ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Πτυχιακή Εργασία

Τεχνολογίες και τεχνικές ανάλυσης των μεγάλων δεδομένων σε θέματα υγείας

ΛΑΖΑΡΗ ΜΑΡΙΑ-ΕΛΕΝΗ

Επιβλέπων καθηγητής: Δρ. Ταμπακάς Βασίλειος

ΠΑΤΡΑ 2023

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή

Πάτρα, /10/2023

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Ονοματεπώνυμο, Υπογραφή

2. Ονοματεπώνυμο, Υπογραφή

3. Ονοματεπώνυμο, Υπογραφή

Υπεύθυνη Δήλωση Φοιτητή

Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τη συγκεκριμένη εργασία.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Πελοποννήσου δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα εκ μέρους του Τμήματος.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή κ. Ταμπακά Βασίλειο για την καθοδήγηση του στην εκπόνηση της παρούσας πτυχιακής εργασίας.

Περίληψη

Η παρούσα πτυχιακή αποτελεί την συνέχεια της πτυχιακής εργασίας που είχα υλοποιήσει για το ακαδημαϊκό έτος 2021-2022 με θέμα: Η διαχείριση μεγάλων δεδομένων με τις NoSQL ΒΔ και το Spark.

Η επιστήμη της ιατρικής και η επιστήμη των υπολογιστών αναπτύσσονται παράλληλα και αρκετές φορές οι δρόμοι ανάπτυξης διασταυρώνονται. Ο χώρος της υγείας αποτελεί πρόκληση για την επιστήμη των δεδομένων, καθώς παράγει με εκθετικά αυξανόμενο ρυθμό δεδομένα ποικίλης μορφής. Τα υγειονομικά δεδομένα απαιτούν ταχεία επεξεργασία, ακριβή ανάλυση και ασφαλή αποθήκευση από τα υπολογιστικά συστήματα. Η επιστήμη των Big Data έχει αναπτύξει συστήματα τα οποία μπορούν να ανταποκριθούν σε αυτές τις απαιτήσεις. Η παρούσα πτυχιακή εργασία έχει στόχο την διερεύνηση των δυνατοτήτων και των χαρακτηριστικών της NoSQL, Apache Cassandra και του Spark για τα Big Data, καθώς και χρήση αυτών των εργαλείων για ανάπτυξη εφαρμογής στα Big Data Healthcare analytics.

Πίνακας περιεχόμενων

Κεφάλαιο 1: Ψηφιακή εποχή

1.1 Εισαγωγή

Η επιστήμη των υπολογιστών έχει μικρή ιστορία συγκριτικά με τους υπόλοιπους επιστημονικούς κλάδους, ενώ ταυτόχρονα τους βοηθά σημαντικά μέσω των εργαλείων που έχουν αναπτυχθεί. Παρατηρώντας την ιστορία της ανθρωπότητας, αντιλαμβανόμαστε ότι ζούμε μέσα στην ψηφιακή επανάσταση. Αυτή η επανάσταση, όπως και η Βιομηχανική, επιφέρει αλλαγή στον τρόπο επικοινωνίας, εκπαίδευσης, εργασίας και γενικότερα σε όλο τον τρόπο ζωής. Καθημερινά δημιουργούνται και συλλέγονται διαφόρων ειδών δεδομένα (κείμενο, εικόνα, ήχος κτλ.), τα οποία είναι χρήσιμα για την αέναη ανάπτυξη της τεχνολογίας.

1.2 Σκοπός πτυχιακής

Σκοπός της πτυχιακής εργασίας είναι η διαχείριση μεγάλων δεδομένων (Big Data) με την αξιοποίηση όλων των δυνατοτήτων και των χαρακτηριστικών NoSQL βάσεων δεδομένων Apache Cassandra. Το εργαλείο που χρησιμοποιείται για την επεξεργασία BD είναι το Apache Spark.

Κεφάλαιο 2: Big Data

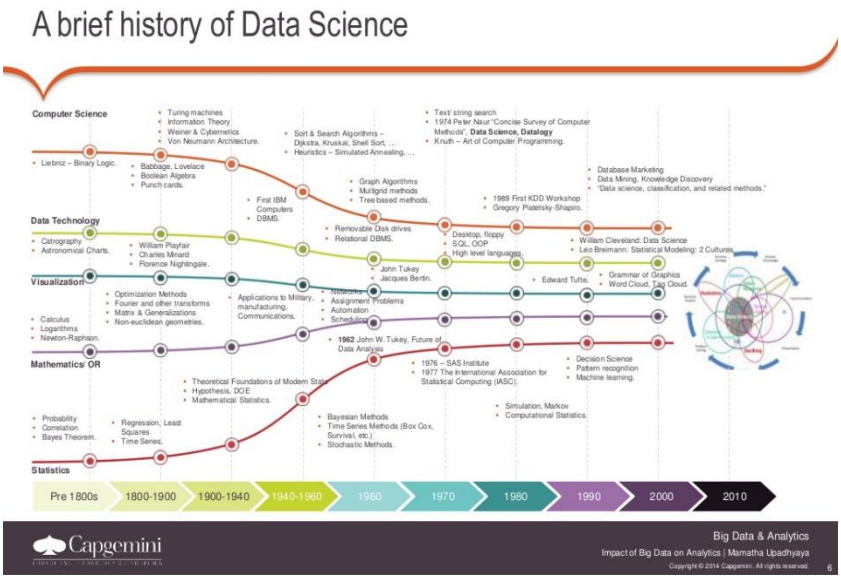
2.1: Τι είναι Big Data;

Η έννοια των Μεγάλων Δεδομένων εμφανίστηκε στα τέλη του 20ού αιώνα, αλλά στηρίχθηκε σε θεμέλια που τέθηκαν πριν την εμφάνιση των πρώτων υπολογιστών. Από την αρχαιότητα μέχρι και σήμερα υπάρχει ανάγκη για αποθήκευση και αξιοποίηση πληροφοριών. Τα τελευταία 20 χρόνια παρατηρούμε ραγδαία αύξηση στον όγκο των πληροφοριών που διαχειριζόμαστε, εξαιτίας της ραγδαίας εξέλιξης της τεχνολογίας και της εισβολής της στην ζωή μας.

Ιστορική αναδρομή

Από την αρχαιότητα μέχρι την εμφάνιση του ENIAC (1945)

Η ιστορία της ανάγκης των ανθρώπου να έχει αποθηκευμένα δεδομένα αρχίζει από την αρχαιότητα. Συγκεκριμένα, τον 18ο αιώνα π.Χ. βασικό μέλημα των ανθρώπων ήταν να καταγράφουν το απόθεμα τους και να ελέγχουν την εμπορική δραστηριότητά τους, ώστε να αυξάνουν το κέρδος τους μειώνοντας τα έξοδα. Το 2400 π.Χ. εφευρίσκεται το αριθμητήριο, το οποίο αποτελεί για αρκετούς επιστήμονες την αρχή των υπολογιστών. Αρκετά χρόνια αργότερα, το 300 π.Χ., χτίζεται το πρώτο αποθετήριο δεδομένων, η βιβλιοθήκη της Αλεξάνδρειας. Το πρώτο αρχέγονο υπολογιστικό σύστημα εμφανίζεται το 100 π.Χ., είναι ο μηχανισμός των Αντικυθήρων ένας αρχαίος αναλογικός υπολογιστής και όργανο για την παρατήρηση αστερισμών. Στη νεότερη ιστορία της Ευρώπης, καταγράφεται η προσπάθεια για στατιστική ανάλυση, με σκοπό να περιοριστεί η εξάπλωση της επιδημίας (πανώλη) από τον John Graunt 1663. Στις ΗΠΑ το 1881, ο Herman Hollerith θέλοντας να βοηθήσει την εργασία των απογραφέων δημιούργησε ένα μηχανικό πίνακα με τη χρήση διάτρητων καρτών, ώστε να ταξινομούνται ευκολότερα τα αποτελέσματα των μετρήσεων. Το 1926, ο εφευρέτης του εναλλασσόμενου ηλεκτρισμού Nikola Tesla σε συνέντευξη στο περιοδικό Colliers προέβλεψε ότι οι άνθρωποι στο μέλλον θα έχουν τη δυνατότητα πρόσβασης και ανάλυσης δεδομένων με τη χρήση μικρής συσκευής. Αυτή η μικρή συσκευή που αναφέρει μπορεί να παρομοιαστεί με τα σύγχρονα smart phones, διότι έχουν μικρό μέγεθος, αλλά μεγάλες υπολογιστικές δυνατότητες.



Εικόνα 1: Η ιστορία της επιστήμης των δεδομένων

Από τον ENIAC μέχρι σήμερα

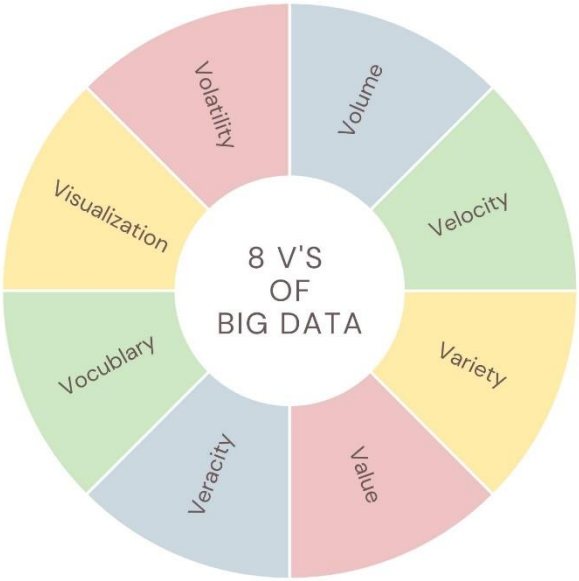
Από το 1945 και μετά η ιστορία των δεδομένων, αλλά και όλου του κλάδου της τεχνολογίας αλλάζει για πάντα με την δημιουργία του πρώτου σύγχρονου υπολογιστή ENIAC. Από το 1958-1968, η IBM αναπτύσσει τις έννοιες της επιχειρηματικής ευφυΐας και της ψηφιακού αποθετηρίου με μαγνητική ταινία, ώστε να αποθηκεύονται εκατομμύρια φορολογικές δηλώσεις και δακτυλικά αποτυπώματα των Αμερικάνων πολιτών. Οι επιστήμονες της IBM συνεχίζουν να πρωτοπορούν στην επιστήμη των δεδομένων, καθώς το 1976 ο Edgar F Codd προτείνει το σχεσιακό μοντέλο δεδομένων, με το οποίο ιεραρχείται ο τρόπος αποθήκευσης δεδομένων. Το μοντέλου το Codd αποτελεί ορόσημο στην ιστορία, διότι ξεκινάει η ευρέως χρήση των δεδομένων έξω από τους επιστημονικούς χώρους. Ο συγγραφέας Eric Larson ανέφερε τον όρο Big Data σε συνέντευξη του στο περιοδικό Harper. Ένα χρόνο αργότερα το 1990, ο John R.Mashley ορίζει την έννοια Μεγάλα Δεδομένα ονομάζεται το σύνολο των δεδομένων που δεν είναι αντιληπτά και διαχειρίσιμα από την κλασσική πληροφορική και τα εργαλεία του λογισμικού. Κατά την εικοσαετία 1990-2010, υπάρχει ραγδαία εξάπλωση στη χρήση πληροφοριακών συστημάτων, το οποίο σήμαινε και απότομη αύξηση των πληροφοριών που αποθηκεύονται. Ορόσημο της εικοσαετίας για τα Big Data είναι το 2009, με το ίδρυμα McKinsey να αναφέρει ότι η ανθρωπότητα βρίσκεται στις αρχές της 4ης βιομηχανικής επανάστασης. Η επανάσταση της ψηφιακής εποχής με αναπόσπαστο μέρος της τα Big Data, επιφέρει αλλαγές σε όλους τους τομείς της ζωής.

Δεν μπορεί να αποτυπωθεί με ακρίβεια ο ορισμός των Big Data, διότι υπάρχουν παντού,εξελίσσονται συνεχώς και είναι αυτά που ορίζουν το μέλλον, αξιοποιώντας τη γνώση του παρελθόντος. Ο πιο σύγχρονος ορισμός, αναφέρεται από την Wikipedia: <<Τα μεγάλα δεδομένα είναι το επιστημονικό πεδίο, όπου γίνεται επεξεργασία των τρόπων ανάλυσης και συστηματικής εξόρυξης πληροφοριών από σύνολα δεδομένων που είναι πολύπλοκα διασυνδεδεμένα και διαχειρίζονται δύσκολα με την χρήση του παραδοσιακού λογισμικού >>.

2.2: Χαρακτηριστικά Big Data

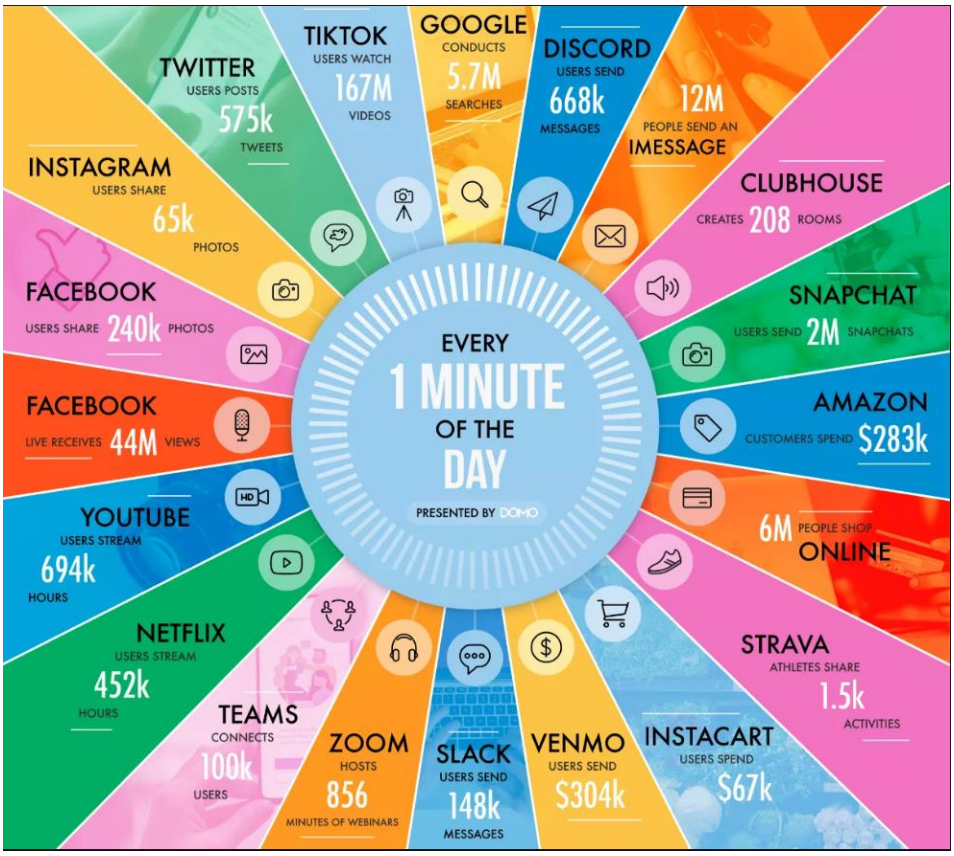
2.2.1: Τα 8V των BD

Στη διεθνή βιβλιογραφία, όλοι οι ορισμοί των Big Data συνοψίζονται σε τρεις βασικούς άξονες: όγκος(Volume), ταχύτητα (Velocity), ποικιλία (Variety). Ωστόσο, η ραγδαία ανάπτυξη της τεχνολογίας επέφερε την επέκταση των χαρακτηριστικών με στόχο την λεπτομερέστερη ακρίβεια. Ανάλογα με την χρήση και τον βαθμό χρήσης των Big Data, η κάθε εταιρία και ερευνητής θέτει το πλήθος των χαρακτηριστικών που τον εξυπηρετούν.



Πίνακας 1: 8 V's of Big Data

<u>Όγκος (Volume):</u> Τα τελευταία χρόνια παρατηρείται εκθετική αύξηση του πλήθους των δεδομένων που καλούμαστε να αποθηκεύσουμε. Στις αρχές του 21 αιώνα είχαν δημιουργηθεί 15.6 Exabytes (~17 εκατομμύρια GB). Είκοσι χρόνια αργότερα αυτός ο αριθμός εκατονταπλασιάστηκε. Το 2021 υπολογίστηκε ότι ο μέσος άνθρωπος παρήγαγε 50MB κάθε δευτερόλεπτο. Παρατηρώντας την παρακάτω εικόνα μπορεί να γίνει κατανοητό το μέγεθος των πληροφοριών που αποθηκεύονται και επεξεργάζονται κάθε λεπτό. Οι ερευνητές προβλέπουν ότι, το 2035 ο όγκος των δεδομένων θα είναι 2.142 Zetabytes. Γίνεται κατανοητό ότι υπάρχει επιτακτική ανάγκη για ανάπτυξη νέων εφαρμογών και επαναπροσδιορισμό των υπαρχόντων αρχιτεκτονικών, ώστε τα δεδομένα να εξάγονται (***Extract***) , να επεξεργάζονται (***Transform***) και να φορτώνονται (***Load***) σε ευρείας χρήσης εφαρμογές *ταχύτερα.*



Εικόνα 2: Τα δεδομένα που παράγονται κάθε λεπτό το 2021

Ταχύτητα (Velocity):Ο τεράστιος όγκος δεδομένων προήλθε από την αύξηση της ταχύτητας κατά την οποία παράγονται, διανέμονται και συλλέγονται τα δεδομένα. Τα σύγχρονα συστήματα χρησιμοποιούν ειδικές τεχνικές επεξεργασίας BD, ώστε να μπορέσουν να ανταποκριθούν στις ανάγκες του συστήματος χωρίς να χαθούν τα πολύτιμα δεδομένα. Όσο υψηλότερος είναι ο ρυθμός ταχύτητας, τόσο πιο γρήγορα τα δεδομένα επεξεργάζονται αυξάνοντας την αξίας τους. Οι δύο τύποι ταχύτητας που σχετίζονται με τα μεγάλα δεδομένα είναι η συχνότητα δημιουργίας και η συχνότητα χειρισμού, καταγραφής και αναφοράς [4] . Για τον έλεγχο της ταχύτητας ροής των δεδομένων έχουν αναπτυχθεί εφαρμογές που μπορούν να διαχειρίζονται μεγάλα πακέτα χρήσιμης πληροφορίας σε πραγματικό χρόνο, παραδείγματος χάριν Apache Spark .Παρόλα αυτά απαιτείται χρόνος και προσπάθεια για την δημιουργία αγωγών δεδομένων(data pipelines) για ομαλή διακίνηση των δεδομένων μέσα σε ένα σύστημα.

Ποικιλία (Variety):Τα μεγάλα δεδομένα αντλούν δεδομένα από διαφορετικές πηγές (αισθητήρες, Internet), ώστε να συνθέσουν τις χρήσιμες πληροφορίες που χρειάζεται ένα σύστημα. Η ραγδαία αύξηση της χρήσης αισθητήρων και του IoT, δημιούργησε νέες μορφές δεδομένων όπως δομημένα κείμενα, ήχος, εικόνα, βίντεο, email, καταγραφή διαδικτυακής κίνησης για website, διαγράμματα καρδιακών παλμών κ.ά.. Αυτοί οι τύποι δεδομένων είναι πολύ διαφορετικοί σχέση

με τις πρώτες DB όπου αποτελούνταν αποκλειστικά από πλειάδες και γραμμές. Αυτό σημαίνει ότι τις περισσότερες φορές είναι αδόμητα. Ένας τρόπος διαχείρισης της ποικιλομορφίας των δεδομένων είναι η δημιουργία οροσήμων σε κάθε στάδιο της επεξεργασίας τους. Τα ακατέργαστα δεδομένα αποθηκεύονται προσωρινά σε data lake και στην συνέχεια γίνεται μετατροπή τους σε συγκεκριμένους τύπους, ώστε να φορτωθούν σε RDBMS(σχεσιακή βάση δεδομένων).

Εγκυρότητα (Veracity):Τα δεδομένα συχνά θεωρούνται επικαιροποιημένα και αξιόπιστα. Ωστόσο, στα πραγματικά σύνολα δεδομένων συχνά εμφανίζονται δεδομένα με ανακρίβεια και αβεβαιότητα ως προς την εγκυρότητά τους. Η εγκυρότητα των δεδομένων δεν έγκειται μόνο στην αληθοφάνεια, αλλά και στην ποιότητα των δεδομένων που πρόκειται να μετασχηματιστούν και να χρησιμοποιηθούν για την λήψη κρίσιμων αποφάσεων σε ένα σύστημα. Η ποιότητα των δεδομένων εξαρτάται από ορισμένους παράγοντες όπως: πηγή εξόρυξης, τεχνική συλλογής και τρόπος ανάλυσης κ.ά.. Δεδομένα χαμηλής εγκυρότητας, συνήθως περιέχουν υψηλό ποσοστό «θορύβων» και ανούσιων δεδομένων, τα οποία είναι ανώφελα προς διαχείριση. Από την άλλη πλευρά, τα δεδομένα υψηλής ακρίβειας περιέχουν πολλές εγγραφές, που είναι πολύτιμες για επεξεργασία, συμβάλλοντας σημαντικά στην επίτευξη των στόχων ενός συστήματος. Η βελτίωση της ποιότητας των δεδομένων, μπορεί να επιτευχθεί με την προσθήκη φίλτρων κατά την εξόρυξη τους. Η χρήση φίλτρων χρήζει ιδιαίτερη προσοχή, διότι μπορεί να αφαιρεθεί μεγάλο πλήθος «θορύβων», διπλότυπων εγγραφών, αλλά να καταστήσουν ένα σύστημα ασταθές.

Μεταβλητότητα (Volatility): Στην πάροδο του χρόνου το περιεχόμενο των αποθηκευμένων δεδομένων μεταβάλλεται εξαιτίας διαφόρων παραγόντων. Οι ερευνητές δίνουν έμφαση στον ρυθμό μεταβολής αυτών των τιμών, διότι οι παλαιότερες πληροφορίες ίσως να είναι λανθασμένες, ασαφείς και μη χρήσιμες. Το σημείο στο οποίο υπάρχει διχασμός απόψεων στην επιστήμη των δεδομένων αφορά τη διάρκεια ζωής των δεδομένων. Η διατήρηση και αξιοποίηση παλιωμένων δεδομένων ενέχει τους προαναφερθέντες κινδύνους, όμως η εξέλιξη ενός συστήματος βασίζεται σε παλαιότερες πληροφορίες. Το 2021 ο Sandvik όρισε το μοντέλο μεταβλητότητας. Το μοντέλο εξετάζει τις διαφορές μεταξύ του τρόπου διαχείρισης δεδομένων από τις συσκευές αποθήκευσης και την μέθοδο για την εξόρυξη τους.

Αξία (Value):Πριν ξεκινήσει οποιαδήποτε διαδικασία μετασχηματισμού των δεδομένων είναι σημαντικό να εξεταστεί η αξία τους. Αναγκαία είναι η χρήση αποδοτικών τεχνικών ανάλυσης ώστε να εξαχθούν πρωτοπόρες γνώσεις για την ορθή λήψη αποφάσεων. Ο υπολογισμός της αξίας των δεδομένων είναι απαραίτητος για την ευημερία ενός συστήματος, διότι προκύπτει με την χρήση στατιστικών μοντέλων εμφανίζοντας όλους τους παράγοντες από τους οποίους επηρεάζεται αυτή η επιτυχία.

