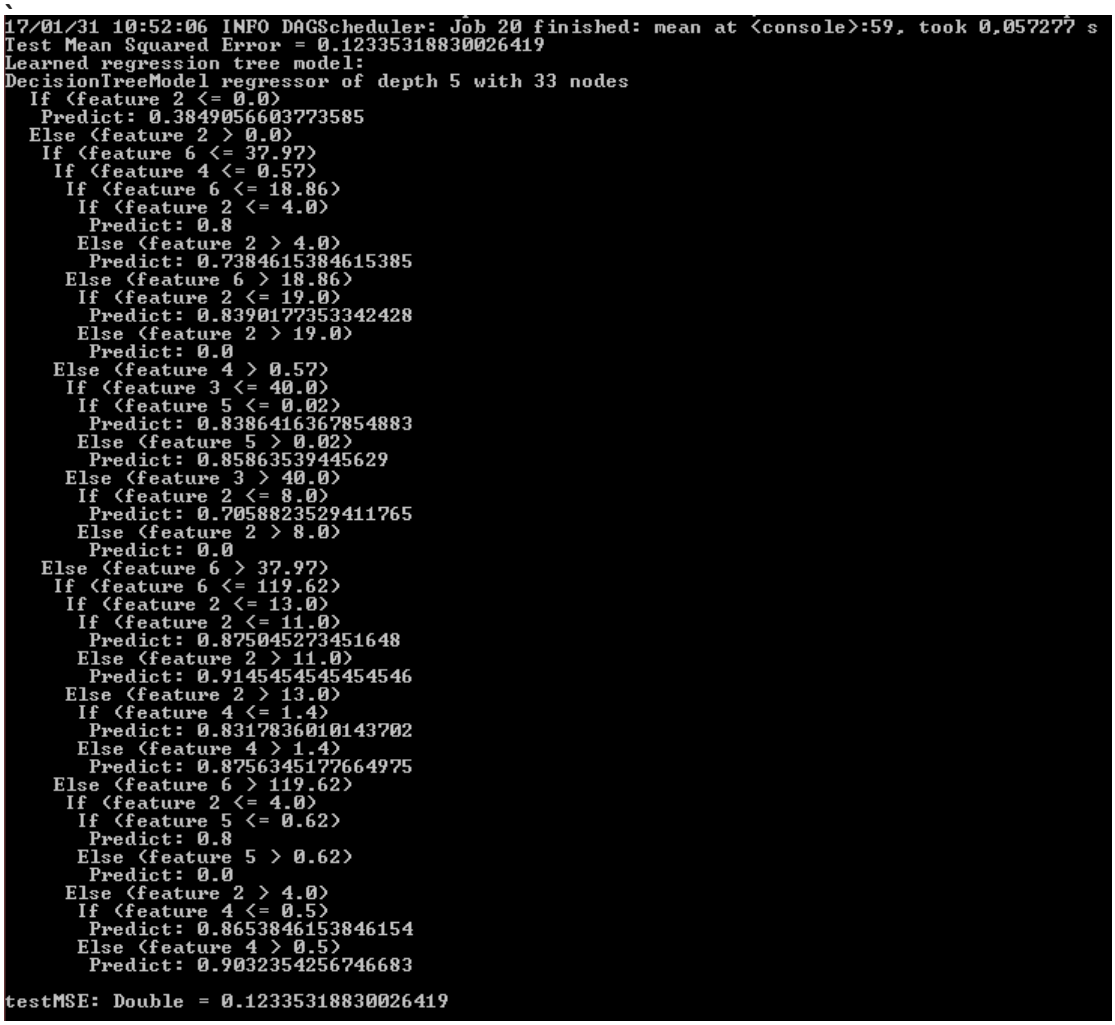
Εικόνα 32: Μοντέλο παλινδρόμησης

```
// Αξιολόγηση του μοντέλου που δημιουργήθηκε
val valuesAndPreds = parsedData.map { point =>
val prediction = model.predict(point.features)
(point.label, prediction)
}
```

```
17/01/31 10:49:17 INFO DAGScheduler: Job 19 finished: mean at <console>:47, took 0,135466 s
training Mean Squared Error = 0.12335318830026419
MSE: Double = 0.12335318830026419
```

Εικόνα 33: Αξιολόγηση μοντέλου παλινδρόμησης

```
// Εμφάνιση του μοντέλου
val MSE = valuesAndPreds.map{ case(v, p) => math.pow((v - p), 2)}.mean()
println("training Mean Squared Error = " + MSE)
```



Εικόνα 34: Εμφάνιση μοντέλου παλινδρόμησης

Το πρόγραμμα ανάλυσης Naïve Bayes

Ο NaiveBayes υλοποιεί την συνάρτηση multinomial naive Bayes. Παίρνει ένα RDD των LabeledPoint και μία προαιρετική λάμδα (λ) παράμετρο εξομάλυνσης ως είσοδο, και έξοδο ένα μοντέλο NaiveBayesModel, το οποίο μπορεί να χρησιμοποιηθεί για την αξιολόγηση και την πρόβλεψη.

Για τα δεδομένα που έχουμε, χρησιμοποιούμε την παρακάτω διαδικασία:

```
// Φορτώνουμε τις απαιτούμενες βιβλιοθήκες
import org.apache.spark.mllib.classification.NaiveBayes
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.regression.LabeledPoint
import org.apache.spark.mllib.evaluation.MulticlassMetrics
```

```
// Φόρτωση του αρχείου
val data = sc.textFile("C:/Users/adamopoulos/Desktop/thesis/dataClas.txt")
val parsedData = data.map { line =>
val parts = line.split(' ')
LabeledPoint(parts(0).toDouble, Vectors.dense(parts(1).split(' ').map(_.toDouble)))
}
// Διαχωρισμός των δεδομένων για εκπαίδευση (60%) κα για δοκιμή (40%)
val splits = parsedData.randomSplit(Array(0.6, 0.4), seed = 11L)
val training = splits(0)
val test = splits(1)

// Τρέχουμε το μοντέλο Naïve Bayes
val model = NaiveBayes.train(training, lambda = 1.0)

// Αξιολόγηση του μοντέλου που δημιουργήθηκε
val predictionAndLabel = test.map(p => (model.predict(p.features), p.label))
val accuracy = 1.0 * predictionAndLabel.filter(x => x._1 == x._2).count() / test.count()
```

```
17/01/31 12:25:54 INFO DAGScheduler: ResultStage 3 (count at <console>:51) finished in 0,062 s
17/01/31 12:25:54 INFO DAGScheduler: Job 2 finished: count at <console>:51, took 0,128949 s
accuracy: Double = 0.8526389742689031
```

Εικόνα 35: Αξιολόγηση του μοντέλου Naïve Bayes

```
val predictionAndLabels = test.map { point =>
val score = model.predict(point.features)
(score, point.label)
}
val metrics = new MulticlassMetrics(predictionAndLabels)
metrics.labels.foreach( l => println(metrics.fMeasure(l)))
```

```
17/01/31 12:36:27 INFO DAGScheduler: ResultStage 15 (countByValue at MulticlassMetrics.scala:44) finished in 0,010 s
17/01/31 12:36:27 INFO DAGScheduler: Job 8 finished: countByValue at MulticlassMetrics.scala:44, took 0,113978 s
0.0
0.9204588547591961
```

Εικόνα 36: Αξιολόγηση του μοντέλου Naïve Bayes

3.5 Αποτίμηση Αποτελεσμάτων

Σε αυτή την παράγραφο θα παρουσιάσουμε τα αποτελέσματα των δύο πειραμάτων που διεξήχθηκαν στις προηγούμενες δύο παραγράφους. Συγκεκριμένα θα παρουσιάσουμε τα τετραγωνικά σφάλματα των πειραμάτων ώστε να καταλήξουμε στις παραμέτρους που μας δίνουν το μικρότερο μέσο τετραγωνικό σφάλμα. Ακόμα, θα αναφερθούμε σε πιθανές προτάσεις που μπορούν να χρησιμοποιηθούν οι παραπάνω μέθοδοι σε συστήματα υποστήριξης αποφάσεων.