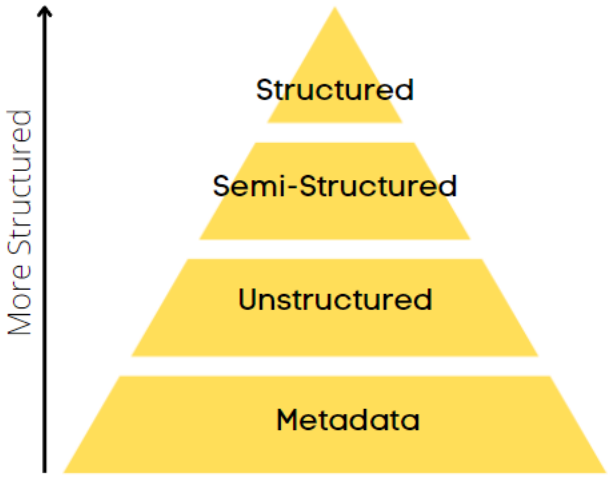
Οπτικοποίηση (Visualization):Τα μεγάλα δεδομένα περιέχουν εκατομμύριες ή και δισεκατομμύριες ακατέργαστες πληροφορίες, οι οποίες με την χρήση εργαλείων οπτικοποίησης

κατανοούνται από τους χρήστες ενός συστήματος. Η οπτικοποίηση περιγράφει σχεδόν όλους του τύπους δεδομένων (αριθμοί, τριγωνομετρικές συναρτήσεις, στατιστικοί αλγόριθμοι) σε **οπτική μορφή** (εκθέσεις αναλυτικών στοιχείων, διαγράμματα). Τα γραφήματα αυτά διαφέρουν από τα τυπικά μαθηματικά διαγράμματα, καθώς περιέχουν συνθετότερες αναπαραστάσεις, όπως heatmaps, wordclouds κ.ά., αναδεικνύοντας απροσδόκητες συσχετίσεις και μοτίβα μεταξύ των δεδομένων, ώστε να ληφθούν αποτελεσματικότερες αποφάσεις προς όφελος ενός συστήματος.

Λεξιλόγιο (Vocabulary):Η επεξεργασία και η ανάλυση τεράστιων όγκων δεδομένων κάποιες φορές δημιουργεί αοριστίες συσχετίσεων, διότι δεν συνεπάγεται αιτιώδη και σημασιολογική συνάφεια. Η λέξη «λεξιλόγιο» στον κλάδο της επιστήμης των δεδομένων έχει δύο ερμηνείες. Η πρώτη ερμηνεία αναφέρεται στο ζήτημα της επικοινωνίας μεταξύ του παρόχου και του χρήστη ενός συστήματος καθώς και της γλώσσας που χρησιμοποιείται για να περιγραφεί το επιθυμητό αποτέλεσμα. Η δεύτερη ερμηνεία διακλαδίζεται στην σημασιολογική αναζήτηση και στις λειτουργίες μέσα σε ένα σημασιολογικό χώρο. Καθορίζονται με σαφήνεια οντολογίες που αντιπροσωπεύουν συγκεκριμένους ορισμούς αλλά και ταυτόχρονα είναι αλληλένδετες με άλλους όρους. Για παράδειγμα στον χώρο της τεχνητής νοημοσύνης ο όρος «παιδί» υποχρεωτικά συνεπάγεται ότι έχει «γονέα».

2.2.2: Δομή BD

Στην επιστήμη των υπολογιστών, μια δομή δεδομένων είναι ο ιδιαίτερος τρόπος οργάνωσης και αποθήκευσης δεδομένων σε έναν υπολογιστή ώστε να υπάρχει πρόσβαση και αποτελεσματική διαχείριση τους. Τα δεδομένα παράγονται από την αλληλεπίδραση ανθρώπου-υπολογιστή, την διεπαφή μηχανών χωρίς την ανθρώπινη παρέμβαση. Όλες αυτές οι αλληλεπιδράσεις δημιουργούν τέσσερις τύπους δομών.



Πίνακας 2: Πυραμίδα δομής τωνBig Data

Δομημένα δεδομένα (Structured data):Τα δομημένα δεδομένα αποτελούν την πιο «κλασική» μορφή αποθήκευσης δεδομένων, αφού τα πρώτα συστήματα διαχείρισης βάσεων δεδομένων μπορούσαν να αποθηκεύουν, να επεξεργάζονται και να έχουν πρόσβαση μόνο σε αυτά. Αποτελούνται από πλειάδες και στήλες, και τα δεδομένα εγγράφονται σύμφωνα με τις προκαθορισμένες συσχετίσεις του εκάστοτε συστήματος. Εξαιτίας της αυστηρής εξάρτησης από το μοντέλο/πρότυπο διαχείρισης που ακολουθούν, τα δομημένα δεδομένα είναι εξαιρετικά ισχυρά και μπορούν εύκολα να συλλεχθούν από διαφορετικές (κατανεμημένες ή και μη-κατανεμημένες) βάσεις δεδομένων και να αναλυθούν.

Ημιδομημένα δεδομένα(Semi-Structured data):Τα ημιδεδομένα δεδομένα, όπως αρχεία XML και JSON,διαθέτουν καθορισμένα και συνεπή χαρακτηριστικά, αλλά δεν δομούνται σύμφωνα με προεπιλεγμένες συσχετίσεις όπως τα δομημένα δεδομένα. Το γεγονός ότι δεν εξαρτώνται τόσο έντονα από πρότυπα διαχείρισης, δεν σημαίνει ότι είναι σημασιολογικά ανοργάνωτα και μη προσβάσιμα για επεξεργασία.

Αδόμητα δεδομένα(Unstructured data):Τα αδόμητα δεδομένα, όπως ήχος, εικόνες, email, δορυφορικά σήματα, ούτε διαθέτουν προκαθορισμένο μοντέλο δεδομένων ούτε οργανώνονται με αυστηρές συσχετίσεις. Αυτοί οι δύο παράγοντες προκαλούν παρατυπίες και ασάφειες κάνοντας τα αδόμητα δεδομένα δύσκολα ως προς την διαχείριση και ιδιαίτερα την ανάλυση τους. Η επιτακτική ανάγκη εξόρυξης πολύτιμων πληροφοριών από τα αδόμητα αποτελεί μία από τις βασικότερες αιτίες για την ταχεία ανάπτυξη των μεγάλων δεδομένων, καθώς αναπτύχθηκαν πολλές νέες τεχνολογίες και εργαλεία.

Μεταδεδομένα (Metadata):Τα μεταδεδομένα είναι ένα από τα σημαντικότερα στοιχεία για την ανάλυση των Big Data. Παρέχουν πρόσθετες πληροφορίες για ένα ήδη υπάρχον σύνολο δεδομένων, δηλαδή δεδομένα για τα δεδομένα. Το χαρακτηριστικότερο παράδειγμα χρήσης των μεταδεδομένων είναι οι αναρτημένες φωτογραφίες στα μέσα κοινωνικής δικτύωσης, οι οποίες αποτελούνται από αδόμητα δεδομένα -pixels-, δομημένα δεδομένα – σύντομο επεξηγηματικό κείμενο (λεζάντα)- και ημιδεδομένα δεδομένα-URL-. Τα μεταδεδομένα είναι η ημερομηνία και ώρα δημιουργία της ανάρτησης, το όνομα του δημιουργού. Τα μεταδεδομένα είναι ευμετάβλητα, επιτρέποντας την κατηγοριοποίηση και ανάλυση τους.

2.3: Αρχιτεκτονική των Big Data

Η αρχιτεκτονική των μεγάλων δεδομένων αποτελεί το πρωταρχικό σύστημα που χρησιμοποιείται για τη διαχείριση των BD, χρησιμοποιώντας τα κατάλληλα εργαλεία ανάλυσης δεδομένων ούτως ώστε να εξαχθούν ωφέλιμες πληροφορίες από τα πολύπλοκα δεδομένα. Στην αρχιτεκτονική καθορίζονται με σαφήνεια τα συστατικά στοιχεία, τα επίπεδα και τους αγωγούς ροής δεδομένων (data pipelines) που θα χρησιμοποιηθούν.

Πηγή δεδομένων(Data source):Ο σχεδιασμός της αρχιτεκτονικής εξαρτάται σε μεγάλο βαθμό από τις πηγές δεδομένων. Προτού εφαρμοστεί οποιαδήποτε αρχιτεκτονική μέθοδος, θα πρέπει οι πηγές δεδομένων να προσδιοριστούν και να κατηγοριοποιηθούν. Όπως προαναφέρθηκε τα δεδομένα προέρχονται από πολλές πηγές και σε διαφορετικές μορφές.

Αποθήκευση δεδομένων (Data storage):Η αποθήκευση δεδομένων αποτελεί το βασικότερο στοιχείο ή και επίπεδο της αρχιτεκτονικής των BD. Αυτό είναι το επίπεδο λήψης δεδομένων, που ενοποιεί δεδομένα από διάφορες πηγές, τα αποθηκεύει *ιδανικά* σε ένα κατανεμημένο χώρο αποθήκευσης και μετασχηματίζει τα αδόμητα και τα μεταδεδομένα σε μορφές δεδομένων σύμφωνες με τις απαιτήσεις ενός συστήματος. Τα structured data συνήθως αποθηκεύονται σε σχεσιακή DB, ενώ τα unstructured data μπορούν τοποθετηθούν σε μη-σχεσιακές (*NoSQL)*βάσεις, όπως Apache Cassandra.

Ομαδοποιημένη επεξεργασία (Batch processing):Τα σύνολα των δεδομένων είναι τεράστια, συχνά ως λύση βέλτιστης διαχείρισης απαιτείται ένα σύστημα ομαδοποιημένης επεξεργασίας. Το σύστημα αποτελείται από μακροχρόνιες διαδικασίες, οι οποίες φιλτράρουν, συγκεντρώνουν και επεξεργάζονται δεδομένα παλαιοτέρων αναλυτικών στοιχείων. Κατά την διάρκεια της ομαδοποιημένης επεξεργασίας, τα αρχεία, που δημιουργούνται από την ενοποίηση δεδομένων διαφόρων πηγών, διαβάζονται επεξεργάζονται και τα νέα αποτελέσματα τους εγγράφονται σε νέα αρχεία.

Επεξεργασία σε πραγματικό χρόνο (Real-time processing):Τα περισσότερα συστήματα επεξεργάζονται δεδομένα σε πραγματικό χρόνο και έχουν ανάγκη για την ύπαρξη κάποιου μηχανισμού απορροφώντας και αποθηκεύοντας μηνύματα για ομαλή ροή των δεδομένων. Η απορρόφηση των μηνυμάτων συνήθως πραγματοποιείται με την προσωρινή αποθήκευση μηνυμάτων, η οποία υποστηρίζει την βαθμιαία επεξεργασία, την αξιόπιστη παράδοση και την σημασιολογική αλληλουχία των πληροφοριών. Αφού καταγραφούν τα μηνύματα, τα δεδομένα πρέπει να φιλτραριστούν, να ενοποιηθούν και να προετοιμαστούν για ανάλυση. Αυτό το επίπεδο αρχιτεκτονικής εστιάζει στην κατηγοριοποίηση δεδομένων για ομαλή μετάβαση στα βαθύτερα στρώματα του συστήματος.

Ανάλυση και αναφορά δεδομένων(Data Analysis&reporting):Ο σκοπός των περισσότερων πλατφορμών διαχείρισης μεγάλων δεδομένων είναι η λήψη ορθών αποφάσεων, η οποία προέρχεται από την ανάλυση δεδομένων και τη δημιουργία αναφορών. Το επίπεδο ανάλυσης αλληλοεπιδρά με το επίπεδο αποθήκευσης θέτοντας καίρια ερωτήματα με στόχο την εξόρυξη πολύτιμων γνώσεων. Οι αναφορές αποτελούν την λογική διασύνδεση όλων των αποτελεσμάτων της επεξεργασίας και ανάλυσης δεδομένων, αναδεικνύοντας την βέλτιστη απόφαση. Τα τελευταία χρόνια, η ανάλυση δεδομένων και η δημιουργία αναφορών έχει εδραιωθεί εξαιτίας των εργαλείων που έχουν αναπτυχθεί, όπως Apache Spark.

Αυτοματοποίηση (Automation):Ο τρόπος διαχείρισης μεγάλων δεδομένων αποτελείται από επαναλαμβανόμενες λειτουργίες επεξεργασίας δεδομένων, που ακολουθούνται από μετασχηματισμό δεδομένων, μετακίνηση μεταξύ των πηγών και φόρτωση των επεξεργασμένων δεδομένων σε έναν χώρο αποθήκευσης αναλυτικών δεδομένων. Όλες αυτές οι ενσωματωμένες εργασίες πρέπει να αυτοματοποιηθούν χρησιμοποιώντας κάποιο εργαλείο συντονισμού όπως το Apache Oozzie ώστε να λαμβάνονται συνεχώς πολύτιμες πληροφορίες.

2.4: Εφαρμογές των Big Data;

Τα μεγάλα δεδομένα μπορούν να χρησιμοποιηθούν για την αποκάλυψη κρυφών μοτίβων και τάσεων, τα οποία είναι εξαιρετικά χρήσιμα για όλους τους τομείς. Ιδιαίτερα για τους οργανισμούς παρέχει την ευκαιρία σε εμπειρογνώμονες των επιχειρήσεων να ερωτούν και να κατανοούν τις πολύτιμες πληροφορίες σύμφωνα με τις επιχειρηματικές ανάγκες, ανεξάρτητα από τη δυσκολία και τον όγκο των δεδομένων, παρουσιάζοντας τα δεδομένα με κατανοητό τρόπο. Η ανάλυση αυτών των πληροφοριών μπορούν να εμφανιστούν νέες κατευθύνσεις για την λήψη αποφάσεων που θα αποφέρουν αποδοτική λειτουργία του οργανισμού, μεγάλα οικονομικά κέρδη και περισσότερους ευχαριστημένους πελάτες.

Εικόνα 3: Big Data Εφαρμογές

<u>Κυβέρνηση κράτους</u>: Τα μεγάλα δεδομένα για τις κυβερνήσεις των κρατών προέρχονται από διάφορες πηγές όπως κάμερες κυκλοφορίας/CCTV, δορυφόρους, αισθητήρες, email και τα μέσα κοινωνικής δικτύωσης καθώς και τις μυστικές υπηρεσίες. Το περιεχόμενο των πληροφοριών είναι πολύτιμο για την βέλτιστη διακυβέρνηση και διαχείριση του δημόσιου τομέα. Τα κράτη διαθέτουν διαδικτυακά αποθετήρια open-data, όπου δημοσιεύονται για χρήση από τους πολίτες. Τα μέλη μιας χώρας έχουν την δυνατότητα είτε να διαβάσουν λεπτομερείς εκθέσεις σχετικά με τον τρόπο διακυβέρνησης, είτε να επεξεργαστούν τα ακατέργαστα δεδομένα, οπτικοποιώντας τα στατιστικά με την χρήση διαγραμμάτων και τα αποτελέσματα να δημοσιευτούν, με απώτερο σκοπό την αύξηση της συμμετοχής για την λήψη αποφάσεων τους που τους αφορά. Επιπρόσθετα, παγκοσμίως όλες οι κυβερνήσεις επενδύουν ετησίως τεράστια χρηματικά ποσά σε συστήματα Τεχνητής Νοημοσύνης και υποκείμενες πλατφόρμες cloudκαι δεδομένων για την θωράκιση έναντι σε οποιοδήποτε κίνδυνο. Η επεξεργασία αυτών των δεδομένων αποτελεί υψίστης σημασίας αφού μπορεί να γίνει λεπτομερής πρόβλεψη για ακραία καιρικά φαινόμενα και να γίνει έγκυρη ενημέρωση των πολιτών, ώστε να προστατευτούν κατά την διάρκεια της κακοκαιρίας. Μεγάλο πλήθος εφαρμογών των Big Data υλοποιήθηκε για στρατιωτική χρήση, όπως και το Internet,ώστε να εντοπίσουν ασυνήθιστες ενέργειες στα σύνορα ενός κράτος, ενεργοποιώντας άμεσα τους μηχανισμούς άμυνας. Τέλος, η ανάλυση δεδομένων από τις κάμερες κυκλοφορίας, συμβάλλοντας στην διαχείριση του δημόσιου οδικού δικτύου και την μείωση των παραβάσεων του κώδικα οδικής ασφάλειας. Ταυτόχρονα, η λεπτομερής ανάλυση δεδομένων χρήσης των Μέσων Μαζικής Μεταφοράς, βοηθά στην βέλτιστη διαχείριση των πόρων (καύσιμα ΜΜΜ, πλήθος υπαλλήλων), κατανοώντας τις μέρες και ώρες αιχμής κάθε διαδρομής.

Υγεία: Ο επιστημονικός τομέας της υγείας καθημερινά συλλέγει ένα τεράστιο πλήθος διαφόρων δεδομένων. Με την τεχνολογική πρόοδο πλέον τα δεδομένα δεν δημιουργούνται μόνο στους υγειονομικούς χώρους (νοσοκομεία, μικροβιολογικά εργαστήρια κ.τ.λ.), καθώς στα smart-watches υπάρχουν αισθητήρες, οι οποίοι μπορούν να καταγράψουν θερμοκρασία, καρδιακούς παλμούς, μυϊκούς σπασμούς και τις συνήθειες των χρηστών τους. Η λεπτομερή ανάλυση αυτών των δεδομένων εξυπηρετεί το ιατρικό προσωπικό, αφού μπορεί να κατανοήσει πλήρως την κατάσταση του κάθε ασθενή, εξατομικεύοντας την φαρμακευτική αγωγή. Ταυτόχρονα, τα δεδομένα κάθε ανθρώπου θα μπορούν να συσχετίζονται με τα ιατρικά δεδομένα της οικογενείας του, τα οποία παρέχουν πληροφορίες γονιδιακές παθήσεις που ίσως να εμφανιστούν στον μέλλον. Πέρα από την εξατομικευμένη θεραπεία, δημιουργούνται μοντέλα νοσημάτων, τα οποία θα μπορούν να προβλέπουν σοβαρές ασθένειες, όπως καρκίνος, AIDS, ακόμα και από ασθενείς που δεν εμφανίζουν κανένα σύμπτωμα. Η αναλυτική υγειονομικών δεδομένων απαλλάσσει την ταλαιπωρία των ασθενών από τις δοκιμές θεραπειών για την εύρεση της καταλληλότερη αγωγής, μειώνοντας τον χρόνο ίασης.

Εκπαίδευση: Ο κλάδος της εκπαίδευσης είναι πλημμυρισμένος από τεράστιο όγκο δεδομένων που σχετίζονται με εκπαιδευτικούς, εκπαιδευόμενους, μαθήματα, αποτελέσματα εξετάσεων κ.ά. Η ορθή μελέτη και ανάλυση αυτών των δεδομένων μπορεί να παρέχει πληροφορίες για βελτίωση αποτελεσματικότητας της λειτουργίας των εκπαιδευτικών ιδρυμάτων. Το διοικητικό προσωπικό των εκπαιδευτικών ιδρυμάτων θα έχουν πρόσβαση σε περισσότερες πληροφορίες, λαμβάνοντας αποφάσεις για τις μελλοντικές ανάγκες. Τα εκπαιδευτικά ιδρύματα αποτελούν ένας από τους πιο κοστοβόρους οργανισμούς και με την επεξεργασία των μεγάλων δεδομένων παρέχονται οι πληροφορίες για τα συνολικά έξοδα σε σχέση με τις ανάγκες τους. Τα αποτελέσματα του data analytics ίσως να οδηγήσει είτε σε καλύτερη αξιοποίηση ή την εμφάνιση ανάγκης για αύξηση των εσόδων είτε σε μείωση των σπαταλών του εκπαιδευτικού φορέα. Επιπρόσθετα, τα Big Data επιτρέπουν στο διδακτικό προσωπικό να προσαρμόζει τη δομή των μαθημάτων ανάλογα με τις ανάγκες των εκπαιδευόμενων και να εκσυγχρονίζει το περιεχόμενο της διδακτέας ύλης. Παράλληλα μπορούν να αξιοποιηθούν εργαλεία Μηχανικής Μάθησης, τα οποία στην αυτοματοποιημένη διαχείριση μάθησης. Τέλος, με την κατάλληλη ανάλυση των δεδομένων κάθε εκπαιδευόμενου, μπορεί να βοηθήσει στην κατανόηση της προόδου του και την εύρεση των αδυναμιών, των ενδιαφερόντων του, συμβάλλοντας στην ανάπτυξη στρατηγικών για εξατομικευμένη μάθηση από τους εκπαιδευτικούς.

Ενέργεια: Η ενέργεια είναι ένα πλεονέκτημα που δεν πρόκειται να διαρκέσει για πάντα. Πολλές τεχνολογικές εξελίξεις έχουν προωθήσει τους ανανεώσιμους και επαχρησιμοποιούμενους ενεργειακούς πόρους έναντι του πετρελαίου και του φυσικού αέριου, αλλά εξακολουθεί να υπάρχει η ανάγκη διατήρησης της περιορισμένης διαθέσιμης ενέργειας. Οι περισσότερες χώρες δεν είναι αυτόνομες στην παραγωγή ενέργειας. Η ανάλυση των μεγάλων δεδομένων μπορεί να χρησιμοποιηθεί για την δημιουργία νέων ευρημάτων σχετικά με τον τρόπο εκμετάλλευσης τους από τους δημόσιους οργανισμούς διαχείρισης και να προβλεφθεί η αποδοτικότητα τους σε ακραία καιρικά φαινόμενα. Το έξυπνο δίκτυο των σύγχρονων συστημάτων ηλεκτρικής ενέργειας που

κάνει αμφίδρομες τις ροές ενέργειας, επικοινωνίας και ελέγχου. Το δίκτυο ενεργειακής τροφοδοσίας χρησιμοποιείται για της παραγωγή ηλεκτρικής ενέργειας, διανομής και αξιοποίησης. Τέλος, οι κατασκευαστές δομών παραγωγής και διανομής ενέργειας έχουν πρόσβαση σε ιστορικά δεδομένα σε σχέση με τα ατυχήματα όπως διαρροή φυσικού αερίου σε δημόσιους δρόμους, έκρηξη εργοστασίων πυρηνικής ενέργειας, διαρροή καύσιμων από την εξόρυξη πετρελαίου. Αν αυτές οι πολύτιμες πληροφορίες διαχειριστούν καταλλήλως, θα μπορέσουν να ανοικοδομηθεί όλο ο απαραίτητος εξοπλισμός με σκοπό την ασφάλεια των εργαζομένων και του περιβάλλοντος. Οι τεχνικές ανάλυσης big data αποτελούν σημαντικό στοιχείο για την καλύτερη διαχείριση των διαθέσιμων ενεργειακών πόρων, τα οποία είναι ζωτικής σημασίας.