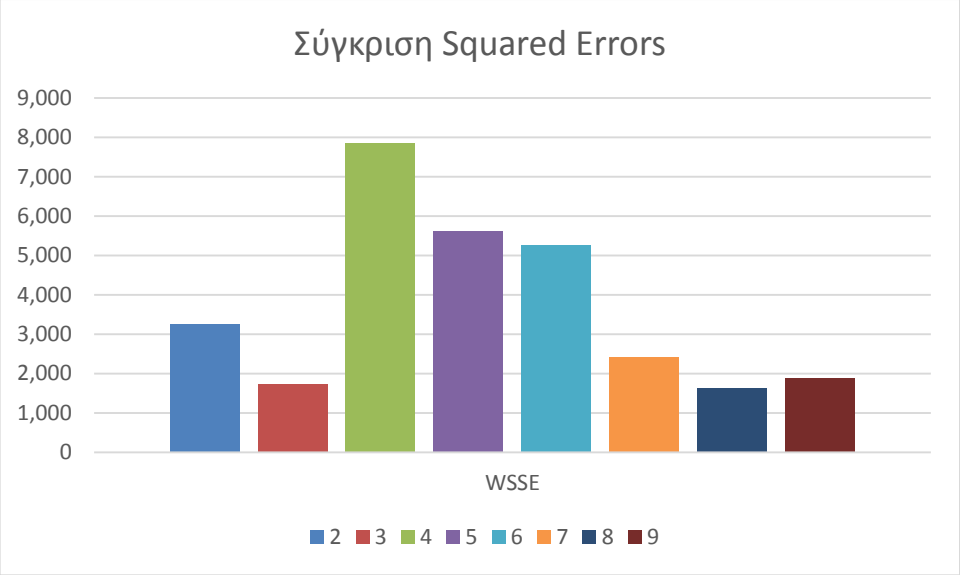
Το μέσο τετραγωνικό σφάλμα είναι το πιο γνωστό και συνάμα ευρέως χρησιμοποιούμενο, στη θεωρία και στις εφαρμογές της Στατιστικής, κριτήριο σύγκρισης και αξιολόγησης εκτιμητών. Επιπροσθέτως, μέσω αυτού του κριτηρίου, παρουσιάζουμε μια ιδιότητα των εκτιμητών, την αμεροληψία. Εκτιμητές που έχουν αυτήν την ιδιότητα αναφέρονται ως αμερόληπτοι εκτιμητές. Ακολούθως δίνουμε αμερόληπτους εκτιμητές σημαντικών παραμέτρων μιας κατανομής, όπως η μέση τιμή και η διασπορά της.

3.5.1 Αποτίμηση αποτελεσμάτων για το πείραμα συσταδοποίησης (clustering)

Στον αλγόριθμο k-means η μόνη παράμετρος που μπορούμε να αλλάξουμε για να μπορέσουμε να πάρουμε το μικρότερο τετραγωνικό λάθος είναι η παράμετρος k, η οποία μας δείχνει τον αριθμό των συστάδων (clusters) που θέλουμε να δημιουργήσει ο αλγόριθμος.

Έτσι, ο αλγόριθμος του πειράματος μας έτρεξε με διαφορετικούς αριθμούς της παραμέτρου k όπως 2, 3, 4, 5, 6, 7, 8, 9.

Στον παρακάτω πίνακα γίνεται σύγκριση των τετραγωνικών λαθών του αλγορίθμου k-means με όλα τα διαφορετικά k. Έτσι συγκρίνοντας τα αποτελέσματα στην εικόνα 33 παρατηρείται ότι με k = 8 και k = 3 εμφανίζονται τα μικρότερα τετραφωνικά σφάλματα.