Γεωργία: Η γεωργία αποτελεί ένας από τους αρχαιότερους τομείς έρευνας και η φιλοσοφία των τρόπων καλλιέργειας μεταδίδεται από τη μια γενιά στην άλλη. Στην σύγχρονη εποχή υπάρχει επείγουσα ανάγκη να παραχθεί περισσότερη τροφή για τον αυξανόμενο πληθυσμό με λιγότερη καλλιεργήσιμη γη. Οι υπεύθυνοι του γεωργικού κλάδου αναζητούν βοήθεια στα τεχνολογικά επιτεύγματα- όπως IoT, ανάλυση μεγάλων δεδομένων και cloud computing–τα οποία τους παρέχουν την ικανότητα παρακολούθησης και πρόβλεψης φυσικών φαινομένων, συλλογή δεδομένων πραγματικού χρόνου που βρίσκονται στις καλλιεργήσιμες εκτάσεις και στα γεωργικά μηχανήματα. Τα Big Data παρέχουν στους αγρότες αναλυτικά δεδομένα σχετικά με τα μοτίβα βροχοπτώσεων, τις απαιτήσεις λιπασμάτων, την υγρασία του εδάφους κ.ά. που επικρατούν στις αγροτικές περιοχές, καθοδηγώντας τους στην επιλογή του καταλληλότερου φυτού και την αποδοτικότερη εκμετάλλευση της διαθέσιμης γης. Επιπρόσθετα, τα δεδομένα που εξάγονται από τους αισθητήρες του γεωργικού εξοπλισμού, αξιοποιούνται για την διαχείριση των εξόδων για καύσιμα, ηλεκτρική ενέργεια και για την εύρεση του χρήσιμων μηχανημάτων για κάθε συνθήκη. Τέλος, δυστυχώς η παραγωγή τροφίμων αρκετές φορές συσχετίζεται με ασθένειες και περιβαλλοντικές καταστροφές που επηρεάζουν εκατομμύρια ανθρώπους. Οι παγκόσμιοι περιβαλλοντικοί οργανισμοί αναλύουν τα δεδομένα και θέτουν αυστηρούς κανονισμούς για την χρήση χημικών φυτοφαρμάκων για την μείωση των μολύνσεων, την διατήρηση του οικοσυστήματος και την αύξηση της απόδοσης των καλλιεργειών.

Ναυτιλία: Στο ναυτιλιακό τομέα, από την αρχαιότητα, η σωστή παρακολούθηση του φορτίου διασφαλίζει την ασφάλεια του περιεχομένου των πλοίων. Τα μεγάλα δεδομένα χρησιμοποιούνται για την διαχείριση των αισθητήρων από τα πλοία εξυπηρετώντας την προγνωστική ανάλυση, η οποία είναι απαραίτητη για την αποφυγή καθυστερήσεων και τη βελτίωση της συνολικής λειτουργικής αποτελεσματικότητας του ναυτιλιακού κλάδου. Μέσω της επεξεργασίας των δεδομένων μπορούν να εξαχθούν οι βασικές αιτίες των απωλειών καραβιών και εμπορευμάτων εντός ή εκτός τερματικών σταθμών και τους τρόπους αποφυγής δαπανηρών προβλημάτων. Τα πλοία, όπως και τα αυτοκίνητα, μπορούν να λειτουργήσουν σε βέλτιστες ταχύτητες με μικρή κατανάλωση ενέργειας, όμως η αποδοτικότητα με την πάροδο του χρόνου μειώνεται εξαιτίας της φθοράς των μηχανικών μερών του κινητήρα. Επιπλέον, οι λιμενικοί χρειάζονται να γνωρίζουν πληροφορίες για την εκτιμώμενη ώρα άφιξης και το περιεχόμενο του φορτίου. Οποιεσδήποτε αλλαγές στην ταχύτητα, στην ώρα αναχώρησης/άφιξης και στο βάρος του φορτίου

παρακολουθούνται, αφού επηρεάζουν σημαντικά την κερδοφορία του κάθε εμπλεκόμενου οργανισμού. Η ανάλυση των δεδομένων επωφελή τους πλοιοκτήτες καθώς τους παρουσιάζονται αναλυτικά τα κόστη για καύσιμα και συντήρηση και οι περιβαλλοντικοί ρύποι, οδηγώντας τους στην λήψη αποφάσεων.

Αθλητισμός: Ο αθλητισμός πραγματοποιούσε στατιστική ανάλυση πολύ πριν εμφανιστεί ο επιστημονικός κλάδος της επιστήμης των δεδομένων. Εκατοντάδες εκατομμύρια αθλητικά δεδομένα παράγονται από διάφορες αθλητικές ομάδες. Οι εφαρμογές των Big Data αξιοποιούνται από τους αθλητικούς οργανισμούς και τις εταιρίες χρηματοδότησης των ομάδων, καθώς μπορούν να έχουν στην διάθεση τους τα στατιστικά στοιχεία για την απόδοση των παιχτών και να γνωρίζουν την κατάσταση της υγείας τους. Οι παίκτες κατανοούν τις αδυναμίες τους από τα δεδομένα της απόδοσής τους, τα οποία τους προσδιορίζουν τους στόχους της φυσικής κατάστασης για να έχουν με την μέγιστη απόδοση με την ελάχιστη κόπωση κατά την διάρκεια των αγώνων και την οικονομική τους αξία. Η μεγαλύτερη κερδοφορία στον αθλητικό τομέα εμφανίζεται από τα παιχνίδια στοιχημάτων. Οι θαυμαστές των αθλητικών αγώνων αναλύουν συνεχώς δεδομένα πραγματικού χρόνου, είτε πρόκειται για την παροχή ενημερώσεων play-by-play είτε για την συζήτηση προβλέψεων. Ο ιδρυτής της ιστοσελίδας «Advanced NFL Stars» Brian Burke [34], δηλώνει ότι χρησιμοποιώντας τα μεγάλα δεδομένα προπονητές και παίκτες μπορούν να προβλέψουν τα αποτελέσματα του αγώνα. Τέλος, αναπόσπαστο μέρος όλων των αθλητικών εκδηλώσεων αποτελούν οι επιβλέποντες/διαιτητές, οι οποίοι κατά την διάρκεια εκπαίδευσης τους έχουν πρόσβαση σε λάθη αθλητών και προβλήματα που δημιουργήθηκαν σε παλαιότερες αθλητικούς αγώνες. Αναπτύσσονται μοντέλα συσχέτισης παραβατικότητας και τρόπο επίλυσης για μη ομαλότερη διεξαγωγή των αθλητικών αγώνων.

2.5: Τι δεν προσφέρουν Big Data;

Τα δεδομένα που συλλέγονται μπορούν να ταξινομηθούν ή να παρουσιαστούν με την μορφή τυχαίων πληροφοριών. Εάν οι πληροφορίες είναι κατεστραμμένες και σημασιολογικά ανοργάνωτες, πολλοί χρήστες μπορεί να τις παρερμηνεύσουν εξάγοντας λανθασμένα συμπεράσματα. Παρά την σπουδαιότητα των μεγάλων δεδομένων, υπάρχουν μειονεκτήματα, τα οποία οι επιστήμονες προσπαθούν να εξαλείψουν.

Ασφάλεια(Security):Η αποθήκευση και η μεταφορά μεγάλων δεδομένων, ιδιαίτερα των ευαίσθητων αποτελεί ελκυστικό στόχο για κυβερνοεπιθέσεις. Στην έρευνα της AtScale το 2019, το 80% των ερωτηθέντων χρηστών του internet δήλωσαν πως αισθάνονται ανασφάλεια για τα προσωπικά τους δεδομένα που υπάρχουν στο internet. Με την αύξηση της παγκόσμιας πολιτικής κρίσης και τις περίπλοκες καταστάσεις μεταξύ των εθνών, τα δεδομένα που έχουν διαρρεύσει μπορούν να χρησιμοποιηθούν με δόλιο σκοπό. Επιπλέον, στον γενικότερο τομέα της επιστήμης των υπολογιστών υπάρχει ανάγκη για συμμόρφωση με τους κυβερνητικούς κανονισμούς. Όλες οι

πληροφορίες, που περιλαμβάνονται στα αποθήκες δεδομένων, πρέπει να διασφαλιστεί ότι πληρούν τα πρότυπα του κλάδου ή τις κρατικές απαιτήσεις κατά τον χειρισμό τους.

Οικονομικό Κόστος (Economic Cost):Τα μεγάλα δεδομένα μπορούν να εντοπίσουν τους αποτελεσματικότερους τρόπους για εξοικονόμηση επιχειρησιακών πόρων, αλλά ταυτόχρονα μπορούν να επιβαρύνουν οικονομικά την εταιρία/τον οργανισμό. Οι δαπάνες σχετίζονται με την εφαρμογή λογισμικού, τις τακτικές ενημερώσεις, την συντήρηση, την αποθήκη δεδομένων σε επίπεδο εξοπλισμού, την εκπαίδευση εργαζομένων στις νέες τεχνολογίες και στην πρόσληψη επιστημόνων δεδομένων για την διαχείριση των μεγάλων δεδομένων. Πολλά από τα εργαλεία διαχείρισης βασίζονται στην τεχνολογία ανοιχτού κώδικα, η οποία μειώνει το κόστος λογισμικού. Εξίσου κοστοβόρο είναι η ανάγκη σε υλικό εξοπλισμού για την δημιουργία αποθηκών δεδομένων(clusters,servers). Ορισμένοι οργανισμοί/επιχειρήσεις για να περιορίσουν αυτό το ζήτημα στρέφονται σε τεχνολογίες νέφους (cloud based technology) -όπως AWS- , αλλά αυτό δεν εξαλείφει εντελώς τα προβλήματα υποδομής. Δεν είναι ασυνήθιστο η διαχείριση των μεγάλων δεδομένων να υπερβαίνει τον προϋπολογισμό και τον απαιτούμενο χρόνο σε σχέση με άλλες διεργασίες ενός συστήματος.

Δυσκολία ενσωμάτωσης στα παλαιά συστήματα (Difficulty integrating legacy systems):Το λογισμικό που αναπτύσσεται θα πρέπει να είναι πάντα συμβατό με τις υπάρχοντες τεχνολογίες. Τα συστήματα που έχουν σχεδιαστεί πριν από 30~20 χρόνια δεν παρέχουν την δυνατότητα επέκτασης τους, μέσω της ενσωμάτωσης τους σε άλλα συστήματα. Τα παλαιού τύπου συστήματα τείνουν να είναι αργά, καθιστώντας τα ακατάλληλα για την διαχείριση του όγκου των μεγάλων δεδομένων. Ταυτόχρονα είναι ευάλωτα ως προς τις επιθέσεις κακόβουλων εισβολέων (hacker). Παρόλα αυτά, οι αποθηκευμένες πληροφορίες είναι πολύτιμες για την ανάλυση και τη λήψη αποφάσεων. Ο κλάδος των μεγάλων δεδομένων έχει εδραιωθεί σε όλους τους τομείς της τεχνολογίας τα τελευταία χρόνια, αναδεικνύοντας την ανάγκη για εκπαίδευση του υπάρχοντος προσωπικού που τις περισσότερες φορές δεν είναι εξοικειωμένο με τα νέα εργαλεία.

Κεφάλαιο 3: NoSQL

Τα Big Data έχουν υψηλές υπολογιστικές και αποθηκευτικές απαιτήσεις και τα περισσότερα δεδομένα που παράγονται δεν έχουν συγκεκριμένη μορφή και δομή. Τα παραδοσιακά μοντέλα διαχείρισης δεδομένων δεν καλύπτουν αυτές τις ανάγκες, δημιουργώντας διάφορα προβλήματα. Προκειμένου να καταπολεμήσουν αυτά τα προβλήματα, οι ερευνητές του κλάδου της επιστήμης των δεδομένων τροποποίησαν τις σχεσιακές βάσεις, αναπτύσσοντας τις μη-σχεσιακές βάσεις,NoSQL.

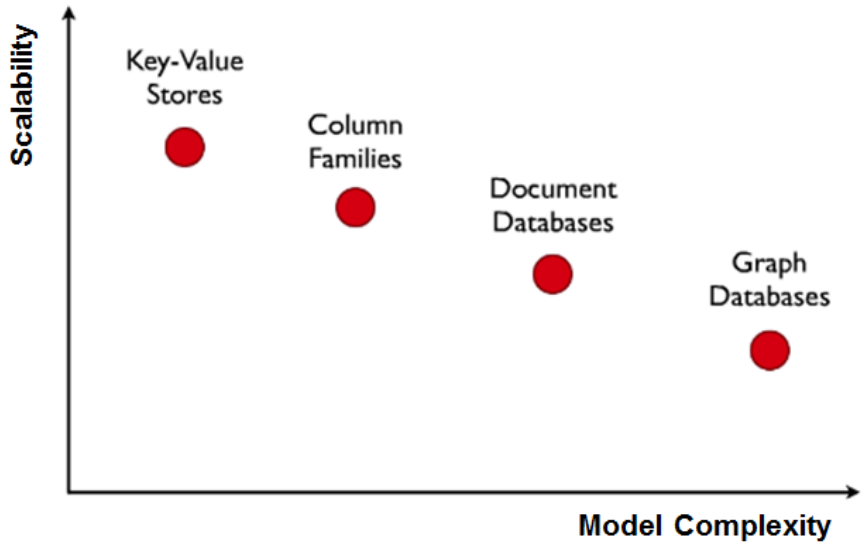
Εικόνα 4: NoSQL

3.1: Τι είναι NoSQL;

Πολλοί επιστήμονες θεωρούν την NoSQL -NotOnlySQL- ως την σύγχρονη βάση δεδομένων, διότι προσαρμόζεται στις απαιτήσεις του Παγκόσμιου Ιστού και του Διαδικτύου των πραγμάτων. Οι NoSQL βάσεις δεδομένων είναι μια μη-σχεσιακή βάση δεδομένων, αποθηκεύοντας και αποκτώντας πρόσβαση σε δεδομένα με την χρήση κλειδιών-τιμών (***key-value***). Το κάθε δεδομένο αποθηκεύεται **ξεχωριστά** με ένα **μοναδικό** κλειδί, παρέχοντας ευελιξία στην αποθήκευση.

Οι NoSQL ανήκουν στην κατηγορία συστημάτων διαχείρισης DB που δεν ακολουθούν όλους του κανόνες των RDBMS, χωρίς να αξιοποιούν την SQL. Για την υποστήριξη των παλαιών συστημάτων που βασίζονται σε RDBMS και την ομαλή μετάβαση των συστημάτων από τις παραδοσιακές τεχνικές διαχείρισης στις μη-σχεσιακές, τα NoSQL συστήματα υποστηρίζουν γλώσσες ερωτημάτων που μοιάζουν με την SQL. Δεδομένου ότι για τις NRDBMS δεν είναι απαραίτητη προϋπόθεση η ύπαρξη προκαθορισμένου σχεδίου (**schema database**), προσφέρεται η δυνατότητα για ταχεία κλιμάκωση διαχείρισης μεγάλων δεδομένων.

3.2: Είδη NoSQL βάσεων δεδομένων



Πίνακας 3: Σύγκριση των Μοντέλων δεδομένων NoSQL ως προς την πολυπλοκότητα και την κλιμάκωση

Κλειδί-τιμή(Key-Value stores):Οι βάσεις δεδομένων κλειδιού-τιμής αποτελούν την απλούστερη μορφή δεδομένων. Κάθε στοιχείο ταξινομείται με ένα μοναδικό σημασιολογικό ζεύγος key-value. Αυτός ο τύπος NoSQL DB έχει μια δομή δεδομένων λεξικού, που αποτελείται από ένα σύνολο αντικειμένων και αντιπροσωπεύουν πεδία δεδομένων. Η χρήση του κλειδιού μπορεί να παρομοιαστεί το primary key που υπάρχει στις RDBMS, ενώ η τιμή είναι μια σειρά απλών τύπων δεδομένων, όπως συμβολοσειρές. Η ανάκτηση των αποθηκευμένων δεδομένων υλοποιείται με την χρήση συγκεκριμένου κλειδιού και στην συνέχεια λαμβάνεται η τιμή που έχει εκχωρηθεί στο κλειδί. Χάρη στην απλότητα τους, εμφανίζουν μεγαλύτερη ευελιξία ως προς την αποθήκευση και ταχύτερη απόδοση, επιτρέποντας την οριζόντια κλιμάκωση μεγάλων ποσοτήτων δεδομένων. Ωστόσο, δεν είναι ιδανικό για ταυτόχρονη εξόρυξη δεδομένων. Οι περιπτώσεις χρήσης των βάσεων δεδομένων κλειδιού-τιμής περιλαμβάνουν καλάθια ηλεκτρονικών αγορών, προτιμήσεις χρηστών και προφίλ χρηστών σε όλες τις ηλεκτρονικές πλατφόρμες.

Ευρείας στήλης(Wide-Column families): Οι βάσεις δεδομένων ευρείας στήλης αποθηκεύουν τις πληροφορίες σε στήλες, ενώ στις RDBMS τα δεδομένα αποθηκεύονται σε σειρές και διαβάζονται πλειάδα προς πλειάδα. Οι NoSQL τύπου ευρείας στήλης έχουν σχεδιαστεί με σκοπό την αποτελεσματικότερη ανάγνωση δεδομένων και την ταχύτερη ανάκτηση τους. Κατά την διαδικασία εφαρμογής προγραμμάτων για ανάλυση των δεδομένων, δεν καταναλώνεται μνήμη για τις μη-χρήσιμες και ανεπιθύμητες πληροφορίες. Επιπλέον, στα κατανεμημένα συστήματα οι στήλες που δεν υποβάλλονται συχνά σε ερωτήματα κατανέμονται σε διαφορετικούς και απομακρυσμένους κόμβους, ώστε να μην υπερφορτώνεται ο κεντρικός κόμβος (διακομιστής).Το Apache Cassandra είναι παράδειγμα βάσεων δεδομένων ανοιχτού κώδικα, ευρείας στήλης και

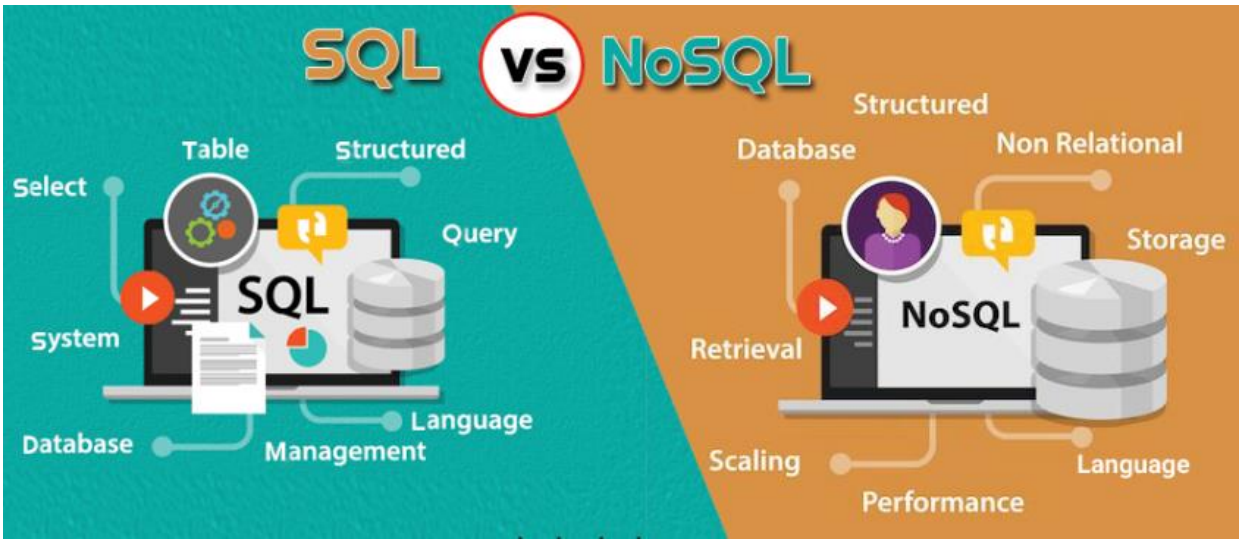
είναι σχεδιασμένο να διαχειρίζεται μεγάλες ποσότητες δεδομένων σε πολλούς διακομιστές και ομαδοποίηση που εκτείνεται σε κέντρα πολλαπλών δεδομένων.

Εγγραφή (Document databases):Οι βάσεις δεδομένων εγγράφων αποθηκεύουν τα δεδομένα σε έγγραφα JSON, BSON ή XML. Τα έγγραφα είναι ένθετα και μπορούν να ταξινομηθούν για ταχεία αναζήτηση. Τα δεδομένα διατηρούνται ενωμένα όταν χρησιμοποιούνται σε εφαρμογές, μειώνοντας τον όγκο μετάφρασης που απαιτείται για την χρήση τους. Η βάση δεδομένων εγγράφων προσφέρει την ευελιξία ως προς το schema, καθώς δεν χρειάζεται να υπάρχει ομοιότητα στις μεταβλητές μεταξύ των εγγράφων. Παρόλα αυτά, η ανομοιότητα μεταξύ των εγγράφων μπορεί να δημιουργήσει μείζονα ζήτημα στις περιπτώσεις των πολύπλοκων συναλλαγών, προκαλώντας καταστροφή των πολύτιμων πληροφοριών.

Γραφήματα(Graph databases): Οι βάσεις δεδομένων γραφήματος εστιάζουν στην σχέση μεταξύ των στοιχείων δεδομένων. Κάθε στοιχείο αποθηκεύεται ως κόμβος, ακμή και ιδιότητα. Οι κόμβοι συνδέονται μεταξύ τους με τις ακμές, προσδιορίζοντας τις συσχετίσεις μεταξύ των δεδομένων. Μια βάση δεδομένων γραφημάτων είναι βελτιστοποιημένη για να καταγράφει και να αναζητά τις συνδέσεις μεταξύ των στοιχείων δεδομένων, ξεπερνώντας τις δυσκολίες από την χρήση του JOINσε πολλαπλούς πίνακες στην SQL. Ως αποτέλεσμα, όλες οι διεργασίες που πραγματοποιούνται στις βάσεις δεδομένων γραφήματος εκτελούνται παράλληλα.

3.3: NoSQL VS RDBMS

Οι NoSQL βασίζονται στις SQL βάσεις δεδομένων, αλλά διαφοροποιηθεί με την προσθήκη χαρακτηριστικών, τα οποία περιλαμβάνουν την έλλειψη schema database, την ομαδοποίηση δεδομένων και την υποστήριξη διπλότυπων.



Εικόνα 3: SQL VS NoSQL

Ακεραιότητα των δεδομένων (Data Integrity):Οι βάσεις δεδομένων SQL και NoSQL χρησιμοποιούν διαφορετικές προσεγγίσεις για την προστασία της ακεραιότητας των δεδομένων, καθώς δημιουργούνται, διαβάζονται, ενημερώνονται και διαγράφονται από διάφορους χρήστες και εφαρμογές. Οι SQL βάσεις δεδομένων βασίζονται στο μοντέλο ACID. Το μοντέλο βασίζεται σε τέσσερις βασικούς πυλώνες: Ατομικότητα(Atomicity), Συνοχή(Consistency), Απομόνωση(Isolation) και Ανθεκτικότητα(Durability). Κάθε διεργασία στον server: (α) εκτελείται μόνη της είτε επιτυχημένα είτε αποτυχημένα,(β) χωρίς να παραβιάζει τους περιορισμούς που έχουν ορισθεί για το σύνολο των δεδομένων και (γ) αποκρύπτεται από τις άλλες διεργασίες μέχρι να ολοκληρωθεί η εκτέλεσή της. Οι αλλαγές που θα προκύψουν στα δεδομένα διατηρούνται στον server. Αντίθετα, οι NoSQL βάσεις δεδομένων βασίζονται στο μοντέλο BASE. Το μοντέλο αποτελείται από τρεις βασικές ιδιότητες: Βασική Διαθεσιμότητα (Basic Availability), Μεταβλητότητα Συνοχής(Soft-State) και Ενδεχόμενη Συνέπεια(Eventual Consistency). Τα δεδομένα: (α) τις περισσότερες φορές είναι διαθέσιμα, (β) αναπαράγονται και διαδίδονται συνεχώς, προκαλώντας μερική ασυνέχεια στην ροή των δεδομένων για μικρό χρονικό διάστημα και (γ) η συνέπεια θα επανέλθει σε άγνωστη στιγμή. Ορισμένες εφαρμογές παρουσιάζουν ανοχή ως προς την ασυνέπεια των δεδομένων.