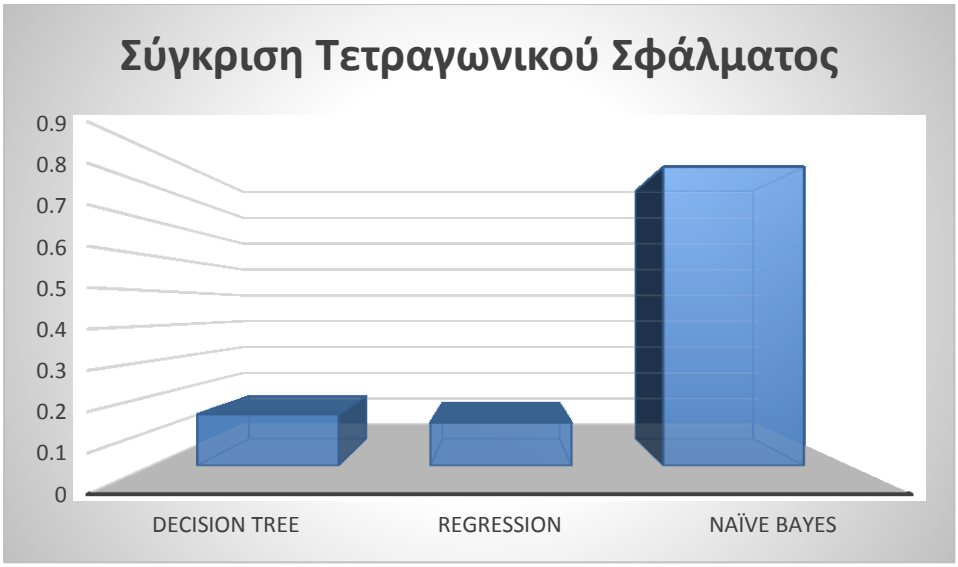


Εικόνα 37: Σύγκριση τετραγωνικών λαθών με διαφορετικό αριθμό συστάδων

Χρησιμοποιώντας, λοιπόν, την μεθοδολογία του πειράματος της συσταδοποίησης θα μπορούσε κάποια εταιρεία μεταφορών, η οποία έχει πολλά καταστήματα σε όλη την Ελλάδα αλλά και στο εξωτερικό, να χωρήσει τα καταστήματα της στις κατάλληλες συστάδες με στόχο την αποδοτικότητα τους. Έτσι λοιπόν, θα μπορούσαν να εντοπιστούν ποια καταστήματα δεν είναι αποδοτικά και στη συνέχεια με περεταίρω ανάλυση να εντοπιστεί η αιτία στην οποία οφείλεται η χαμηλή αποδοτικότητα τους με σκοπό την εξάλειψη της. Επίσης, ένα άλλο κομμάτι που θα μπορούσε να εντοπιστεί είναι το ποια καταστήματα έχουν υψηλή αποδοτικότητα και με τον ίδιο τρόπο να εντοπίσουν το λόγο που είναι πιο αποδοτικά ώστε να μπορέσουν να εφαρμόσουν τις αντίστοιχες μεθόδους και στα λιγότερο αποδοτικά καταστήματα ώστε να αυξηθεί η συνολική αποδοτικότητα της εταιρείας και να αυξηθεί το κέρδος της.

3.5.2 Αποτίμηση αποτελεσμάτων για το πείραμα της πρόβλεψης του κέρδους των πελατών της

Στο δεύτερο πείραμα, όπως προαναφέρθηκε, χρησιμοποιήθηκαν τρεις αλγόριθμοι κατηγοριοποίησης, ο αλγόριθμος Decision Tree, ο αλγόριθμος Regression και ο αλγόριθμος Naïve Bayes. Στην εικόνα 34 παρατηρείται ότι ο αλγόριθμος Naïve Bayes παρουσιάζει το μεγαλύτερο τετραγωνικό σφάλμα σε σύγκριση με τους άλλους δύο αλγορίθμους. Το χαμηλότερο τετραγωνικό σφάλμα παρουσιάζεται με τον αλγόριθμο Regression όπου δείχνει να είναι και ο πιο αποδοτικός αλγόριθμος. Ο Naïve Bayes, για τα δεδομένα που χρησιμοποιήσαμε, φαίνεται να μην είναι καθόλου αποτελεσματικός, αφού παρουσιάζει τεράστιο τετραγωνικό σφάλμα.