Σχηματισμός(Schema): Μία από τις σημαντικότερες διαφορές μεταξύ SQL και NoSQL βάσεων δεδομένων αποτελεί ο τρόπος σχεδιασμός μιας DB. Οι NoSQL βάσεις δεδομένων είτε δεν ακολουθούν κάποιο προκαθορισμένο σχεσιακό μοντέλο, είτε διαθέτουν δυναμικό σχεσιακό μοντέλο. Η έλλειψη ή χαλαρότητα στο schema, εξυπηρετεί τους τύπους δεδομένων στις NRDBMS. Οι NoSQL DB είναι εύκαμπτες ως προς τις σχεσιακές αλλαγές. Αντίθετα, οι SQL βάσεις δεδομένων λειτουργούν σύμφωνα με ένα αυστηρά προκαθορισμένο σχεσιακό μοντέλο. Το ίδιο αυστηρό και προκαθορισμένο schema υποχρεωτικά διατηρούν τα δεδομένα. Οι SQL DB είναι

άκαμπτες στις πιθανές αλλαγές του schema, απαιτώντας λεπτομερή ανάλυση στις ανάγκες ενός συστήματος πριν το σχεδιασμό της βάσης δεδομένων του.

Επεκτασιμότητα(Scalability): Σημαντική διαφορά αποτελεί ο τρόπος επέκτασης μιας βάσης δεδομένων, διότι τα δεδομένα αυξάνονται με εκθετικούς ρυθμούς. Οι περισσότερες SQL βάσεις δεδομένων, χρησιμοποιούν έναν μοναδικό server για την αποθήκευση κάθε DB, υποστηρίζοντας την κάθετη κλιμάκωση. Αυτή η κλιμάκωση επιβαρύνει είτε τον server με την αύξηση των διεργασιών που έχει να εκτελέσει, μειώνοντας την αποδοτικότητά του είτε οικονομικά τον οργανισμό/εταιρεία χρήσης της βάσης δεδομένων, με την προσθήκη ενός νέου server ή και αντικατάσταση του υπάρχοντος με μεγαλύτερες δυνατότητες. Αντίθετα, οι βάσεις δεδομένων NoSQL χρησιμοποιούν ένα σύστημα κατανεμημένων servers διαφορετικής τεχνολογίας ο καθένας, υποστηρίζοντας την οριζόντια κλιμάκωση. Αυτή η κλιμάκωση προσφέρει τη δυνατότητα διάσπασης των συνόλων δεδομένων σε μικρότερους servers, βελτιώνοντας την διαχείρισης μεγάλων ποσοτήτων δεδομένων.

Ανίχνευση απάτης και έλεγχος ταυτότητας: Η προστασία προσωπικών δεδομένων και η διασφάλιση πρόσβασης στις εφαρμογές μόνο πραγματικών χρηστών/πελατών αποτελεί την κορυφαία προτεραιότητα όλων των εταιρειών ανάπτυξης λογισμικού. Ιδιαιτέρως στις χρηματοοικονομικές και υγειονομικές υπηρεσίες. Οι hackers συνεχώς αναζητούν τρόπους να παραβιάσουν τις δικλίδες ασφαλείας, ώστε να μπορέσουν να έχουν πρόσβαση σε αυτές τις ευαίσθητες και πολύτιμες πληροφορίες. Για την ανίχνευση παραπλανητικών πράξεων, απαιτείται συνεχή ανάλυση δεδομένων σε πραγματικό χρόνο όλων των τύπων δεδομένων, ώστε να εντοπιστούν ασυνήθιστες ενέργειες των χρηστών του συστήματος. Αυτές οι ανωμαλίες μπορούν να ανιχνευθούν ακόμα και πριν συμβεί οποιαδήποτε απάτη. Ο συνδυασμός της αυτοματοποιημένης ανάλυσης σε πραγματικό χρόνο, των μεγάλων και συνεχώς αυξανόμενων συνόλων δεδομένων, των πολυάριθμων τύπων data, μαζί με την διεξαγωγή μοντέλων μηχανικής μάθησης και τεχνητής νοημοσύνης, καθιστούν τις NoSQL βάσεις ιδανικές για τον εντοπισμό απατών και πιστοποιήσεων ταυτότητας χρηστών.

3.4: Εφαρμογές NoSQL

Διαχείριση ηλεκτρονικού περιεχομένου (Content Management): Οι διαδικτυακές αγορές ξεπερνούν τις «φυσικές» πωλήσεις και το οπτικοακουστικό υλικό κυριαρχεί σε χιλιάδες ηλεκτρονικές αγορές και στις βιτρίνες των ιστοσελίδων. Οι εταιρίες διαδικτυακών πωλήσεων αξιοποιούν τα πολυμεσικά εργαλεία συμπεριλαμβανομένου του υλικού (φωτογραφία, βίντεο, κριτική του προϊόντος) που δημιουργείται από χρήστες στα μέσα κοινωνικής δικτύωσης, προσδίδοντας την ψευδαίσθηση της στιγμιαίας αλληλεπίδρασης με ένα προϊόν/υπηρεσία στους μελλοντικούς αγοραστές. Οι NoSQL document βάσεις προσφέρουν ένα ευέλικτο, ανοιχτού τύπου μοντέλο δεδομένων το οποίο είναι ιδανικό για την αποθήκευση ενός κράματος όλων των τύπων

δεδομένων. Επιπλέον, καθίσταται η δυνατότητα για συγκέντρωση δεδομένων που εξυπηρετούν πολλαπλές επιχειρηματικές εφαρμογές σε μια ενιαία βάση δεδομένων καταλόγου. Ενώ οι σχεσιακές βάσης δεδομένων διαχείρισης με τα σταθερά μοντέλα τείνει να έχει ως αποτέλεσμα τον πολλαπλασιασμό πολλαπλών, επικαλυπτόμενων καταλόγων για διαφορετικούς σκοπούς. Χαρακτηριστικό παράδειγμα του Content Management με NoSQL αποτελεί ο δημοσιογραφικός κολοσσός Forbes [44].Το Forbes ανέπτυξε ταχύτατα ένα προσαρμοσμένο σύστημα διαχείρισης περιεχομένου βασισμένο στο MongoDB μέσα σε λίγους μήνες παρέχοντας μεγαλύτερη ευελιξία με χαμηλότερο κόστος. Οι οικονομικοί πόροι της εταιρείας προέρχονται από τις προβολές των άρθρων, τις διαφημίσεις και την ενσωμάτωση του περιεχομένου συνεργατών και τον διαμοιρασμό (Share→clickstream) στα social media.

Εφαρμογές για κινητά (Mobile applications): Η χρήση κινητών τηλεφώνων και tablet ξεπέρασε τους ηλεκτρονικούς υπολογιστές ως η κορυφαία διαδικτυακή πλατφόρμα για αναζήτηση πληροφοριών, αγορές και προβολή περιεχομένου ιστοσελίδων. Ενδιαφέρον αποτελεί το γεγονός ότι το 90% των δεδομένων κινητής τηλεφωνίας εξυπηρετείται μέσω εφαρμογών και μόνο το 10% μέσω φυλλομετρητών (Browser apps) μια συντριπτική αλλαγή τα τελευταία χρόνια [Google statics]. Η ταχεία κλιμάκωση mobile-apps παγκοσμίως για την εξυπηρέτηση δεκάδων εκατομμυρίων χρηστών απαιτεί συχνά κατανεμημένες βάσεις δεδομένων, οι οποίες με την σειρά τους απαιτούν την υποστήριξη από τις τεχνολογίες NoSQL. Τα ευέλικτα μοντέλα των δεδομένων NoSQL έχουν αντοχή στους γρήγορους κύκλους ενημέρωσης εφαρμογών καλύτερα από τα μοντέλα σχεσιακών δεδομένων σε πολλές περιπτώσεις. Πλέον οι περισσότερες επιχειρήσεις επιθυμούν να αυξήσουν τα έσοδα τους από το περιεχόμενο ιστότοπων, χρησιμοποιώντας NoSQL data stores για τις εφαρμογές τους. Η mobile εφαρμογή The Weather Channel αποτελεί το βέλτιστο παράδειγμα χρήση NoSQL βάσεων [44]. Η συγκεκριμένη εφαρμογή είναι προ-εγκατεστημένη στις περισσότερες συσκευές κινητής τηλεφωνίας και tablet και χρησιμοποιείται κυρίως παθητικά από τους χρήστες καταναλώνοντας ελάχιστους ενεργειακούς πόρους. Το The Weather Channel διαχειρίζεται εκατομμύρια αιτήματα ανά λεπτό, ενώ παράλληλα επεξεργάζεται δεδομένα χρηστών και καταφέρνει να υλοποιήσει ενημερώσεις καιρού από δεκάδες χιλιάδες τοποθεσίες σε όλο τον κόσμο.

Εμπλουτισμός ψηφιακής εμπειρίας (Digital Customer Experience): Μια συναρπαστική διαφοροποιημένη ψηφιακή εμπειρία βασιζόμενη στις δυνατότητες υψηλής έντασης δεδομένων, κρίσιμες για το χρόνο, όπως η εξατομίκευση, η διαχείριση προφίλ χρήστη και μια ενοποιημένη άποψη του πελάτη σε όλα τα σημεία επαφής με την εταιρεία που παρέχει ένα προϊόν ή μια ηλεκτρονική υπηρεσία. Πολλά δημογραφικά, συμπεριφοριστικά και υλικοτεχνικά δεδομένα προέρχονται από διαδικτυακά clickstreams, δημιουργώντας ένα φόρτο εργασίας πολλών schema εγγραφής που δυσκολεύουν την λειτουργία των σχεσιακών βάσεων διαχείρισης δεδομένων. Μια κατανεμημένη βάση δεδομένων NoSQL μπορεί να κλιμακωθεί με χαμηλό οικονομικό κόστος, να διαχειριστεί ένα συνεχώς αυξανόμενο αριθμό χαρακτηριστικών με λιγότερη διοικητική ταλαιπωρία και χωρίς να υπάρχουν χρονικές καθυστερήσεις. Ο παγκόσμιος πάροχος πολυμεσικών υπηρεσιών Comcast χρησιμοποιεί μια πλατφόρμα Couch base NoSQL για να προσφέρει θετική

εμπειρία υποστήριξης πελατών σε πολλαπλούς τομείς δραστηριότητας. Η πλατφόρμα καταγράφει δεδομένα από άπειρους αριθμούς καναλιών αλληλεπίδρασης και τα συσχετίζει με τους λογαριασμούς και την κατάσταση της υπηρεσίας μεμονωμένων πελατών, εμφανίζοντας ανάγκη για συνεχή επεκτασιμότητα και για ανθεκτικότητα.

3.5: Τι δεν προσφέρουν;

Οι NoSQL βάσεις δεδομένων αναπτύχθηκαν στις αρχές του 21ου αιώνα και είναι αρκετά νεότερες σε σχέση με τις SQL. Οι μη σχεσιακές βάσεις δεδομένων αποτελούν μία από τις μεγαλύτερες καινοτομίες του κλάδου της επιστήμης των δεδομένων, προσφέροντας ευελιξία ως προς την επεκτασιμότητα. Δυστυχώς, ακόμα δεν υπάρχει ολοκληρωμένη βιβλιογραφική τεκμηρίωση και παρατηρούνται αρκετά σημεία που χρήζουν βελτίωσης ώστε να χρησιμοποιούνται ευρέως [19].

Μη ύπαρξη τυποποιημένης γλώσσας(No standardized language): Δεν υπάρχει τυπική γλώσσα για την διενέργεια ερωτημάτων NoSQL. Η σύνταξη των ερωτημάτων διαφοροποιείται ανάλογα με τον τύπο των δεδομένων και δυσκολεύει την ευρεία χρήση της. Η ευελιξία των NoSQL ερωτημάτων είναι λιγότερη σε σχέση με την αποθήκευση τους. Συνήθως, δεν μπορούν να επιβάλουν ή και να εγγυηθούν την μοναδικότητα των κλειδιών εντός της βάσεις δεδομένων, όπως τα σχεσιακά συστήματα διαχείρισης δεδομένων. Η αξιοποίηση NRDBMS καθίσταται αναποτελεσματική σε εφαρμογές, όπου προβλέπεται αυστηρά η χρήση μοναδικών κλειδιών και τιμών.

Ασυνέπεια ανάκτησης δεδομένων(Data retrieval inconsistency): Οι NoSQL βάσεις δεδομένων αξιοποιούν τους κατανεμημένους servers, για ταχεία διαθεσιμότητα. Ταυτόχρονα, δυσχεραίνεται η συνέπεια των δεδομένων, αυξάνοντας τις πιθανότητες για ανολοκλήρωτα αποτελέσματα από τα ερωτήματα που δέχεται η βάση δεδομένων. Επίσης, ίσως να επιστρέφονται διαφορετικά αποτελέσματα ανάλογα των server που θα είναι διαθέσιμος για ανταπόκριση του ερωτήματος. Όπως προαναφέρθηκε, οι NoSQL βασίζονται στο πρότυποBASE, όπου περιέχει **ενδεχόμενη συνέπεια**. Όμως, οι SQL θεμελιώνονται με το μοντέλο ACID, στο οποίο τα δεδομένα πρέπει να είναι έγκυρα και συνεπή κατά την διάρκεια των διεργασιών στις βάσεις δεδομένων. Η ασυνέπεια, στις εφαρμογές εμφανίζεται για μερικά νανοδευτερόλεπτα και δεν είναι αντιληπτή από τον χρήστη.

Ασφάλεια (Security): Όλες οι βάσεις δεδομένων διατηρούν αντίγραφα όλων των δεδομένων και των διεργασιών που έχουν συμβεί, ακόμα και αν κάποιος χρήστης/διαχειριστής/προγραμματιστής τα έχει χειροκίνητα διαγράψει. Δυστυχώς, οι NoSQL βάσεις δεδομένων δεν επιτρέπουν την δημιουργία αντιγράφων. Τα τελευταία χρόνια, τα εργαλεία NoSQL παρέχουν κάποια εργαλεία για

την δημιουργία αντιγράφων ασφαλείας, αλλά δεν είναι αρκετά ώριμα για να εξασφαλισθεί η καταλληλότερη και η ασφαλέστερη διαχείριση της βάσης δεδομένων.

Κεφάλαιο 4: Apache Cassandra

Εικόνα 5: Λογότυπο Apache Cassandra

4.1 Τι είναι το Apache Cassandra;

Το εργαλείο Apache Cassandra ανήκει στην οικογένεια Ευρειών Στηλών των NoSQL βάσεων δεδομένων ανοιχτού κώδικα. Έχει σχεδιαστεί για τον χειρισμό Big Data που κατανέμονται σε πολλούς servers. Με την χρήση του Cassandra διασφαλίζεται η διαρκής διαθεσιμότητα ενός συστήματος και ελάχιστα σημεία αποτυχίας, εξαιτίας της peer-to-peer αρχιτεκτονικής, της οριζόντιας επεκτασιμότητας, της ανοχής σε σφάλματα και τους ταχύτατους χρόνους απόκρισης.

Το Cassandra αναπτύχθηκε το 2008 από τον Avinash Lakshman που εργαζόταν στην εταιρία Metaverse (πρώην Facebook), με σκοπό η πλατφόρμα κοινωνικής δικτύωσης Facebook να μπορεί να είναι πάντα διαθέσιμη και να επεκτείνεται χωρίς να προκαλεί πρόβλημα στο προ-υπάρχον σύστημα. Το 2009 ο Lakshman σε συνεργασία με το Prashant Malik εκδώσαν επιστημονικό άρθρο με τις αρχές του Cassandra. Το Μάρτιο 2009, το εργαλείο Cassandra εξαγοράστηκε και περιλαμβάνεται στα project του ιδρύματος Apache. Μέχρι το 2022 το Cassandra, χρησιμοποιείται από μεγάλο πλήθος εταιριών όπως: Cern, Wood Hole Oceanography Institution, IBM, Metaverse, Netflix κτλ.

4.2 Χαρακτηριστικά

Το εργαλείο Apache Cassandra εμπεριέχει τα βασικά χαρακτηριστικά των NoSQL βάσεων δεδομένων. Όπως όλες οι NoSQL DB είτε δεν ακολουθούν κάποιο προκαθορισμένο schema, είτε διαθέτουν σχεσιακό τμήμα. Το Cassandra βασιζόμενο στις βασικές αρχές των μη σχεσιακών βάσεων δεδομένων, έχει αναπτύξει μηχανισμούς οι οποίοι επεκτείνουν τις δυνατότητες αυτών των αρχών.

Αποκέντρωση (Decentralization): Όπως προαναφέρθηκε το Cassandra είναι κατανεμημένο σύστημα, οι κόμβοι του έχουν τους ίδιους ρόλους. Κάθε κόμβος εμπεριέχει διαφορετικά δεδομένα,

αφού τα σύνολα των δεδομένων κατανέμονται σε όλο το σύμπλεγμα κόμβων. Με την αποκέντρωση του Cassandra διευκολύνεται η λειτουργία του συστήματος και δεν εμφανίζονται σημεία αποτυχίας εκτέλεσης διεργασιών αυξάνοντας την αποδοτικότητά του.

Επεκτασιμότητα (Scalability):Στις NoSQL βάσεις δεδομένων, το σύστημα επεκτείνεται οριζόντια αυξομειώνοντας την κλίμακα του προσθέτοντας ή αφαιρώντας servers. Το λογισμικό του Cassandra περιλαμβάνει έναν εσωτερικό μηχανισμό για να διατηρεί τα δεδομένα του συγχρονισμένα και όταν αντιληφθεί την ύπαρξη ενός νέου server αυτομάτως τροποποιεί το υπάρχον σύστημα για βέλτιστη λειτουργικότητα.

Ανοχή σε σφάλματα (Fault-tolerant): Τα συστήματα πραγματικού χρόνου πρέπει να είναι συνεχώς διαθέσιμα προς χρήση ακόμα και αν παρουσιάζουν σφάλματα. Το Cassandra μπορεί να αντικαταστήσει τους κόμβους που εμφανίζονται σφάλματα, χωρίς να υπάρξει κάποιο χρονικό διάστημα μη λειτουργίας του συστήματος. Εξαιτίας της κατανεμημένης φύσης του εργαλείου αναπαράγει τα δεδομένα του αποτυχημένου κόμβου σε άλλους κόμβους, αποφεύγοντας την διακοπή της λειτουργίας του και ταυτόχρονα διατηρείται η αποδοτικότητά του.

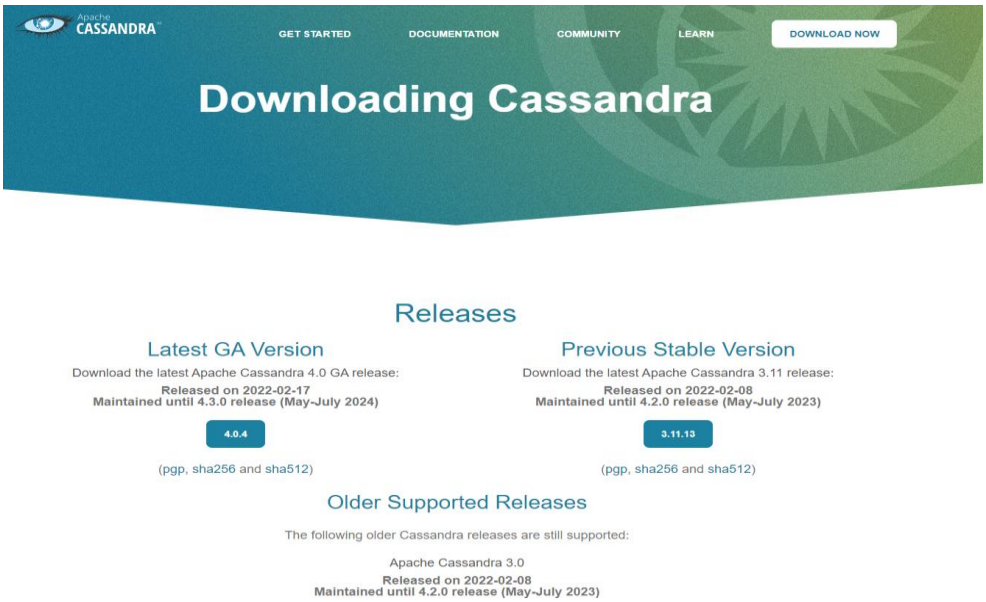
Ρυθμιζόμενη Συνέπεια (Tunable consistency): Οι NoSQL βάσεις δεδομένων, όπως αναφέρθηκε, βασίζονται στο μοντέλο BASE για την διασφάλιση της ακεραιότητας των δεδομένων τους. Ο τελευταίος βασικός πυλώνας αυτού του μοντέλου αφορά στην Ενδεχόμενη Συνέπεια. Ο όρος συνέπεια αναφέρεται ότι κατά την διαδικασία του «ανάγνωσης» μιας πληροφορίας, θα πρέπει να επιστρέφεται η πιο πρόσφατη τιμή που εγγράφθηκε για αυτή την πληροφορία. Το εργαλείο Cassandra επιτρέπει τον προσδιορισμό του μέτρου της συνέπειας, που είναι αναγκαίο για το κάθε σύστημα, εξισορροπώντας το με το επίπεδο διαθεσιμότητας. Μπορούν να προσδιοριστούν τρία επίπεδα συνέπειας: **αυστηρή** (strict), **αιτιώδης** (casual) και **αδύναμη** (weak). Η αυστηρή συνέπεια απαιτεί από το σύστημα σε κάθε «ανάγνωση» να επιστρέφεται πάντα η πιο πρόσφατη «εγγραφή». Αυτό σημαίνει ότι οι κόμβοι θα πρέπει να είναι απόλυτα συγχρονισμένοι για να ανταποκριθούν στις απαιτήσεις του συστήματος. Η αιτιώδης συνέπεια εφαρμόζεται στις περιπτώσεις που υπάρχει ανάγκη για σημασιολογική ακολουθία εγγραφών, ώστε να αναγνωσθούν σε λογική σειρά. Στην αδύναμη συνέπεια όλες οι πιο πρόσφατες εγγραφές θα διαδοθούν σε όλο το κατανεμημένο σύστημα με σχετικά μεγάλη χρονική καθυστέρηση.

Cassandra Query Language (CQL): Αρχικά το εργαλείο Cassandra απαιτούσε την χρήση ενός API, το οποίο εκτελούσε Εισαγωγή, Λήψη και Διαγραφή των δεδομένων από την βάση. Όσο η χρήση του Cassandra αυξανόταν, δημιουργήθηκε η γλώσσα CQL. Η CQL είναι γλώσσα ερωτημάτων, όπως και η SQL, χωρίς την δυνατότητα σύνδεσης πολλών πινάκων. Σε κάθε κόμβο του συστήματος ορίζεται μια σταθερά σημασιολογική λίστα key-value. Η σύνταξη της λίστας απαιτεί ιδιαίτερη προσοχή, καθώς η CQL είναι case insensitive.

4.3Εγκατάσταση σε Windows10

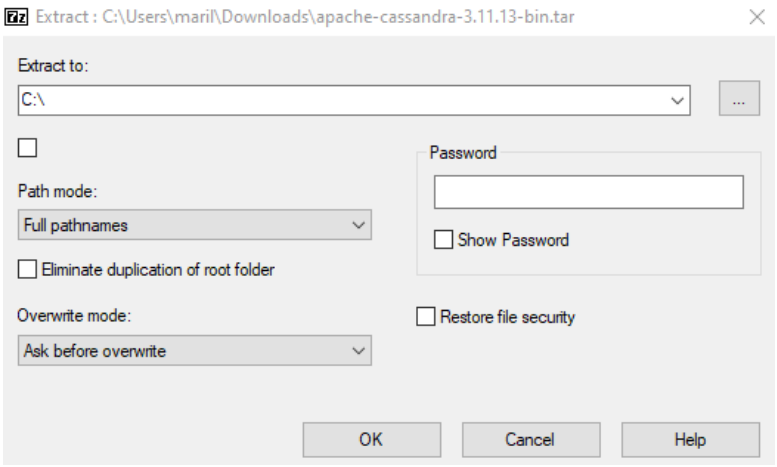
Το εργαλείο Apache Cassandra απαιτεί 2 βασικές προϋποθέσεις για την εγκατάσταση του: Java 8 και Python 2.7.