Εικόνα 38: Σύγκριση τετραγωνικών λαθών

Με βάση τη μεθοδολογία που χρησιμοποιήσαμε, μια εταιρεία μεταφορών θα μπορούσε να δημιουργήσει ένα μοντέλο για τους πελάτες της με βάση τη δυναμικότητα – σημαντικότητα τους. Έτσι κάνοντας πρόβλεψη του κέρδους των πελατών της θα μπορούσε να τους κατατάξει σε δύο κατηγορίες:

- ✓ **Μη Αποδοτικοί Πελάτες :** Είναι οι πελάτες οι οποίοι επιφέρουν το μικρότερο κέρδος στην εταιρεία, Θα μπορούσαμε λοιπόν με μία περεταίρω ανάλυση να δούμε τα προβλήματα που μπορεί να υπάρχουν σχετικά με αυτούς τους πελάτες ώστε να μπορέσει η εταιρεία να βελτιστοποιήσει τις υπηρεσίες της ώστε να είναι ικανοποιημένοι οι πελάτες της και να προσδώσουν μεγαλύτερο κέρδος.

- ✓ **Αποδοτικοί Πελάτες :** Είναι οι πέλάτες που επιφέρουν το μεγαλύτερο κέρδος στην εταιρεία αφού χρησιμοποιούν τις υπηρεσίες της σε μόνιμη βάση. Είναι οι πέλατες υψίστης σημασίας οπότε θα πρέπει η εταιρεία να στοχεύσει στην πλήρη ικανοποίηση τους μέσω της συνεχούς βελτίωσης των υπηρεσιών που τους προσφέρει και ικανοποιητικών κινήτρων (π.χ. σημαντικών εκπτώσεων).

Κεφάλαιο 4 – Συμπεράσματα

Χρησιμοποιώντας αλγόριθμους μηχανικής μάθησης από την βιβλιοθήκη MLlib του εργαλείου διαχείρησης δεδομένων Apache Spark προσπαθήσαμε να λύσουμε δύο προβλήματα της εταιρείας μεταφορών (η συσταδοποίηση των καταστημάτων της με βάση την αποδοτικότητα τους και η πρόβλεψη του κέρδους των πελατών της). Κάθε πρόβλημα - πείραμα περιείχε διαφορετικά δεδομένα από τις βάσεις της εταιρείας. Επιλέχθηκε ένας αρκετά μεγάλος όγκος δεδομένων ώστε να μπορέσουμε να βγάλουμε όσο το δυνατόν καλύτερα – ακριβέστερα συμπεράσματα. Έγιναν πολλές αλλάγες στις παραμέτρους των αλγορίθμων ώστε να πετύχουμε το επιθυμητό αποτέλεσμα, δηλαδή με τα μικρότερα τετραγωνικά σφάλματα.

Μετά από όλη αυτή τη διαδικασία ανάλυσης μπορούμε να συμπεράνουμε ότι η βιβλιοθήκη MLlib του εργαλείου διαχείρησης δεδομένων Apache Spark περιέχει ένα αρκετά μεγάλο πλήθος αλγορίθμων μηχανικής μάθησης, πράγμα που το καθιστά ένα αξιόπιστο εργαλείο για ανάλυση διαφορετικών τύπων δεδομένων.

Επιπλέον, παρατηρήσαμε μέσω των πειραμάτων μας ότι το εργαλείο αυτό προσφέρει μεγάλες ταχύτητες διαχείρισης δεδομένων. Αυτό το χαρακτηριστικό του δίνει τη δυνατότητα να προσφέρει γρήγορες αναλύσεις σε μεγάλο όγκο δεδομένων. Η ταχύτητα είναι ένα σημαντικό κομμάτι μιας ανάλυσης και πολλοί το έχουν ως ένα απαραίτητο χαρακτηριστικό στην επιλογή του που θα χρησιμοποιήσουν στην ανάλυση τους.