Βήμα 1: Κατεβάζουμε στον υπολογιστή την έκδοση Cassandra 3.11.18 ή οποιαδήποτε έκδοση προτείνει το website ως σταθερή.



Εικόνα 6:Επιλογή και αποθήκευση κατάλληλου λογισμικού από το επίσημο website

Βήμα 2: Τοποθέτηση του αρχείου στον root φάκελο του υπολογιστή



Εικόνα 7: Επιλογή root φακέλου

Βήμα 3: Διόρθωση σφαλμάτων που παρουσιάζονται.

Για τον έλεγχο της ορθής εγκατάστασης του Apache Cassandra, πρέπει να εκτελεσθεί η εντολή cassandraστο φάκελο C:\apache-cassandra-3.11.13\bin , χρησιμοποιώντας το Command Prompt. Σε κάθε υπολογιστή διαφέρουν τα σφάλματα που θα προκύψουν.

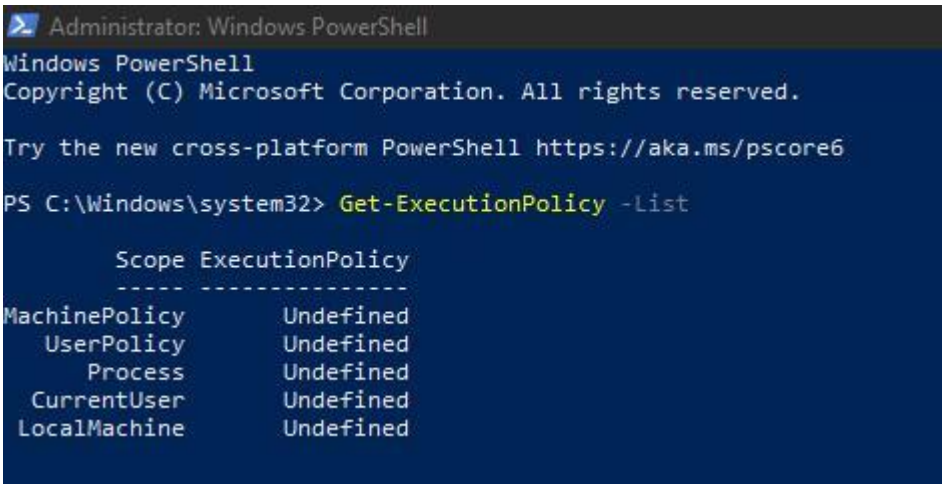
1. Τροποποίηση δικαιωμάτων με την χρήση του PowerShell

```
:\apache-cassandra-3.11.13\bin>cassandra
ARNING! Powershell script execution unavailable.
  Please use 'powershell Set-ExecutionPolicy Unrestricted'
  on this user-account to run cassandra with fully featured
  functionality on this platform.
tarting with legacy startup options
tarting Cassandra Server
NFO  [main] 2022-05-17 14:48:24,666 YamlConfigurationLoader.java:93 - Configuration location: file:/C:/apache-cassandra
3.11.13/conf/cassandra.yaml
NFO  [main] 2022-05-17 14:48:25,570 Config.java:555 - Node configuration:[allocate_tokens_for_keyspace=null; allow_extr
_insecure_udfs=false; allow_insecure_udfs=false; authenticator=AllowAllAuthenticator; authorizer=AllowAllAuthorizer; au
o_bootstrap=true; auto_snapshot=true; back_pressure_enabled=false; back_pressure_strategy=org.apache.cassandra.net.Rate
asedBackPressure{high_ratio=0.9, factor=5, flow=FAST}; batch_size_fail_threshold_in_kb=50; batch_size_warn_threshold_in
kb=5; batchlog_replay_throttle_in_kb=1024; broadcast_address=null; broadcast_rpc_address=null; buffer_pool_use_heap_if_
xhausted=true; cache_load_timeout_seconds=30; cas_contention_timeout_in_ms=1000; cdc_enabled=false; cdc_free_space_chec
_interval_ms=250; cdc_raw_directory=null; cdc_total_space_in_mb=0; check_for_duplicate_rows_during_compaction=true; che
k_for_duplicate_rows_during_reads=true; client_encryption_options=<REDACTED>; cluster_name=Test Cluster; column_index_c
che_size_in_kb=2; column_index_size_in_kb=64; commit_failure_policy=stop; commitlog_compression=null; commitlog_directo
y=null; commitlog_max_compression_buffers_in_pool=3; commitlog_periodic_queue_size=-1; commitlog_segment_size_in_mb=32;
commitlog_sync=periodic; commitlog_sync_batch_window_in_ms=NaN; commitlog_sync_period_in_ms=10000; commitlog_total_spac
_in_mb=null; compaction_large_partition_warning_threshold_mb=100; compaction_throughput_mb_per_sec=16; concurrent_compa
tors=null; concurrent_counter_writes=32; concurrent_materialized_view_writes=32; concurrent_reads=32; concurrent_replic
tes=null; concurrent_writes=32; counter_cache_keys_to_save=2147483647; counter_cache_save_period=7200; counter_cache_si
e_in_mb=null; counter_write_request_timeout_in_ms=5000; credentials_cache_max_entries=1000; credentials_update_interval
```

Εικόνα 8: Εντοπισμός του 1ου σφάλματος

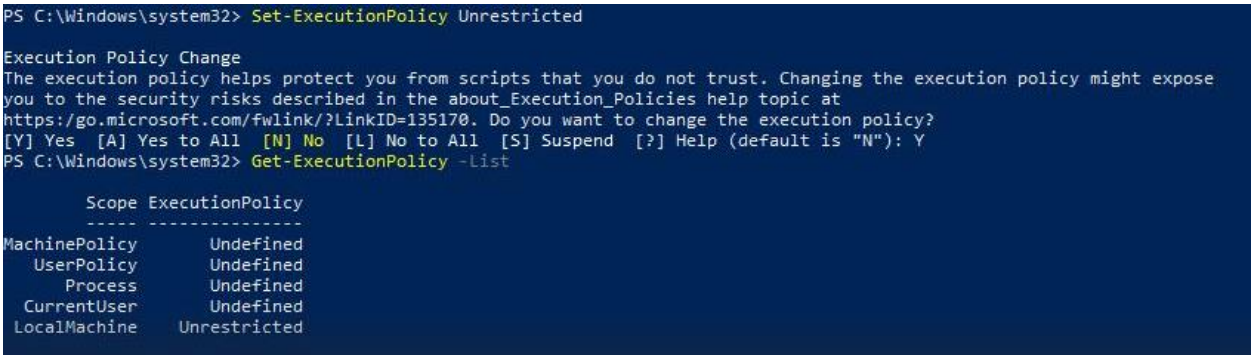
Εντολές επίλυσης:

a) ΣτοWindows PowerShell και εκτελώντας την εντολή Get-ExecutionPolicy -List , εμφανίζονται τα δικαιώματα των βασικών μεταβλητών του λειτουργικού συστήματοςWindows.
b) Τα δικαιώματα στην μεταβλητή LocalMachineδεν επιτρέπουν την δημιουργία κόμβων.



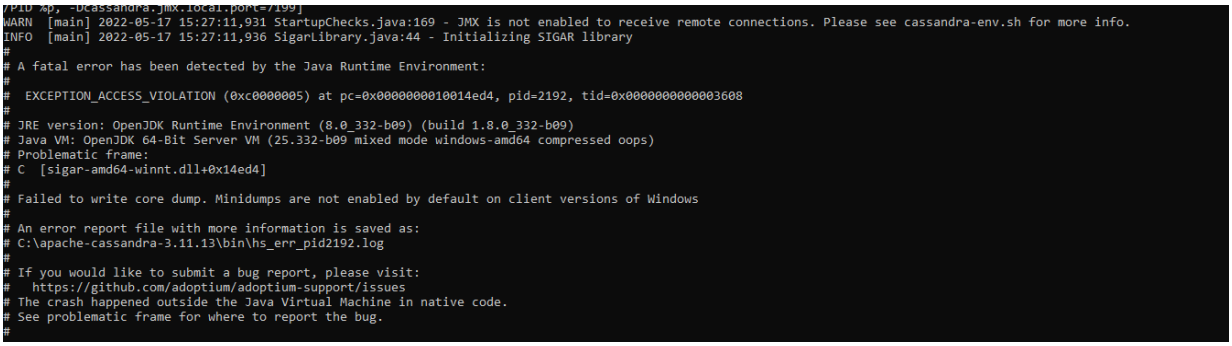
Εικόνα 9: Default δικαιώματα

c) Εκτελώντας την εντολή Set- ExecutionPolicyUnrestricted, τροποποιώντας τα δικαιώματα του υπολογιστή.



Εικόνα 10: Τροποποίηση δικαιώματος της μεταβλητής LocalMachine

2. Bypass SIGAR CHECK



Εικόνα 11: Εντοπισμός 2ου σφάλματος

Σφάλμα: sigar-amd64-winnt.dll+0x14ed4

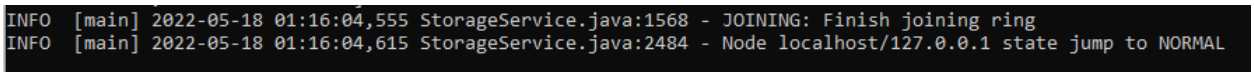
Το πακέτο sigar-amd64-winnt.dll δεν εμπεριέχεται στα βασικά πακέτα της Java 8 και είναι απαραίτητο η προσθήκη του για την λειτουργία του ApacheCassandra.

Επίλυση:

a. Αποθήκευση του συμπιεσμένου αρχείου hyperic-sigar https://sourceforge.net/projects/sigar/files/sigar/1.6/hyperic-sigar-1.6.4.zip/download
b. Στον φάκελο sigar-bin εντοπίζουμε το πακέτο sigar-amd64-winnt.dll και το μεταφέρουμε στον binφάκελο της Java.

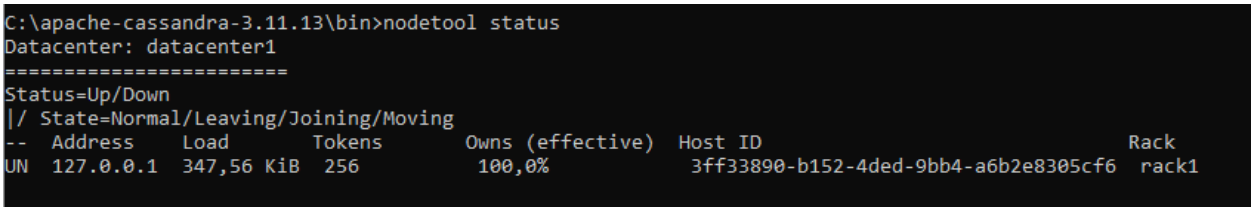
Βήμα 4:Έλεγχος εγκατάστασης

- Εκτέλεση της εντολής cassandra μέχρι να διορθωθούν όλα τα σφάλματα.
- Όταν εμφανισθεί το μήνυμα: <<Nodelocalhost/127.0.0.1 statejumptoNORMAL>>, τότε το εργαλείο έχει εγκατασταθεί επιτυχώς.



Εικόνα 12: Επιτυχής εγκατάσταση του ApacheCassandra

- Έλεγχος για ορθή επικοινωνία των κόμβων με την χρήση και δεύτερου CommandPrompt.
- Στο δεύτερο CommandPrompt πρέπει να εκτελεσθεί η εντολή nodetool status στο φάκελο C:\apache-cassandra-3.11.13\bin



Εικόνα 12:Επικοινωνία κόμβων

Κεφάλαιο 5: Apache Spark

Εικόνα14: Apache Spark

5.1: Τι είναι Apache Spark.

Το Apache Spark ανήκει στο ίδρυμα Apache και μπορεί να θεωρηθεί ως η λογική συνέχεια της μη σχεσιακής βάσης δεδομένων Apache Cassandra. Το Spark είναι πλατφόρμα ανοιχτού κώδικα, για κατανεμημένα συστήματα επεξεργασίας υψηλού φόρτου εργασίας Big Data. Μπορεί να εκτελεί ταχύτατα τις διεργασίες διαχείρισης δεδομένων, διανέμοντας τες στο δίκτυο εμβέλειας των διακομιστών του συστήματος. Η κατανομή των εργασιών πραγματοποιείται είτε αυτόματα στους υπολογιστές του είτε σε συνεργασία με άλλα κατανεμημένα συστήματα, όπως AWS. Οι λειτουργικές δυνατότητες του Spark, το κάνουν ιδανικό για την εφαρμογές Big Data και Machine Learning (Μηχανική Μάθηση), οι οποίες έχουν υψηλές και απαιτητικές ανάγκες για υπολογιστική ισχύ.

Το Apache Spark αποτελεί την βελτιωμένη εκδοχή του Hadoop MapReduce.Το Spark βασίζεται στην λειτουργία του MapReduce να διαχωρίζει τις εργασίες επεξεργασίας μεγάλων δεδομένων σε μικρότερες και τις διανέμει σε όλους τους κόμβους του κατανεμημένου συστήματος, μειώνοντας το συνολικό χρόνο εκτέλεσης. Επιπλέον χρησιμοποιεί την προσωρινή αποθήκευση στην μνήμη(Random Access Memory), ώστε να αποκρίνεται άμεσα σε συχνά ερωτήματα που δέχεται η βάση δεδομένων, τα οποία δεν απαιτούν μεγάλη υπολογιστική ισχύ. Το Spark είναι ταχύτερο σε σχέση με το Hadoop MapReduce, διότι έχει μειώσει την πολυπλοκότητα του προγραμματισμού με την χρήση των API. Πολλοί διαχειριστές NRDMS χαρακτηρίζουν το Spark ως πολυεργαλείο, καθώς μπορεί να εκτελέσει κατανεμημένα ερωτήματα με την γλώσσα SQL,να δημιουργήσει data pipelines, να απορροφήσει δεδομένα σε μια βάση δεδομένων, να εκτελέσει αλγόριθμους Μηχανικής Μάθησης και να κατασκευάσει γραφήματα και ροές δεδομένων.

Ιστορική αναδρομή

Το Apache Sparkδημιουργήθηκε το 2009 στο Ερευνητικό Εργαστήριο Αλγόριθμων, Μηχανών και Ανθρώπων (**AMPLAB**) του Πανεπιστημίου της Καλιφόρνιας στο Berkeley από τους Matei

Zaharia, Mosharaf Chowdhurry, Michael Franklin,Scott Shenker και Ion Stoica.Οι ερευνητές του εργαστηρίου παρατήρησαν ότι το εργαλείο Hadoop MapReduce αντιμετώπιζε δυσκολία στη διαχείριση επαναλαμβανόμενων υπολογιστικών διεργασιών, οι οποίες απαιτούν ταχεία επεξεργασία εφαρμόζοντας τεχνικές παράλληλου προγραμματισμού και κοινή δεδομένων χωρίς χρονική καθυστέρηση. Στις 3 Οκτωβρίου 2010 εκδόθηκε η 1^{η}έκδοση του Spark στο GitHub, η οποία ήταν δέκα φορές ταχύτερο από το MapReduceκαι οι πρώτοι χρήστες του ήταν ερευνητικά εργαστήρια του Πανεπιστημίου της Καλιφόρνιας.

Η πρώτη έκδοση του Spark υποστήριζε μόνο batch εφαρμογές, αλλά σύντομα αποσαφηνίστηκε η ουσιαστική χρήση του. Η διαδραστική αλληλεπίδραση χρήστη με τις βάσεις δεδομένων και τα ad-hoc ερωτήματα είναι τα κυριότερα χαρακτηριστικά που έκαναν το εργαλείο επιτυχημένο. Μετά από τις αρχικές εκδόσεις, έγινε σαφές ότι το εργαλείο θα επέκτεινε τις δυνατότητες του με την προσθήκη εξειδικευμένων βιβλιοθηκών, όπου οι Data Scientists θα μπορούν να επιλέξουν τα κατάλληλα plugins ανάλογα τις απαιτήσεις του κάθε συστήματος. Οι δημοφιλέστερες βιβλιοθήκες είναι MLlib για μηχανική μάθηση,Streaming για έλεγχο των ροών δεδομένων, Spark SQLκαι Graphx για επεξεργασία γραφημάτων.

5.2: Χαρακτηριστικά Apache Spark

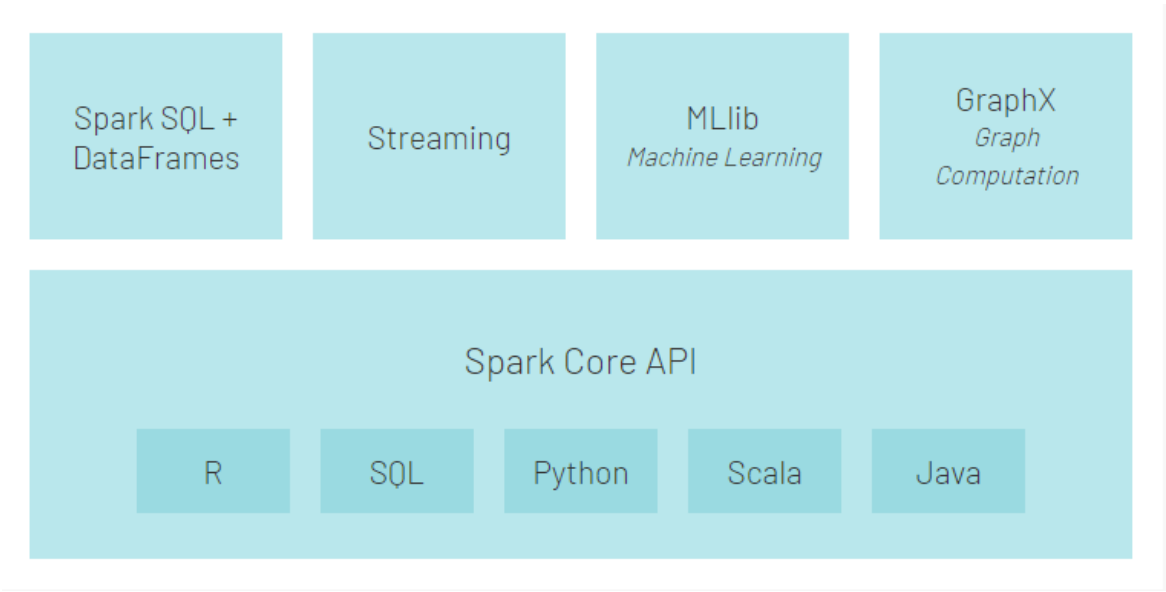
Ταχύτητα (Speed): Το Spark έχει σχεδιαστεί με σκοπό να αυξήσει την ταχύτητα εκτέλεσης υπολογιστικών εργασιών λειτουργώντας τόσο στη μνήμη όσο και στο δίσκο. Η εσωτερική του εφαρμογή επωφελείται την τεράστια πρόοδο της βιομηχανίας του hardware με την βελτίωση τιμής και απόδοσης των CPUκαι της μνήμης. Οι σύγχρονοι severs διαθέτουν εκατοντάδες πυρήνες και το λειτουργικό τους σύστημα βασίζεται στο Unix, εκμεταλλευόμενο την πολυνηματική (multithreading) και παράλληλη επεξεργασία με χαμηλό οικονομικό κόστος. Επιπλέον, το Spark αποκρίνεται στα ερωτήματα που δέχεται ο server με την δημιουργία υπολογιστικού κατευθυνόμενου ακυκλικού γραφήματος (DAG). Το γράφημα έχει σχεδιαστεί με πολλαπλά στάδια για αποτελεσματικότερη διανομή των εργασιών στους κόμβους, αναλύοντας τες που εκτελούνται στο σύμπλεγμα του συστήματος. Τέλος, η μηχανή φυσικής εκτέλεσης χρησιμοποιεί τη δημιουργία μαζικού και συμπαγούς κώδικα για την εκτέλεση[30].

Αρθρωτότητα (Modularity): Το Spark επιτυγχάνει την απλότητα αφαιρώντας την πολυπλοκότητα του προγραμματισμού και παρέχοντας μια θεμελιώδη και απλή λογική δομής δεδομένων. Στο Ανθεκτικό Κατανεμημένο Σύνολο Δεδομένων (**R**esilient **D**istributed **D**ataset) βασίζονται όλες οι άλλες αποδομήσεις δομημένων δεδομένων υψηλότερου επιπέδου, όπως Data Frames και Datasets. Επιπλέον, το Spark παρέχει ένα σύνολο μετασχηματισμών και ενεργειών ως λειτουργίες και ένα απλοποιημένο μοντέλο προγραμματισμού, που μπορεί να χρησιμοποιείται για την δημιουργία εφαρμογών Big Data στις υποστηριζόμενες γλώσσες: SQL, Java, Python,R και Scala. Όλες οι δυνατότητες του είναι προσβάσιμες Spark μέσω API. Τα API είναι τεκμηριωμένα

και δομημένα με περίτεχνο τρόπο καθιστώντας την αλληλεπίδραση των Data Scientists και τους developers των εφαρμογών με τις NRDBMS εύχρηστη. Επιστήμονες των δεδομένων και προγραμματιστές χρησιμοποιούν στο μέγιστο των δυνατοτήτων τα κλιμακούμενα εργαλεία και την αξιόπιστη απόδοση και ταχύτητα χωρίς πολλές αναζητήσεις και λεπτομέρειες.

5.3: Spark Ecosystem

Το προσφέρει ενοποιημένες βιβλιοθήκες με καλά τεκμηριωμένα API που περιλαμβάνουν τις ακόλουθες ενότητες ως βασικά στοιχεία: SparkSQL, Spark Structured Streaming, Spark MLlib και Graphx, συνδυάζοντας όλους τους φόρτους εργασίας που εκτελούνται σε έναν διακομιστή.



Εικόνα 13:Apache Spark Ecosystem

Spark Core: Όλες οι λειτουργίες του εργαλείου Spark βασίζονται στο Spark Core, καθιστώντας το ως το θεμελιώδες στοιχείο για παράλληλη και κατανεμημένη επεξεργασία των μεγάλων δεδομένων. Το Core είναι υπεύθυνο για όλες τις βασικές λειτουργίες Εισόδων/Εξόδων (I/O), τον προγραμματισμό και την παρακολούθηση των εργασιών στο σύμπλεγμα κόμβων του συστήματος. Επίσης, διατηρεί όλα τα στοιχεία που συσχετίζονται με το συγχρονισμό των διεργασιών, την δικτύωση με τα διάφορα συστήματα αποθήκευσης που συνεργάζεται, την ανάκτηση σφαλμάτων και την αποτελεσματική διαχείριση μνήμης. Τέλος, το Spark Core αξιοποιεί μια ειδική ανθεκτική δομή δεδομένων **R**esilient **D**istributed **D**atasets, τα οποία επαναχρησιμοποιούν τα δεδομένα κατανεμημένων υπολογιστικών συστημάτων. Τα RDDείναι ανθεκτικές/αμετάβλητες και κατακερματισμένες συλλογές εγγράφων και μπορούν να περιέχουν οποιοδήποτε γλώσσα αντικειμενοστραφή προγραμματισμού (Python,Java,Scala) ή και αντικείμενα κλάσης που ορίζονται από τον χρήστη ενός συστήματος. Τα RDD υλοποιούνται είτε με τον μετασχηματισμό των υπαρχόντων RDD είτε με τη μεταφόρτωση ενός εξωτερικού dataset από μια σταθερή αποθήκευση όπως HBase, HDFS.

Spark SQL(Shark): Το Spark SQL αποτελεί το κατανεμημένο πλαίσιο για δομημένη επεξεργασία δεδομένων, βασιζόμενο στον πυρήνα του Spark. Οι προγραμματιστές του Shark έχουν την δυνατότητα να αξιοποιήσουν την ισχύ των δηλωτικών ερωτημάτων και τη βελτιστοποιημένη αποθήκευση εκτελώντας ερωτήματα τύπου SQL, τα οποία υπάρχουν σε RDD και σε άλλες εξωτερικές πηγές, χωρίς να εξαρτάται από το API ή την γλώσσα που χρησιμοποιείται για εκτέλεση υπολογιστικών εργασιών. Επίσης, επιτρέπει την διαδραστική και αναλυτική εφαρμογή τόσο σε data pipelines όσο και σε ιστορικά δεδομένα. Οι διαδραστικές εφαρμογές εξυπηρετούν τους χρήστες του συστήματος, παρέχοντας την δυνατότητα εκτέλεσης, εξαγωγής, μετασχηματισμού και φόρτωσης λειτουργιών στα δεδομένα, τα οποία προέρχονται από αρχεία JSON και σε συνεχεία να διεξάγουν ad-hoc ερωτήματα. Τέλος, το Shark διευκολύνει τη διαδικασία εξαγωγής και σύμπτυξης διαφόρων datasets, ώστε να είναι έτοιμα για εφαρμογή μηχανικής μάθησης.

Catalyst: Από την πρώτη έκδοση του Apache Spark, μέλος του βασικού στοιχείου του Spark SQL αποτελεί το Catalyst. Το Catalyst είναι ένα πλαίσιο, το οποίο βελτιστοποιεί την ενσωματωμένη γλώσσα Scala, αυξάνοντας την απόδοση των δηλωτικών ερωτημάτων που γράφουν οι προγραμματιστές των Big Data συστημάτων. Με μεγάλη ευκολία επιτυγχάνεται ο καθορισμός σύνθετων σχεσιακών βελτιστοποιήσεων και ο μετασχηματισμός ερωτημάτων, αξιοποιώντας ισχυρές δομές προγραμματισμού. Τέλος, το Catalyst διευκολύνει την προσθήκη νέων κανόνων βελτιστοποίησης πηγών και τύπων δεδομένων, καθώς παρατηρείται ότι τα συστήματα μεγάλων δεδομένων αυξάνουν με ραγδαία ταχύτητα την εδραίωση τους.

Data Frame: Στην έκδοση 1.6.3 του Apache Spark (7Νοεμβρίου 2016), αφαιρέθηκε το Data Frame από Spark SQL, δημιουργώντας ένα νέο στοιχείο στο οικοσύστημα του εργαλείου. Στις παλαιότερες εκδόσεις αυτό το στοιχείο περιλαμβανόταν στο Schema του RDD. Πλέον το Data Frame API αποτελεί μια ξεχωριστή κατανεμημένη συλλογή δεδομένων, τα οποία ταξινομούνται σε στήλες και ενσωματώνονται με διαδικαστικό κώδικα και σχεσιακή επεξεργασία. Οι λειτουργίες αξιολογούνται με επιεική κριτήρια, παρέχοντας υποστήριξη για σχεσιακές βελτιώσεις και βελτιστοποίηση της συνολικής ροής εργασιών επεξεργασίας δεδομένων.

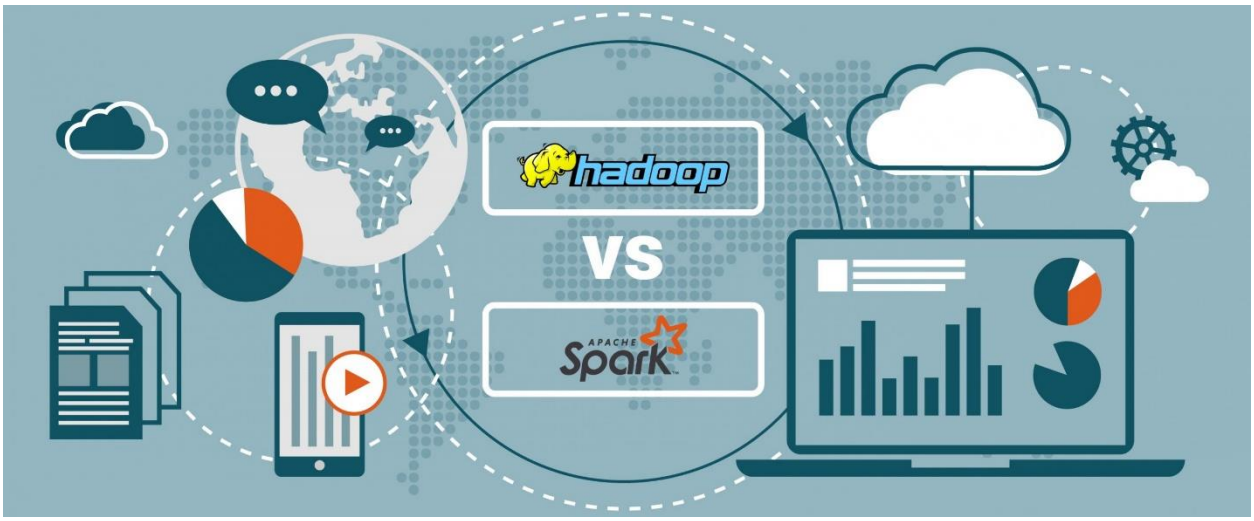
Spark Streaming: Από τα αρχεία καταγραφής έως τα δεδομένα αισθητήρων, οι προγραμματιστές των εφαρμογών αναγκάζονται να αντιμετωπίσουν ολοένα και περισσότερο το πρόβλημα διαχείρισης πραγματικού χρόνου ροής δεδομένων (data pipelines). Το Spark Streaming επιτυγχάνει την υψηλή απόδοση και την ανοχή σε σφάλματα κατά την διαδικασία ζωντανών ροών δεδομένων. Το Spark λειτουργεί αξιοποιώντας πολλούς και διαφορετικούς αλγόριθμους, οι οποίοι λαμβάνουν τα δεδομένα σε ένα σύστημα αρχείων, μια βάση δεδομένων και σε πίνακα ελέγχου πραγματικού χρόνου. Η βασικότερη τεχνική που χρησιμοποιεί τοSpark Streaming είναι το Micro-batching, η οποία επιτρέπει σε μια διεργασία να αντιμετωπίσει έναdata pipeline ως ακολουθία

μικρών πακέτων πληροφοριών. Ως εκ τούτου, είναι κατανοητό ότι το Spark Streaming λειτουργεί σε 3 φάσεις. Αρχικά, **συγκεντρώνει** όλες τις πηγές ροών δεδομένων από τα διασυνδεδεμένα API και τις υποδοχές TCP. Έπειτα τα συγκεντρωμένα δεδομένα **επεξεργάζονται** χρησιμοποιώντας πολύπλοκους αλγόριθμος εκφράζοντας συναρτήσεις υψηλού επιπέδου και πολύπλοκους μετασχηματισμούς. Τέλος, τα επεξεργασμένα δεδομένα προωθούνται και **αποθηκεύονται** σε βάσεις δεδομένων.

Spark MLlib: Όπως έχει προαναφερθεί, ο όγκος των δεδομένων αυξάνεται συνεχώς, και οι αλγόριθμοι μηχανικής μάθησης παρουσιάζουν ευκολία χρήσης σε ένα τεράστιο εύρος συστημάτων και τα αποτελέσματα τους αναδύουν μεγάλη ακρίβεια, προβάλλοντας την ανάγκη για την ύπαρξη βιβλιοθηκών ML. Το project Apache Spark εντόπισε αυτή την ανάγκη και πρόσθεσε στο βασικό οικοσύστημα του την βιβλιοθήκη MLlib. Το Spark MLlib αποτελεί μια επεκτάσιμη βιβλιοθήκη μηχανικής μάθησης, η οποία αναφέρεται στους αλγορίθμους υψηλής ποιότητας και ταχύτητας. Περιέχει βιβλιοθήκες ML που έχουν εφαρμογή σε διαφόρους αλγορίθμους, όπως ομαδοποίηση, ταξινόμηση και ταυτόχρονος καθαρισμός δεδομένων. Παρόλα αυτά το MLlib διατηρεί τους αρχέγονους χαμηλού επιπέδου αλγορίθμους, για λόγους συμβατότητας με τα παλαιά συστήματα και τις αναχρονισμένες τεχνικές μηχανικής μάθησης. Το χρονολογικό ορόσημο για την MLlib, αποτελεί η έκδοση 2.2.0 (17 Ιουλίου 2017) του Spark. Στην έκδοση 2.0,0 (26 Ιουλίου 2016) το API βασιζόταν σε RDD στο πακέτο spark.mllib εισήχθη σε λειτουργία συντήρησης. Αντίθετα από την έκδοση 2.2.0 και έπειτα το API της MLlib βασίζεται στο Data Frame, κάνοντας την φιλικότερη προς το χρήστη σχέση με το RDD. Τέλος, η MLlib χρησιμοποιεί το πακέτο γραμμικής άλγεβρας Breeze, αυξάνοντας την ταχύτητα αριθμητικών υπολογισμών.

Spark Graphx (Graph Computation): Η πιο πρόσφατη βιβλιοθήκη που προστέθηκε στο οικοσύστημα του Spark αποτελεί το Graphx (έκδοση 3.0.0, 18 Ιουνίου 2020). Το Graphx είναι API για παράλληλη εκτέλεση και αναπαράσταση υπολογιστικών αποτελεσμάτων σε γραφήματα. Ταυτόχρονα, αυτή η βιβλιοθήκη παρέχει την δυνατότητα για ομαδοποίηση, ανάλυση και εύρεση της βέλτιστης διαδρομής αναζήτησης γραφήματος, καθώς πραγματοποιεί λεπτομερής ανάλυση του δικτύου στα οποία αποθηκεύονται τα δεδομένα. Επιπλέον το Spark Graphx συμβάλει στην ανάκτηση πολύτιμων πληροφοριών από τα ασυνεπή δεδομένα, αξιοποιώντας την διαδικασία ELT και μειώνοντας τον χρόνο και το κόστος ανάλυσης δεδομένων. Καθώς το Spark είναι ικανό να αποθηκεύει πληροφορίες στη μνήμη και μπορεί να εκτελεί συνεχόμενα ερωτήματα γρήγορα, καθιστά εύκολο τον εντοπισμό των αλγορίθμων μηχανικής μάθησης που μπορούν να επαναχρησιμοποιηθούν για ένα συγκεκριμένο είδος δεδομένων. Τέλος, η βιβλιοθήκη Graphx αποτελεί το ιδανικότερο εργαλείο για αναπαράσταση δεδομένων πραγματικού χρόνου, αξιοποιώντας το Spark Streaming.

5.4: Hadoop Vs Spark



Εικόνα 16:Hadoop VS Spark

Το Apache Spark και Hadoop αποτελούν τα δύο από τα πιο σημαντικά κατανεμημένα συστήματα επεξεργασίας δεδομένων. Αμφότερα ανήκουν στα project του ομίλου Apache και συχνά χρησιμοποιούνται ταυτόχρονα, καθώς παρομοιάζουν αρκετές ομοιότητες. Παρόλες τις ομοιότητες κρίνεται απαραίτητο να κατανοηθούν τα χαρακτηριστικά και οι μέγιστες δυνατότητες που προσφέρει καθένα από τα εργαλεία.

Αρχιτεκτονική: Στο Hadoop, όλα τα δεδομένα χωρίζονται σε block που αναπαράγονται στις μονάδες δίσκου των διαφόρων servers σε ένα cluster, με το HDFS (Hadoop Distributed File System) να παρέχει υψηλά επίπεδα πλεονασμού και ανοχής σφαλμάτων. Οι εφαρμογές του Hadoop μπορούν στη συνέχεια να εκτελεστούν ως μία εργασία ή ως κατευθυνόμενο άκυκλο γράφημα που περιέχει πολλαπλές εργασίες. Στην πρώτη έκδοση του Hadoop (1 Απριλίου 2006), μια κεντρική υπηρεσία Job Tracker κατένειμε εργασίες σε κόμβους που θα μπορούσαν να εκτελούνται ανεξάρτητα ο ένας από τον άλλο και μια τοπική υπηρεσία Task Tracker διαχειριζόταν την εκτέλεση εργασιών από μεμονωμένους κόμβους. Από την έκδοση 2.0.0 (23 Μαΐου 2012) αυτές οι δύο υπηρεσίες αντικαταστάθηκαν από το υποσύστημα YARN. Το YARN περιέχει τρεις υπηρεσίες διαχείρισης διεργασιών Resource Manager, Node Manager και Application Managerοι οποίες λειτουργούν στο υπόβαθρο όλων των συστημάτων Hadoop.Ο Resource Manager λειτουργεί ως παγκόσμιος χρονοπρογραμματιστής εργασιών και κατανέμει τους διαθέσιμους υπολογιστικούς πόρους. Ο Node Manager είναι εγκατεστημένος σε κάθε κόμβο του συμπλέγματος, παρακολουθώντας την χρήση πόρων. Ο Application Manager υλοποιεί για κάθε εφαρμογή που διαπραγματεύεται τους πόρους από τον Resource Manager και συνεργάζεται με τον Node Manager για την εκτέλεση εργασιών επεξεργασίας δεδομένων. Ταυτόχρονα παρέχεται μίαabstract πηγή πόρων, όπου συγκρατούνται όλες οι πληροφορίες σχετικά με τους πόρους που έχουν εκχωρηθεί σε διαφορετικούς κόμβους και εφαρμογές. Όπως αναφέρθηκε και στην αρχή το Spark αποτελεί την βελτιωμένη έκδοση του Hadoop.Η θεμελιώδης διαφορά μεταξύ Hadoop και

Spark σχετίζεται με τον τρόπο οργάνωσης των δεδομένων για επεξεργασία. Στο Spark, η πρόσβαση στα δεδομένα υλοποιείται από εξωτερικά αποθετήρια αποθήκευσης όπως HDFS, cloud ή διάφορες databases και άλλους τύπους data stores. Παρόλο που οι περισσότερες επεξεργασίες γίνονται στη μνήμη, η πλατφόρμα μπορεί να διαμοιράσει δεδομένα στον δίσκο αξιοποιώντας όλες τις δυνατότητες του cluster.

Κλιμάκωση (Scalability): Τα συστήματα Hadoop μπορούν να κλιμακωθούν για να φιλοξενήσουν μεγαλύτερα σύνολα δεδομένων στα οποία γίνεται σταδιακή πρόσβαση, επειδή τα δεδομένα μπορούν να αποθηκευτούν και να υποβληθούν για επεξεργασία με την μικρή χρήση του δίσκου σε σχέση με την μνήμη. Το YARN επιτρέπει στα clusters (συμπλέγματα) να υποστηρίζουν δεκάδες χιλιάδες κόμβους συνδέοντας πολλά υποσυμπλέγματα που έχουν τους δικούς τους διαχειριστές πόρων. Απαραίτητη επένδυση για την επέκταση συστημάτων Hadoop αποτελεί το εργατικό δυναμικό, καθώς είναι αναγκαία η εγκατάσταση εσωτερικών εφαρμογών για την παροχή νέων κόμβων και την προσθήκη τους σε ένα σύμπλεγμα. Επιπρόσθετα, με το Hadoop, η αποθήκευση συγκεντρώνεται με υπολογιστικούς πόρους στους κόμβους των συμπλεγμάτων, γεγονός που μπορεί να δυσκολέψει την λειτουργία των εφαρμογών και των χρηστών εκτός του cluster να έχουν πρόσβαση στα δεδομένα. Παρόλα αυτά κάποια από τα ζητήματα επεκτασιμότητας μπορούν να επιλυθούν με τις υπηρεσίες Hadoop Cloud. Εν αντιθέσει, στο Spark ο χώρος αποθήκευσης και οι υπολογιστικές διαδικασίες διαχωρίζονται, γεγονός που μπορεί να διευκολύνει τις εφαρμογές και τους χρήστες να έχουν πρόσβαση στα δεδομένα από οπουδήποτε. Επιπλέον, το Spark περιλαμβάνει εργαλεία που μπορούν να βοηθήσουν του χρήστες να κλιμακώσουν δυναμικά τους κόμβους ανάλογα με τις απαιτήσεις των διεργασιών. Στις περισσότερες περιπτώσεις δεν χρειάζεται επένδυση σε εργατικό δυναμικό, για ανακατανομή κόμβων στα συμπλέγματα, καθώς για τα συστήματα του Spark αυτή η διαδικασία είναι αυτοματοποιημένη. Ωστόσο, για την κλιμάκωση των εφαρμογών στο Spark πρέπει να διασφαλιστεί ο διαμοιρασμός του φόρτου εργασίας μεταξύ των κόμβων για να μειωθεί ο διασκορπισμός μνήμης.

Ασφάλεια(Security): Το Hadoop παρέχει υψηλότερο επίπεδο ασφαλείας με λιγότερα εξωτερικά έξοδα για μακροπρόθεσμη διατήρηση δεδομένων. Το HDFS προσφέρει ολοκληρωμένη κρυπτογράφηση με ξεχωριστές ζώνες κρυπτογράφησης και ενσωματωμένη υπηρεσία διαχείρισης των κλειδιών κρυπτογράφησης. Επιπλέον, περιλαμβάνει ένα μοντέλο που βασίζεται σε δικαιώματα για την επιβολή ελέγχων πρόσβασης για αρχεία και καταλόγους με τη δυνατότητα δημιουργίας λιστών ελέγχου πρόσβασης που μπορούν να χρησιμοποιηθούν για την εφαρμογή ασφαλείας βάσει ρόλων και άλλων τύπων κανόνων για διαφορετικούς χρήστες ή ομάδες. Μπορεί επίσης να εκμεταλλευτεί τα σχετικά εργαλεία όπως το Apache Knox, μια πύλη που παρέχει RESTAPI υπηρεσίες ελέγχου ταυτότητας και διακομιστή μεσολάβησης για την επιβολή πολιτικών ασφαλείας στα συμπλέγματα Hadoop και το Apache Ranger, ένα κεντρικό πλαίσιο διαχείρισης ασφαλείας για περιβάλλοντα Hadoop[45]. Αντιθέτως, το Spark διαθέτει ένα πιο περίπλοκο μοντέλο ασφαλείας που υποστηρίζει διαφορετικά επίπεδα ασφαλείας για διαφορετικούς τύπους ανάπτυξης. Χρησιμοποιεί μια κοινή μυστική προσέγγιση ελέγχου ταυτότητας για κλήσεις

απομακρυσμένων διαδικασιών μεταξύ διεργασιών του Spark, με ειδικούς μηχανισμούς ανάπτυξης για την δημιουργία μυστικών κωδικών πρόσβασης. Σε ορισμένες περιπτώσεις, οι προστασίες ασφαλείας είναι περιορισμένες επειδή όλες οι εφαρμογές διαμοιράζονται κοινά απόρρητα αρχεία. Το μοντέλο Spark κατασκευάστηκε κυρίως για να επιβάλλει την ασφάλεια άνω του επιπέδου των ροών δεδομένων, οι οποίες είναι λιγότερο μόνιμες από τα δεδομένα που αποθηκεύονται για μεγάλες περιόδους, δημιουργώντας ανησυχία για ευρείας κλίμακας κυβερνοεπιθέσεις στις υποδομές του. Ο έλεγχος ταυτότητας και άλλα μέτρα ασφαλείας δεν είναι ενεργοποιημένα από προεπιλογή στο Spark. Ως εκ τούτου, με τον κατάλληλο συνδυασμό αλγορίθμου κρυπτογράφησης με πολιτικές διαχείρισης κλειδιών μπορεί να επιτευχθούν επαρκείς ζώνες προστασίας δεδομένων.

5.5: Τι δεν προσφέρει;

Το Apache Spark αποτελεί ένα από τα βασικότερα εργαλεία στην διαχείριση Μεγάλων Δεδομένων που χρησιμοποιείται ευρέως από τις βιομηχανίες, αλλά εκτός από το πλήθος των δυνατοτήτων όπως προαναφέρθηκαν, έχει ορισμένα μειονεκτήματα, όπως έλλειψη παροχής υποστήριξης σε πραγματικό χρόνο, διαχείριση αρχείων μικρού μεγέθους, κόστος κ.ά..

Έλλειψη υποστήριξης για επεξεργασία δεδομένων σε πραγματικό χρόνο: Το Spark δεν υποστηρίζει πλήρως την επεξεργασία ροής δεδομένων σε πραγματικό χρόνο. Στο Spark Streaming, η εισερχόμενη ροή δεδομένων πραγματικού χρόνου διαμοιράζεται σε προκαθορισμένα διαστήματα και σε ειδικές ανθεκτικές δομές δεδομένων. Τα RDD υποβάλλονται σε επεξεργασία χρησιμοποιώντας λειτουργίες όπως σύνδεση DB, MapReduce. Το αποτέλεσμα αυτών των διεργασιών επιστρέφεται σε Micro-Batch. Επομένως, δεν πρόκειται για επεξεργασία σε πραγματικό χρόνο, αλλά το Spark επεξεργάζεται δεδομένα **σχεδόν** σε πραγματικό χρόνο.

Διαχείριση αρχείων μικρού μεγέθους: Το Spark βασίζεται σε άλλες πλατφόρμες όπως το Hadoopή άλλη Cloud-based εφαρμογή για την διαχείριση αρχείων. Ακόμα και με τον συνδυασμό Hadoop και Spark διαπιστώνεται ότι υπάρχει πρόβλημα με την διαχείριση αρχείων μικρού μεγέθους. Το HDFS (**H**adoop **D**istributed **F**ile **S**ystem) παρέχει περιορισμένο αριθμό μεγάλου μεγέθους αρχείων αντί για μεγάλο αριθμό μικρού όγκου αρχείων. Επιπλέον, όταν στο σύστημα διαθέτει πολλά μικρά gzip δεδομένα, το Spark τα διατηρεί στο δίκτυο και τα αποσυμπιέζει με την προϋπόθεση ότι ολόκληρο το αρχείο βρίσκεται στο Spark Core. Δαπανιέται μεγάλο χρονικό διάστημα για την εγγραφή και αποσυμπίεση τους στο Core. Στο RDD κάθε αρχείο που προκύπτει, διαιρείται σε ένα μεγάλος πλήθος μικρότερων για αποτελεσματικότερη διαχείριση, απαιτώντας εκτενή αναζήτηση μέσα στους κόμβους του συστήματος.

Οικονομική επιβάρυνση: Το Spark όπως προαναφέρθηκε επεξεργάζεται τα δεδομένα απευθείας στην μνήμη (In-Memory), αλλά κάποιες φορές αποτελεί πρόβλημα. Για την In-Memory

επεξεργασία απαιτείται μεγάλη οικονομική επένδυση. Η κατανάλωση μνήμης δεν αντιμετωπίζεται με user-friendly τρόπο. Το Spark απαιτεί πολλή μνήμη RAM, για να λειτουργεί στην μνήμη αυξάνοντας δραματικά το κόστος ενός συστήματος.

Έλλειψη αυτόματων βελτιώσεων: Δυστυχώς, το Apache Spark δεν διαθέτει αυτοματοποιημένη βελτίωση κώδικα. Η χειροκίνητη βελτιστοποίηση είναι επαρκής για συγκεκριμένα σύνολα δεδομένων. Ουσιαστικά, ο προγραμματιστής ορίζει ξεχωριστά κάθε partition, προσθέτοντας ως δεύτερη παράμετρο την μέθοδο της παραλληλοποίησης. Σε κάποιες περιπτώσεις είναι επιθυμητή η χειροκίνητη βελτιστοποίηση για ορθή κατάτμηση και προσωρινή αποθήκευση στο Spark.