Ένα άλλο χρήσιμο χαρακτηριστικό του εργαλείου που χρησιμοποιήσαμε στην συγκεκριμένη διπλωματική εργασία είναι ότι μπορείς να χρησιμοποιήσεις sql κώδικα για να μπορέσεις να επεξεργαστείς πιο γρήγορά και εύκολα – εφόσον πολλοί είναι εξοικιωμένοι με αυτή τη γλώσσα βάσεων δεδομένων – τα δεδομένα σου, πρωτού ξεκινήσεις την ανάλυση σου ή και κατά τη διάρκεια της.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Ibmbigdatahub.com, (2015). The Big Data Hub — Understanding big data for the enterprise. [online] Available at: http://www.ibmbigdatahub.com/

[2] Hadoop: A brief history. Doug Cutting,Yahoo! [online] Available at: research.yahoo.com/files/cutting.pdf

[3] Hadoop.apache.org. HDFS Architecture Guide. [online] Available at: https://hadoop.apache.org/docs/r1.2.1/hdfs design.html

[4] MapReduce: Simplified Data Processing on Large Clusters. Jeffrey Dean and Sanjay Ghemawat. http://static.googleusercontent.com/media/research.google.com/es/us/archive/mapreduce osdi04.pdf

[5] Wikipedia. Machine learning. [online] Available at: https://en.wikipedia.org/?title=Machine learning

[6] Learning Spark Lighting-fast data analysis. Holden Karau, Andy Konwinski, Patrick Wendell & Matei Zaharia.

[7] Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., J. Franklin, M., Shenker, S. and Stoica, I. (2012). Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. [online] Available at: https://www.cs.berkeley.edu/ ˜matei/papers/2012/nsdi spark.pdf

[8] Pentreath, N. and Paunikar, A. (n.d.). Machine learning with Spark.

[9] Spark.apache.org, . Spark Streaming - Spark 1.4.1 Documentation. [online] Available at: https://spark.apache.org/docs/1.4.1/streaming-programming-guide.html

[10] Flume.apache.org, . Welcome to Apache Flume — Apache Flume. [online] Available at: https://flume.apache.org/

[11] Shreedharan, H. . Using Flume. Sebastopol, CA: O'Reilly Media.

[12] Spark.apache.org, . Spark Streaming + Flume Integration Guide - Spark 1.3.0 Documentation. [online] Available at: https://spark.apache.org/docs/1.3.0/streaming-flume-integration.html

[13] Garg, N. (2015). Learning Apache Kafka. Birmingham: Packt Publishing. 55 BIBLIOGRAPHY 56

[14] Team, A. (2015). Apache HBase TM Reference Guide. [online] Hbase.apache.org. Available at: http://hbase.apache.org/book.html#datamodel

[15] https://spark.apache.org/docs/1.4.1/mllib-clustering.html

[16] Es.slideshare.net, (2015). HBaseCon 2012 — HBase Schema Design - Ian Varley, Salesforce. [online] Available at: http://es.slideshare.net/cloudera/5-h-base-schemahbasecon2012?qid=6884da10- f052-45a6-862f-914c298e983c&v=default&b=&from search=3

[17] Wikipedia, (2015). Scala (programming language). [online] Available at: https://en.wikipedia.org/wiki/Scala (programming language)

[18] Odersky, M. (2015). A Brief History of Scala. [online] Artima.com. Available at: http://www.artima.com/weblogs/viewpost.jsp?thread=163733

[19] Wampler, D. and Payne, A. (2009). Programming Scala. Sebastopol, CA: O'Reilly.

[20] Wikipedia, (2015). Cluster manager. [online] Available at: https://en.wikipedia.org/wiki/Cluster manager

[21] Ryza, S. (2015). Apache Spark Resource Management and YARN App Models. [online] Cloudera Developer Blog. Available at: http://blog.cloudera.com/blog/

[22] Yoav Freund, Robert E. Schapire (1999). [online] Available at: http://cseweb.ucsd.edu/

[23] L. Breiman., (1996). [online] Available at: https://www.stat.berkeley.edu/ breiman/OOBestimation.pdf [Accessed 18 Oct. 2015].

[24] L. Breiman., (2001). Random forests. Machine learning

[25] https://en.wikipedia.org/wiki/Naive_Bayes_classifier

[26] https://en.wikipedia.org/wiki/Classification

[27] https://en.wikipedia.org/wiki/K-means_clustering