5.6: Εγκατάσταση σε Windows 10

Το Apache Spark καθώς ανήκει στην κατηγορία των open-source projects, προϋποθέτει η λειτουργία του να πραγματοποιείται σε ελεύθερο λογισμικό Linux. Το Linux και τα Windows βασίζονται στο Unix. Ωστόσο, για την αξιοποίηση των χαρακτηριστικών των Linux στα Windows απαιτείται η ενεργοποίηση του υποσυστήματος WSL (Windows Subsystem for Linux). Το WSL βοηθά στην ενσωμάτωση και στην εκτέλεση εφαρμογών Linux εγγενώς στα Windows, παρέχοντας τη δυνατότητα άμεσης αλληλεπίδρασης, χωρίς να καταναλώνονται επιπλέον πόροι όπως θα γινόταν με την χρήση Virtual Machine. Εκτός από το WSL, χρειάζεται η εγκατάσταση εφαρμογής UBUNTU 20.04 LTS η οποία είναι το περιβάλλον διαχείρισης του Spark. Τέλος, καθώς το Spark βασίζεται στη τεχνολογία του Hadoop, στο εικονικό περιβάλλον που θα αναπτυχθεί υποχρεωτικά εμπεριέχονται όλα τα δομικά στοιχεία του.

Βήμα 1:

Στην εφαρμογή UBUNTU 20.04 LTS πρέπει να εγκατασταθεί το περιβάλλον Hadoop με την χρήση των εντολών[49].

tar –xvzf auto_hadoopuser_WSL2.tar.gz

rm –rfauto_hadoopuser_WSL2.tar.gz

/home/hadoopuser/hadoop/myScripts/install-cluster2.sh

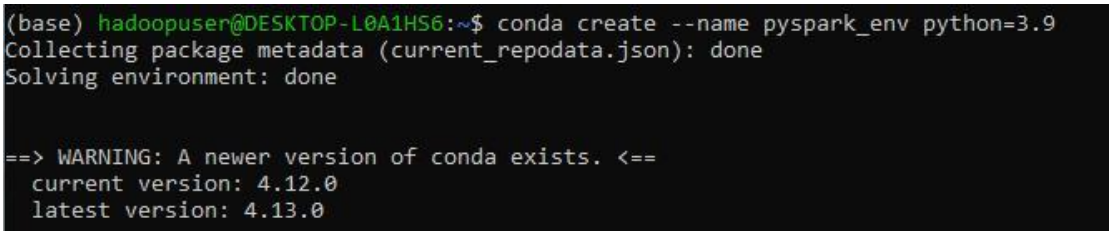
```
(base) hadoopuser@DESKTOP-L0A1HS6:~$ /home/hadoopuser/hadoop/myScripts/install-cluster.sh
============================
Δημιουργία Περιβάλλοντος.
============================
------------------------------
Δημιουργία Group hadoopgroup.
------------------------------
[sudo] password for hadoopuser:
```

Εικόνα 17: Δημιουργία περιβάλλοντοςHadoop

Βήμα 2:

Το Spark Core API αξιοποιεί όλες τις γλώσσες αντικειμενοστραφή προγραμματισμού (Java, Python). Για την εκτέλεση προγραμμάτων γίνεται η επιλογή της γλώσσας Python.

condacreate –namepyspark_envpython=3.9



Εικόνα 18: Δημιουργία περιβάλλοντος Python

Βήμα 3:

Για την ολοκληρωμένη χρήση του Pseudo Cluster που έχει δημιουργηθεί χρειάζεται να γίνει αλλαγή των δικαιωμάτων του χρήστημε την εντολή:

sudochown –R hadoopuser:hadoopgroup /home/hadoopuser/hadoop

Βήμα 4:

Ρύθμιση παραμέτρων, όπου αναφέρεται στη Java, Python, Hadoop, Yarn.

nano ~/.bashrc

```
# <<< conda initialize <<<

#------------------------------------------ SETTINGS -------------------------------------
# >>> conda activate conda env pyspark_env >>>
conda activate pyspark_env
# <<< conda activate conda env pyspark_env <<<
#------------------------------------------ JAVA -----------------------------------------
#-----------------------------------------------------------------------------------------
export JAVA_HOME='/usr/lib/jvm/java-8-openjdk-amd64'
export JRE_HOME='/usr/lib/jvm/java-8-openjdk-amd64/jre'
#------------------------------------------ HADOOP ---------------------------------------
#-----------------------------------------------------------------------------------------
export
HADOOP_CLASSPATH='/usr/lib/jvm/java-8-openjdk-amd64/lib/tools.jar:/home/hadoopuser/spark/yarn/spark-3.1.3-ya
rn-shuffle.jar'
export HADOOP_HOME='/home/hadoopuser/hadoop'
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$HADOOP_HOME/myScripts
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/hadoopuser/hadoop/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
#------------------------------------------ SPARK ----------------------------------------
#-----------------------------------------------------------------------------------------
export PYSPARK_PYTHON='/home/hadoopuser/miniconda3/envs/pyspark_env/bin/python3'
export PYSPARK_DRIVER_PYTHON='/home/hadoopuser/miniconda3/envs/pyspark_env/bin/python3'
export SPARK_HOME='/home/hadoopuser/spark'
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin:$SPARK_HOME/jars:/etc/default
export SPARK_CLASSPATH=$JAVA_HOME/lib/tools.jar:$SPARK_HOME/yarn/spark-3.1.3-yarn-shuffle.jar
export PYTHONPATH=$SPARK_HOME/python:$SPARK_HOME/python/lib/py4j-0.10.9-src.zip:$PYTHONPATH
export HBASE_OPTS="$HBASE_OPTS --add-opens java.base/sun.nio.ch=ALL-UNNAMED"
export SPARK_OPTS="--packages graphframes:graphframes:0.8.2-spark3.1-s_2.12"
#------------------------------------ Settings for Jupyter -------------------------------
alias jupyter-notebook='~/miniconda3/envs/pyspark_env/bin/jupyter-notebook --no-browser'
#------------------------------------ END SETTINGS ---------------------------------------
```

Εικόνα 19: Προσθήκη παραμέτρων στο bashrc

Βήμα 5:

Το WSL αντιλαμβάνεται σαν master κόμβο τον υπολογιστή που χρησιμοποιείται, ο οποίος δεν διαθέτει άλλους slaves κόμβους για την εκτέλεση των εφαρμογών Spark. Με την προσθήκη του αρχείου wsl.conf, ορίζεται ως hostname=master και δεν πραγματοποιείται αυτόματη δημιουργία hosts.

sudonano /etc/wsl.conf

```
# This file was automatically generated by WSL. To stop automatic generation of this file, add the following
entry to /etc/wsl.conf:
[network]
hostname = master
generateHosts = false
```

Εικόνα 20: Παράμετροι στο wsl.conf

Βήμα 6:

Τροποποίηση αρχείου host

sudo nano /etc/hosts

```
# This file was automatically generated by WSL. To stop automatic generation of this file, add the following
entry to /etc/wsl.conf:
# [network]
# generateHosts = false
127.0.1.1 master
0.0.0.0 worker
# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Εικόνα 21: Παράμετροι στο host

Επεξήγηση:

- 127.0.1.1 master : ο τοπικός κόμβος (υπολογιστής δημιουργίας του εικονικού περιβάλλοντος)ορίζεται ως master του περιβάλλοντος
- 0.0.0.0 worker : ο απομακρυσμένος κόμβος, ο οποίος εκτελεί όλες τις διεργασίες

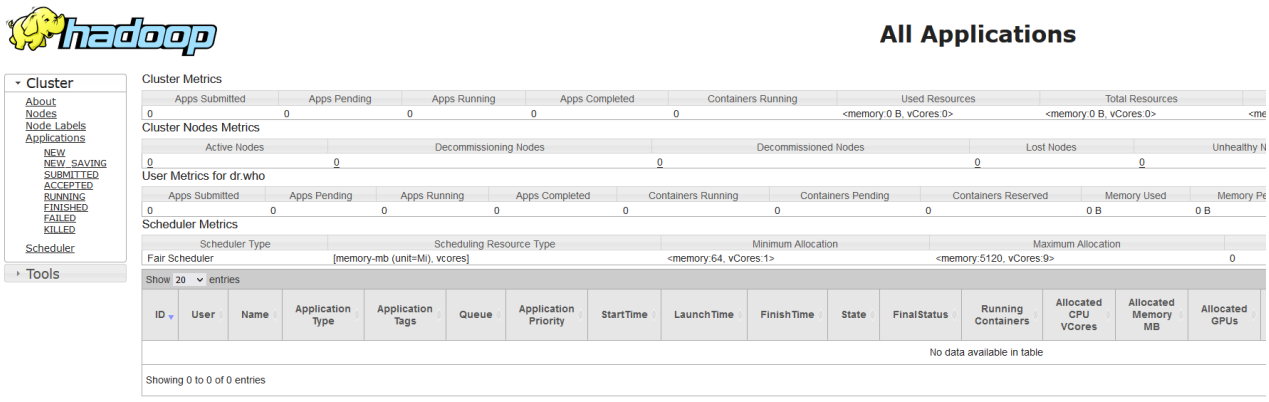
Βήμα 7:

Δημιουργία cluster με την εντολή

create-cluster.sh

Τέλος το cluster έχει δημιουργηθεί και παρέχεται η δυνατότητα ελέγχου της κατάστασής του

http://localhost:8088/cluster



Εικόνα 22: Κέντρο ελέγχου cluster

Κεφάλαιο 6: Ανάλυση δεδομένων υγειονομικής περίθαλψης

Σε όλη την περίοδο ύπαρξης της ανθρωπότητας, βασικό μέλημα των ανθρώπου είναι η διατήρηση καλής υγείας. Η εξέλιξη της τεχνολογίας βοήθησε στην ακριβέστερη διάγνωση ασθενειών. Στην εποχή των big data, καθημερινά παράγεται ένας τεράστιος όγκος ετερογενών υγειονομικών δεδομένων. Τα υγειονομικά δεδομένα, είναι απαραίτητο να ενσωματώνονται και να αποθηκεύονται με τέτοιο τρόπο, ώστε να διευκολύνουν την αποδοτική ανάλυση τους. Οι σχεσιακές βάσεις δεδομένων δεν είναι επαρκής για την διαχείριση αυτών των ετερογενών και συνεχώς αυξανόμενων δεδομένων. Οι NoSQL βάσεις δεδομένων εγγυώνται την βέλτιστη ταξινόμηση των ιατρικών δεδομένων.

Εικόνα 23: Big Data Analytics in Healthcare

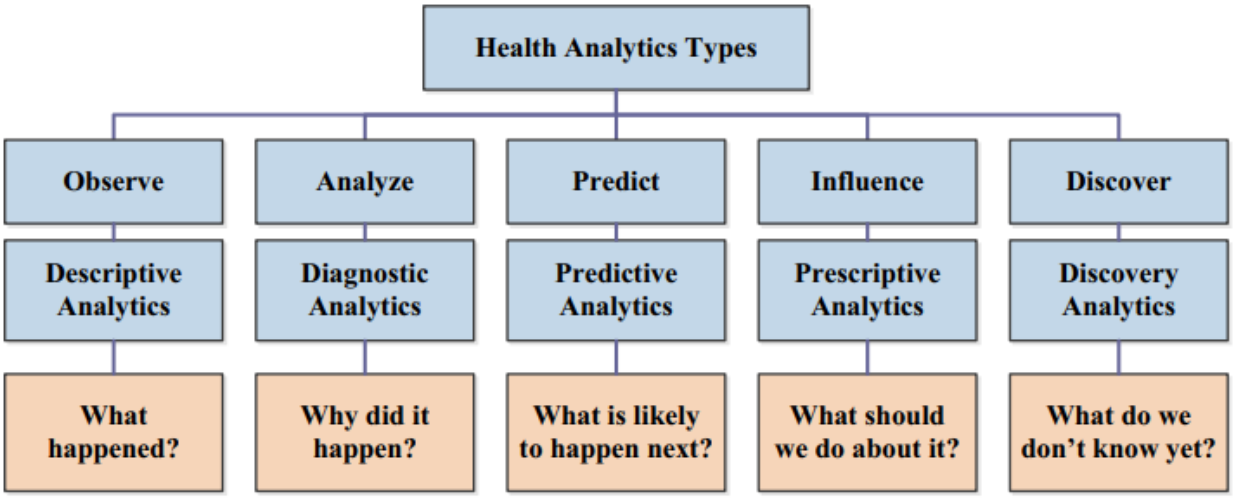
6.1: Τι είναι Big Data Analytics στο σύστημα υγείας;

Ο τομέας της υγειονομικής περίθαλψης, ο οποίος ιστορικά βασίζεται σε τεράστιες ποσότητες δεδομένων, βρίσκεται στο κατώφλι μιας επανάστασης που καθοδηγείται από τα Big Data Analytics. Η ανάλυση μεγάλων δεδομένων στην υγειονομική περίθαλψη αναφέρεται στη συστηματική χρήση προηγμένων αναλυτικών τεχνικών σε ποικίλα και μεγάλα σύνολα δεδομένων που παράγονται στον τομέα της υγειονομικής περίθαλψης. Αυτό περιλαμβάνει

δεδομένα από πηγές όπως ηλεκτρονικά αρχεία υγείας (EHR), ιατρική απεικόνιση, γονιδιακή αλληλουχία, φορητές συσκευές υγείας και μητρώα ασθενών. Στόχος είναι η εξαγωγή σημαντικών πληροφοριών, μοτίβων και τάσεων που μπορούν να ενημερώσουν για τη λήψη κλινικών αποφάσεων, να βελτιστοποιήσουν τη λειτουργική αποτελεσματικότητα, να προβλέψουν τα αποτελέσματα των ασθενών και να υποστηρίξουν ερευνητικές πρωτοβουλίες για τη βελτίωση της συνολικής παροχής υγειονομικής περίθαλψης και της φροντίδας των ασθενών.

6.2: Είδη αναλυτικών εφαρμογών

Η ανάλυση υγειονομικών Big data προσφέρει ευρύ φάσμα δυνατοτήτων για την παροχή υψηλής ποιότητας φροντίδας ασθενών, αλλά εμφανίζεται έλλειψη χρηστικών εφαρμογών πραγματικού χρόνου. Η αξιοποίηση των αναλυτικών στοιχείων μπορούν να μεταμορφώσουν τον τρόπο αντιμετώπισης της υγείας είτε σε ατομικό επίπεδο είτε επίπεδο πληθυσμού, καταγράφοντας της πληθυσμιακές συνήθειες/τάσεις και εντοπίζοντας μοτίβα που διαφορετικά θα ήταν μη ανιχνεύσιμα. Αυτές οι πολύτιμες πληροφορίες εξυπηρετούν στην πρόβλεψη και πρόληψη έκρηξης παθογενειών παθήσεων και βελτιώνουν την ποιότητα περίθαλψης ασθενειών.



Εικόνα 24 Κατηγορίες αναλυτικών δεδομένων[49]

Ηλεκτρονικά αρχεία υγείας (Electronic Health Record - EHR): Μία από τις κρίσιμες εφαρμογές των big data healthcare analytics αφορά τα ηλεκτρονικά αρχεία υγείας ή αλλιώς τον ηλεκτρονικό ιατρικό φάκελο. Τα ηλεκτρονικά ιατρικά αρχεία αποτελούν αναπόσπαστο μέρος της Descriptive Analysis. Κάθε άτομο/ασθενής διαθέτει ένα ψηφιακό αρχείο, το οποίο περιλαμβάνει το ιατρικό ιστορικό, τα αποτελέσματα εργαστηριακών εξετάσεων και τις ουσίες στις οποίες έχει διαπιστωθεί αλλεργική αντίδραση. Τα αρχεία διαμοιράζονται μέσω ασφαλών πληροφοριακών συστημάτων για παρόχους από το δημόσιο και τον ιδιωτικό τομέα. Κάθε εγγραφή εμπεριέχει ένα αρχείο είτε τη διάγνωση και τη συνισταμένη φαρμακευτική θεραπεία, είτε κάποια επιπρόσθετη

ιατρική εξέταση. Τα EHR διευκολύνουν το ιατρικό προσωπικό παρέχοντας γνώση την πλήρη κλινική τους κατάσταση του ασθενούς τους. Επίσης τα EHR mobile apps διαθέτουν προ-ενεργοποιημένες προειδοποιήσεις και υπενθυμίσεις στους χρήστες/ασθενείς για τα κανονισμένα ιατρικά ραντεβού για υποβολή νέων εργαστηριακών εξετάσεων και μη εκτελεσμένων φαρμακευτικών συνταγών[49][66]. Τέλος, τα ηλεκτρονικά αρχεία υγείας συμβάλουν στη μείωση κόστους αποτρέποντας την επανάληψη πολλαπλών εξετάσεων και εξαλείφοντας την ανάγκη για έντυπα αρχεία αμφότερων φορέα περίθαλψης και ασθενούς.

Προγνωστική ανάλυση (Predictive analytics): Η προγνωστική ανάλυση αξιοποιεί τα ιστορικά δεδομένα υγειονομικής περίθαλψης για εντοπισμό προτύπων και τάσεων για εξατομικευμένη, αλλά και οικουμενική πρόβλεψη εμφάνισης μελλοντικών παθήσεων. Οι πάροχοι ιατροφαρμακευτικής περίθαλψης μπορούν να λάβουν υψηλής ποιότητας τεκμηριωμένες αποφάσεις σχετικά με τις θεραπείες που θα προσφέρουν στους ασθενείς και τον βέλτιστο τρόπο προσαρμογής αυτών των θεραπειών στις ατομικές ανάγκες [51]. Παράλληλα, η προγνωστική ανάλυση συμβάλει στον πρόωρο εντοπισμό ασθενών που διατρέχουν κίνδυνο επιπλοκών ή υποτροπής. Η προγνωστική ανάλυση χρησιμοποιήθηκε ευρέως το 2020 για τον προσδιορισμό της επιπτώσεων του COVID-19 ξεχωριστά για κάθε άτομο. Οι ερευνητές της Αμερικανικής Χημικής Εταιρίας ανέπτυξαν μία εξέταση αίματος που χρησιμοποιεί προγνωστικά αναλυτικά στοιχεία και φασματοσκοπία υπέρυθρων μετασχηματισμών Fourier (ATR-FTIR) για να προβλεφθεί εάν ένα άτομο θα παρουσιάσει σοβαρά συμπτώματα COVID-19 ή όχι [50].

Διερευνητική ανάλυση (Exploratory analysis): Η διερευνητική ανάλυση στην υγειονομική περίθαλψη αποτελεί σημαντικό προκαταρκτικό βήμα για την διερεύνηση πολύπλοκων ιατρικών δεδομένων για τον εντοπισμό χρήσιμων μοτίβων, συσχετίσεων και πληροφοριών συμβάλλοντας στη διαδικασία λήψης αποφάσεων[49]. Περιλαμβάνει εκτενές φάσμα περιπτώσεων χρήσης, όπως έλεγχος δημογραφικών στοιχείων των ασθενών για την κατανόηση των αναγκών υγειονομικής περίθαλψης του πληθυσμού ή την αξιολόγηση του επιπολασμού και της γεωγραφικής κατανομής των ασθενειών για την αποτελεσματική κατανομή των πόρων[49][50][57]. Αυτή η ανάλυση, χρησιμοποιεί τα εργαλεία της οπτικοποίησης για διευκόλυνση στην διερμηνεία και τις στατιστικές μεθόδους για σύνοψη των δεδομένων για την ανακάλυψη κρυφών μοτίβων. Η διερευνητική ανάλυση χρησιμοποιείται ευρέως για την ανάλυση της πολυπλοκότητας του αυξανόμενου κόστους υγειονομικής περίθαλψης.

Προδιαγραφική ανάλυση (Prescriptive analytics): Η προδιαγραφική ανάλυση τίθεται σε εφαρμογή για την λήψη αποφάσεων με πληθώρα εναλλακτικών λύσεων, αξιοποιώντας τα αποτελέσματα της προγνωστικής ανάλυσης. Οι θεράποντες ιατροί εκτός από την εξέταση των συνέπειων και των αναμενόμενων αποτελεσμάτων, έχουν την δυνατότητα για τον άμεσο εντοπισμό βέλτιστης και εξατομικευμένης θεραπείας. Αυτή η νέα προσέγγιση, κατακερματίζει την παλαιότερη αντίληψη «one size fits all», το οποίο σε κάποιες περιπτώσεις ήταν επιβλαβής για τους ασθενής [50]. Όσα προαναφέρθηκαν επιτυγχάνονται με την χρήση hybrid data (συνδυασμός structured & unstructured) και προσαρμοστικούς αλγορίθμους, τα οποία χρησιμοποιούν ισχυρούς και αξιόπιστους μηχανισμούς ανατροφοδότησης.

Γνωστική ανάλυση (Cognitive analytics): Η γνωστική ανάλυση δεδομένων αποτελεί τον πιο προηγμένο τρόπο ανάλυσης δεδομένων που χρησιμοποιεί αλγορίθμους τεχνητής νοημοσύνης και μηχανικής μάθησης για την επεξεργασία και ανάλυση τεράστιων ποσοτήτων hybrid data[52]. Η cognitive analysis θεωρείται η εξέλιξη της discovery analysis, και συμβάλλει στην ανακάλυψη νέων φαρμάκων ή εναλλακτικών θεραπειών ή την ανίχνευση νέων ασθενειών ή άγνωστων παρενεργειών. Τα συστήματα πληροφοριών υγείας παρέχουν οριζόντιες κλινικές πληροφορίες σε ατομικό επίπεδο. Αναλύοντας το επίπεδο του ασθενούς δεδομένων μπορεί να αποδώσει συμπεράσματα και αποτελέσματα σε επίπεδο πληθυσμού, όπως η ισχύς των συσχέτισης μεταξύ της έκθεσης σε ιατρικό προϊόν και των επακόλουθων αποτελεσμάτων[49]. Το υγειονομικό προσωπικό αξιοποιεί την γνωστική ανάλυση, ώστε να δοκιμάσει μια υπόθεση πριν την εφαρμόσουν στον ασθενή.

6.3: Προοπτικές των BDA

Το επίκεντρο της βιομηχανίας της υγειονομικής περίθαλψης μετατοπίζεται προς το μοντέλο με επίκεντρο τον ασθενή, καθοδηγώντας τη στη χρήση περισσότερων δεδομένων για την ύπαρξη βέλτιστων αποτελεσμάτων. Οι οργανισμοί υγειονομικής περίθαλψης αρχίζουν να αξιοποιούν αυτήν την τεχνολογία, ωστόσο οι περισσότεροι απέχουν αρκετά από το να ευνοηθούν πλήρως τα οφέλη της.

Διαχείριση της υγείας του πληθυσμού: Η διαχείριση της υγεία του πληθυσμού αναφέρεται στις μεθόδους και τις παρεμβατικές ενέργειες που αποσκοπούν στη βελτίωση των αποτελεσμάτων της υγείας ενός ολόκληρου πληθυσμού, παρά στην εστίαση μεμονωμένων ασθενών. Με την ενσωμάτωση και χρήση των εργαλείων που προσφέρουν τα Big Data, οι φορείς της υγειονομικής περίθαλψης έχουν πλέον μια ευρύτερη προοπτική, που τους επιτρέπει να διακρίνουν μακροσκοπικές τάσεις στην υγεία[62]. Αυτές οι γνώσεις προέρχονται από την ενδελεχή ανάλυση των δεδομένων του πληθυσμού, βοηθούν στην διαμόρφωση δράσεων προσαρμοσμένων στις υγειονομικές προκλήσεις της κάθε κοινότητας. Η γήρανση του πληθυσμού, η αύξηση των χρόνιων ασθενειών και οι ανισότητες στην προσβασιμότητα στις υπηρεσίες υγειονομικής περίθαλψης, είναι μερικά από τα χαρακτηριστικά παραδείγματα που προβάλλουν επιτακτικά την ανάγκη για την υιοθέτηση στρατηγικών ευρείας φροντίδας του πληθυσμού[66]. Με την κατανόηση των υγειονομικών τάσεων, την πρόβλεψη πιθανών προβλημάτων και την εφαρμογή μέτρων για συγκεκριμένες ανάγκες μπορεί να επιτευχθεί μια πιο ισότιμη και προληπτική αντιμετώπιση των υγειονομικών προκλήσεων.

Τηλεϊατρική: Η τηλεϊατρική και το remote monitoring αντιπροσωπεύουν κομβικές αλλαγές στην παροχή υγειονομικών υπηρεσιών, οι οποίες οφείλονται στις τεχνολογικές εξελίξεις και στις

συνεχώς αυξανόμενες ανάγκες του παγκοσμίου πληθυσμού. Ο τομέας της υγειονομικής περίθαλψης έχει φέρει επανάσταση με τις μετρήσεις υγείας σε πραγματικό χρόνο να είναι πλέον προσβάσιμες στους γιατρούς, μειώνοντας την περιοδικότητα των φυσικών διαβουλεύσεων και εξασφαλίζοντας άμεση υγειονομική περίθαλψη [50][65]. Αυτή η άμεση πρόσβαση σε δεδομένα ξεπερνά επίσης τα γεωγραφικά εμπόδια, εκδημοκρατίζοντας την πρόσβαση στην υγειονομική περίθαλψη σε αστικές και αγροτικές περιοχές. Επιπλέον, το λειτουργικό κόστος που συνδέεται με την ενδονοσοκομειακή περίθαλψη έχει μειωθεί αισθητά με την τηλεϊατρική, καθιστώντας την υγειονομική περίθαλψη πιο οικονομικά αποδοτική και προσιτή.

Γονιδιακή ανάλυση: Η γονιδιακή ανάλυση βρίσκεται στο προσκήνιο για την εισαγωγή μιας νέας ιατρικής εποχής ακριβείας, φέρνοντας θεραπείες προσαρμοσμένες στην ατομική γενετική σύνθεση. Η ανάλυση big data εξασφαλίζει την υπολογιστική ισχύ και τους εξελιγμένους αλγορίθμους που είναι απαραίτητοι για τη γρήγορη ταξινόμηση αυτών των εκτεταμένων πληροφοριών. Η λεπτομέρεια της γονιδιακής προσδίδει βαθύτερη κατανόηση των ασθενειών σε μοριακό επίπεδο, παρέχοντας κρίσιμες γνώσεις που προηγουμένως ήταν ανέφικτες. Αυτή η μοριακή εστίαση χρησιμεύει ως άξονας στη φαρμακευτική έρευνα, διευκολύνοντας το σχεδιασμό φαρμάκων που αφορούν συγκεκριμένες μεταλλάξεις και στοχεύουν αποτελεσματικότερα στις ασθένειες [60]. Παρόλη την συνεισφορά ως προς την πρόβλεψη ασθενειών, η γονιδιακή ανάλυση δημιουργεί ηθικά διλήμματα σε σχέση με το γενετικό απόρρητο, την πιθανή κατάχρηση των γενετικών δεδομένων και τις ηθικές επιπτώσεις της γενετικής επεξεργασίας. Τέλος, οι τεχνολογίες που έχουν αναπτυχθεί για την γονιδιακή ανάλυση, όπως η CRISPR και η next-generation DNA sequence, εξυπηρετούν στην κατανόηση της ζωής σε μοριακό επίπεδο και συμβάλλουν ακόμα και στον τομέα της Γεωργίας[61][63].

Ανίχνευση υγειονομικής νοθείας: Η νοθεία στον τομέα της υγειονομικής περίθαλψης, που αποτελεί ζήτημα διάχυτο σε παγκόσμιο επίπεδο, περιλαμβάνει ένα ευρύ φάσμα παραπλανητικών μεθόδων, από την τιμολόγηση μη αποδοθέντων υπηρεσιών έως την κωδικοποίηση, όπου οι φορείς παροχής υπηρεσιών τιμολογούν ακριβότερες θεραπείες από αυτές που πραγματικά εφαρμόστηκαν. Η έκταση και η πολυπλοκότητα των συναλλαγών στον τομέα της υγειονομικής περίθαλψης καθιστούσαν ιστορικά τον εντοπισμό αυτών των αθέμιτων δραστηριοτήτων παρόμοιο με την εύρεση βελόνας στα άχυρα. Ωστόσο, με την έλευση της ανάλυσης μεγάλων δεδομένων, αυτό αλλάζει. Αυτά τα εξελιγμένα εργαλεία μπορούν να αναλύσουν γρήγορα εκατομμύρια συναλλαγές, εντοπίζοντας μοτίβα και ασυνέπειες που μπορεί να υποδηλώνουν απάτη[64]. Η ανίχνευση νοθείας στις υγειονομικές παροχές, προσφέρει αξιόλογη εξοικονόμηση χρημάτων για τα δημόσια συστήματα υγείας, αύξηση της εμπιστοσύνης από τους ασθενείς και την δυνατότητα ανακατεύθυνσης των διασωθέντων κεφαλαίων σε πραγματικές ανάγκες στον ιατροφαρμακευτικό τομέα[66].

6.4: Δεοντολογική χρήση των BDA

Η αλματώδης άνοδος της ανάλυσης μεγάλων δεδομένων στον τομέα της υγειονομικής περίθαλψης έχει προκαλέσει τόσο ενθουσιασμό όσο και τρόμο. Ενώ τα δυνητικά οφέλη είναι πολλαπλά, περιλαμβάνοντας τα πάντα, από την ενισχυμένη φροντίδα των ασθενών έως τη λειτουργική αποτελεσματικότητα, οι ηθικές προεκτάσεις αυτής της εξέλιξης απαιτούν σχολαστικό έλεγχο. Η δεοντολογική χρήση της ανάλυσης μεγάλων δεδομένων στην υγειονομική περίθαλψη είναι θεμελιώδους σημασίας για τη διασφάλιση της εμπιστοσύνης των ασθενών, τη διατήρηση της επαγγελματικής ακεραιότητας και την τήρηση των θεμελιωδών αρχών της υγειονομικής περίθαλψης [57].

Ιατρικό απόρρητο: Ο τομέας της υγειονομικής περίθαλψης, από τη φύση του, συγκεντρώνει βαθιά προσωπικές πληροφορίες για τα άτομα και η μη εξουσιοδοτημένη ή ακούσια αποκάλυψη τέτοιων δεδομένων μπορεί να οδηγήσει σε ολέθριες συνέπειες που κυμαίνονται από κοινωνικό στίγμα έως και απροκάλυπτες διακρίσεις. Εμφανίζεται η επιτακτική ανάγκη για την χρήση προηγμένων τεχνικών κρυπτογράφησης, αυστηρούς ελέγχους πρόσβασης από τα ιδρύματα υγειονομικής περίθαλψης, καθώς και να βρίσκονται σε συνεχή επαγρύπνηση έναντι απειλών στον κυβερνοχώρο [54][64]. Στη σφαίρα ενσωμάτωσης των τεχνολογιών big data στον τομέα της υγείας, η ιδέα της αυτονομίας των ασθενών και της συναίνεσης μετά από ενημέρωση αποκτά νέα σημασία. Σε αντίθεση με τις γνωστές ιατρικές διαδικασίες, όπου οι κίνδυνοι και τα οφέλη είναι συχνά άμεσοι και απτοί, οι επιπτώσεις από την κοινοποίηση των δεδομένων ενός ατόμου μπορεί να εκδηλωθούν χρονικά και με απροσδόκητους τρόπους. Οι ασθενείς επιβάλλουν να διατηρούν τα έντυπα συγκατάθεσης για την δυνητική χρήση, αποθήκευση και κοινοποίηση των ιατρικών δεδομένων τους. Με την χρήση των σαφών και χωρίς ορολογίες έντυπων συγκατάθεσης, οι ασθενείς διατηρούν τα δικαιώματα πρόσβασης, διόρθωσης, ακόμη και διαγραφής [62].

Αλγοριθμική/Τεχνολογική διαφάνεια: Η ασάφεια πλήθους αλγορίθμων που χρησιμοποιούνται στην ανάλυση μεγάλων δεδομένων αποτελεί μια άλλη βαθιά ηθική πρόκληση. Μια αλγοριθμική σύσταση, είτε αφορά διάγνωση, θεραπεία ή διοικητική απόφαση, μπορεί να επηρεάσει σημαντικά τα αποτελέσματα των ασθενών. Τα ενδιαφερόμενα μέρη, και κυρίως οι ασθενείς, δικαιούνται να κατανοήσουν τον τρόπο με τον οποίο αυτοί οι αλγόριθμοι λειτουργούν και λαμβάνουν αποφάσεις[54]. Επιπρόσθετα, τα ιδρύματα υγειονομικών υπηρεσιών οφείλουν να καθιερώσουν σαφή πρωτόκολλα για την ανθρώπινη εποπτεία και την περιοδική επανεξέταση των αλγοριθμικών αποτελεσμάτων, καθώς ο ρυθμός εμπλοκής σε σημαντικές αποφάσεις αυξάνεται συνεχώς και απαιτείται η ύπαρξη μηχανισμών αμφισβήτησης. Τέλος, δεδομένου ότι οι αλγόριθμοι συχνά εκπαιδεύονται με βάση ιστορικά δεδομένα, υπάρχει ο κίνδυνος ενίσχυσης παγιωμένων προκαταλήψεων. Αυτό ενδέχεται να οδηγήσει σε ανόμοια αποτελέσματα υγειονομικής περίθαλψης για διαφορετικές δημογραφικές ομάδες, αν δεν επιβληθεί τακτικοί και αυστηροί έλεγχοι στα ποικίλα σύνολα δεδομένα εκπαίδευσης των αλγορίθμων[64].

Εμπορευματοποίηση των δεδομένων: Στο μετασχηματιστικό τοπίο της υγειονομικής περίθαλψης που διαμορφώνεται από την ανάλυση μεγάλων δεδομένων, η συμβολή των εμπορικών φιλοδοξιών και της θεμελιώδους αρχής της ευημερίας των ασθενών αποτελεί μια ιδιαίτερη πρόκληση. Η δελεαστική δύναμη των δεδομένων υγείας για τις φαρμακευτικές εταιρείες, τους ασφαλιστές, τις νεοσύστατες επιχειρήσεις τεχνολογίας και τις εταιρείες γενετικών δοκιμών έγκειται στις δυνατότητές τους να προωθήσουν την ανάπτυξη φαρμάκων, να προσαρμόσουν τα πακέτα ασφάλισης, να βελτιώσουν τα προϊόντα υγείας και να βελτιώσουν την ακρίβεια των γενετικών δοκιμών [65]. Ωστόσο, εν μέσω αυτών των εμπορικών προοπτικών, οι ηθικές ανησυχίες, που κυμαίνονται από την προστασία της ιδιωτικής ζωής των δεδομένων έως τις πιθανές προκαταλήψεις και συγκρούσεις συμφερόντων, είναι διάχυτες. Καθώς περιηγούμαστε σε αυτό το εξελισσόμενο έδαφος, είναι επιτακτική ανάγκη να υιοθετηθεί μια ισορροπημένη προσέγγιση: μια προσέγγιση όπου η αυστηρή ρυθμιστική εποπτεία, η διαφανής διακυβέρνηση δεδομένων, τα ασθενοκεντρικά εμπορικά μοντέλα και ένα συνεργατικό ήθος με τη συμμετοχή όλων των ενδιαφερόμενων μερών διασφαλίζουν ότι, ενώ η καινοτομία ευδοκιμεί, η ακεραιότητα της υγείας των ασθενών παραμένει αμείωτη[65].

6.5: Συμπέρασμα

Η εφαρμογή των Big Data Analytics στο σύστημα υγείας έχει τη δυνατότητα να φέρει επανάσταση στη φροντίδα των ασθενών, να βελτιώσει την υγεία του πληθυσμού και να εξυγιάνει τις λειτουργίες. Ωστόσο, η αξιοποίηση αυτού του δυναμικού απαιτεί την αντιμετώπιση σημαντικών προκλήσεων που σχετίζονται με την προστασία της ιδιωτικής ζωής των δεδομένων, την ενσωμάτωση και τη ρύθμιση. Η συνεχής επένδυση σε υποδομές, έρευνα και συνεργασία θα είναι καθοριστικής σημασίας για την αξιοποίηση της δύναμης των μεγάλων δεδομένων ώστε να αναδιαμορφωθεί το μέλλον της υγειονομικής περίθαλψης.

Κεφάλαιο 7: Εφαρμογή ανάλυσης δεδομένων

7.1: Εισαγωγή

Κατά την τελευταία δεκαετία, υπήρξε σταδιακή αύξηση στο κόστος της υγειονομικής περίθαλψης, μια τάση που έχει συγκεντρώσει ουσιαστική προσοχή τόσο από τους επιστήμονες της οικονομίας και των δεδομένων, όσο και από τους πολιτικούς κύκλους. Αξίζει να σημειωθεί ότι αυτή η κλιμάκωση εντάθηκε περαιτέρω στον απόηχο των παρατεταμένων περιορισμών που επιβλήθηκαν από την πανδημία του COVID-19. Η διετής περίοδος διαλείπουσας εγκλεισμού και περιορισμών δημόσιας υγείας έχει προκαλέσει αυξημένη ευαισθητοποίηση του παγκόσμιου πληθυσμού σχετικά με τις πολυπλοκότητες, τις ευπάθειες που είναι εγγενείς στα συστήματα υγειονομικής περίθαλψης και την αύξηση χρήσης ιδιωτικών ασφαλιστικών εταιριών για την οικονομική διαχείριση μεταξύ κράτους και πολιτών.

Βασιζόμενη στα θεωρητικά ευρήματα για την χρήση των Big Data στην υγειονομική περίθαλψη αξιοποίησα το Apache Spark Ecosystem για την υλοποίηση διερευνητικής ανάλυσης και προγνωστικής ανάλυσης.

7.2 Δεδομένα

Το ασφαλιστικό σύστημα στις Ηνωμένες Πολιτείες Αμερικής διαχειρίζεται στο 90% από ιδιωτικές ασφαλιστικές εταιρίες. Οι ιδιωτικές ασφαλιστικές εταιρίες προσφέρουν πακέτα ανάλογα το εισόδημα και την οικογενειακή κατάσταση του κάθε πολίτη.

Η πηγή δεδομένων για τις αναλύσεις που έχουν εφαρμοστεί αποτελεί το «US DEPARTMENT OF HEALTH AND HUMAN SERVICES». Συγκεκριμένα τα datasets εξάχθηκαν το Kaggle, τα οποία περιέχουν πληροφορίες για τις χρονιές 2014,2015,2016. Τα Αρχεία Δημόσιας Χρήσης του Health Insurance Marketplace περιέχουν δεδομένα για προγράμματα υγείας και οδοντιατρικής που προσφέρονται σε ιδιώτες και μικρές επιχειρήσεις μέσω του US Health Insurance Marketplace.

Η πηγή δεδομένων για τα γεωγραφικά δεδομένα που αξιοποιήθηκα στην δημιουργία γραφημάτων αποτελεί το Εθνικό Ινστιτούτο Απογραφής Bureau.

7.3 Προδιαγραφική ανάλυση

Τα δεδομένα εξήχθησαν σε μη-κανονικοποιημένη μορφή (raw data), η οποία είναι δυσανάγνωστη και μη κατανοητή από τους ανθρώπους. Η προδιαγραφική ανάλυση εξυπηρετεί στην απλοποίηση και κατανόηση αυτών των πολύπλοκων και πολυάριθμων δεδομένων. Το input data set περιέχει 70 αρχεία με πληροφορίες για κάθε χρονιά ξεχωριστά. Για την εφαρμογή των τεχνικών ανάλυσης δεδομένων αξιοποιήθηκαν μόνο τρία που περιέχουν δεδομένα για όλες τις χρονιές. Για κάθε αρχείο csv

πραγματοποιήθηκε Data Cleansing, ώστε να απομακρυνθούν τυχόν empty ή null πλειάδες και οι στήλες των αρχείων να είναι ορισμένες με τους ορθούς τύπους δεδομένων. Η διαδικασία του καθαρισμού των δεδομένων μειώνει τα λογικά λάθη κατά την διαδικασία της επεξεργασίας δεδομένων.

Αναλυτικότερα:

Όνομα αρχείου	Πλήθος στηλών	Πλήθος πλειάδων
Plan Attributes	175	7.773.353
Rate	24	12.694.448
Benefits Cost Sharing	33	5.048.468

Πίνακας 3: Μέγεθος αρχείων εφαρμογής

Επιπρόσθετα, λαμβάνουμε τις πληροφορίες για τις μέγιστες και ελάχιστες τιμές, στατιστικά στοιχεία για κάθε στήλη. Τέλος, με την κάθαρση των αρχείων από ανεπιθύμητες τιμές, τα δεδομένα αποθηκεύονται σε αρχείο parquet, το οποίο εξυπηρετεί την μείωση του όγκου του αρχείου και ταχύτερη επεξεργασία από τα υπολογιστικά συστήματα.

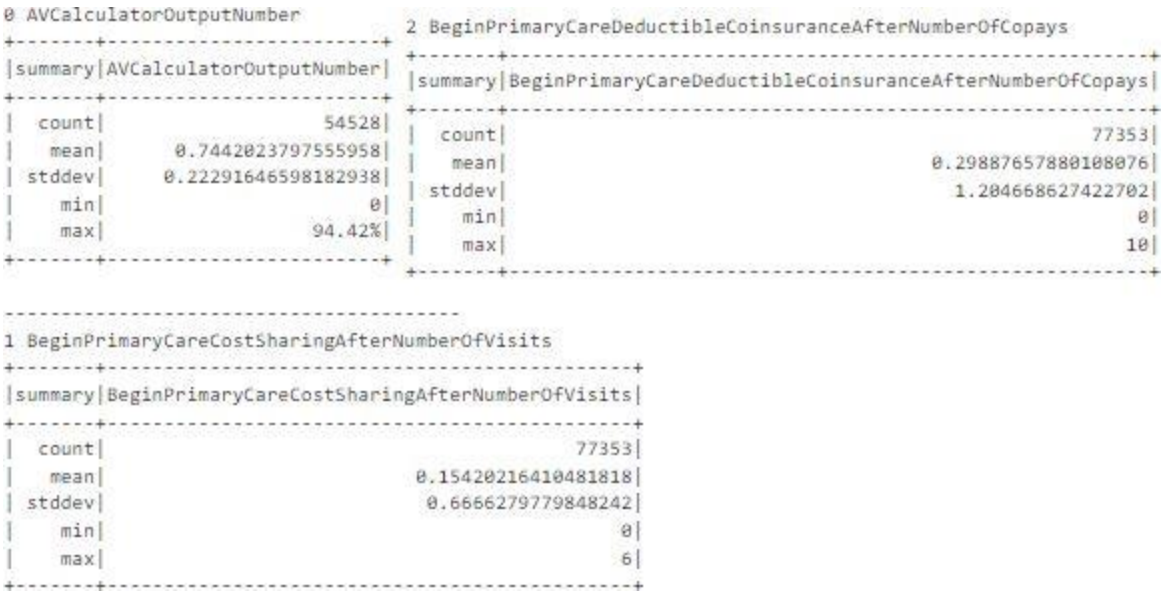
Ο κώδικας για την προδιαγραφική ανάλυση βρίσκεται στο παράρτημα.

Στις παρακάτω εικόνες παρατίθενται ενδεικτικά μερικοί από τους εξαγόμενους πίνακες για κάθε αρχείο κατά την προδιαγραφική ανάλυση.

```
2 CoinsInnTier1
+-------+--------------+
|summary| CoinsInnTier1|
+-------+--------------+
|  count|       3934611|
|   mean|          null|
| stddev|          null|
|    min|            $0|
|    max|Not Applicable|
+-------+--------------+

5 CopayInnTier1
+-------+-------------+
|summary|CopayInnTier1|
+-------+-------------+
|  count|      3934621|
|   mean|          0.0|
| stddev|          0.0|
|    min|           $0|
|    max|          Yes|
+-------+-------------+

4 CoinsOutofNet
+-------+--------------+
|summary| CoinsOutofNet|
+-------+--------------+
|  count|       3934621|
|   mean|          null|
| stddev|          null|
|    min|            $0|
|    max|Not Applicable|
+-------+--------------+

3 CoinsInnTier2
+-------+--------------+
|summary| CoinsInnTier2|
+-------+--------------+
|  count|        476881|
|   mean|          null|
| stddev|          null|
|    min|            $0|
|    max|Not Applicable|
+-------+--------------+

6 CopayInnTier2
+-------+--------------+
|summary| CopayInnTier2|
+-------+--------------+
|  count|        476881|
|   mean|           0.0|
| stddev|           0.0|
|    min|            $0|
|    max|Not Applicable|
+-------+--------------+

7 CopayOutofNet
+-------+--------------+
|summary| CopayOutofNet|
+-------+--------------+
|  count|       3934619|
|   mean|           0.0|
| stddev|           0.0|
|    min|            $0|
|    max|Not Applicable|
+-------+--------------+
```

Εικόνα 25:Benefits Cost Sharing

Στην παραπάνω εικόνα αποτυπώνονται οι πληροφορίες για τις οικονομικές εισφορές των πολιτών (“CoinsInnTier1”,” CoinsInnTier2”,”CoinsOutOfNet”) και του κράτους (“CopayInnTier1”,” CopayInnTier2”,”CopayOutOfNet”) ως προς τις ιδιωτικές ασφαλιστικές εταιρείες.



```
0 AVCalculatorOutputNumber
+-------+-------------------------+
|summary|AVCalculatorOutputNumber|
+-------+-------------------------+
|  count|                    54528|
|   mean|      0.7442023797555958|
| stddev|     0.22291646598182938|
|    min|                       0|
|    max|                  94.42%|
+-------+-------------------------+

2 BeginPrimaryCareDeductibleCoinsuranceAfterNumberOfCopays
+-------+-----------------------------------------------------------+
|summary|BeginPrimaryCareDeductibleCoinsuranceAfterNumberOfCopays|
+-------+-----------------------------------------------------------+
|  count|                                                      77353|
|   mean|                                 0.29887657880108076|
| stddev|                                  1.204668627422702|
|    min|                                                         0|
|    max|                                                        10|
+-------+-----------------------------------------------------------+

----------------------------------------
1 BeginPrimaryCareCostSharingAfterNumberOfVisits
+-------+-----------------------------------------------+
|summary|BeginPrimaryCareCostSharingAfterNumberOfVisits|
+-------+-----------------------------------------------+
|  count|                                          77353|
|   mean|                        0.15420216410481818|
| stddev|                         0.6666279779848242|
|    min|                                             0|
|    max|                                             6|
+-------+-----------------------------------------------+
```

Εικόνα 26: Plan Attributes

Στην παραπάνω εικόνα αποτυπώνονται οι πληροφορίες για το ποσοστό κάλυψης υγειονομικών εξόδων από ένα πρόγραμμα υγείας.

7.4 Διερευνητική ανάλυση

Η διερευνητική ανάλυση αξιοποιείται από τους υπεύθυνους χάραξης δημόσιας υγειονομικής πολιτικής και ασφαλιστικές εταιρίες, καθώς μπορούν να αποκρυπτογραφηθούν πολύπλευροι καθοριστικοί παράγοντες που οδηγούν στην συνεχή αύξηση του κόστους υγειονομικών παροχών και πραγματοποιηθούν προγνωστικά μοντέλα για το μελλοντικό κόστος. Στόχος αυτής της ανάλυσης είναι η απάντηση πολύπλοκων ερωτημάτων.

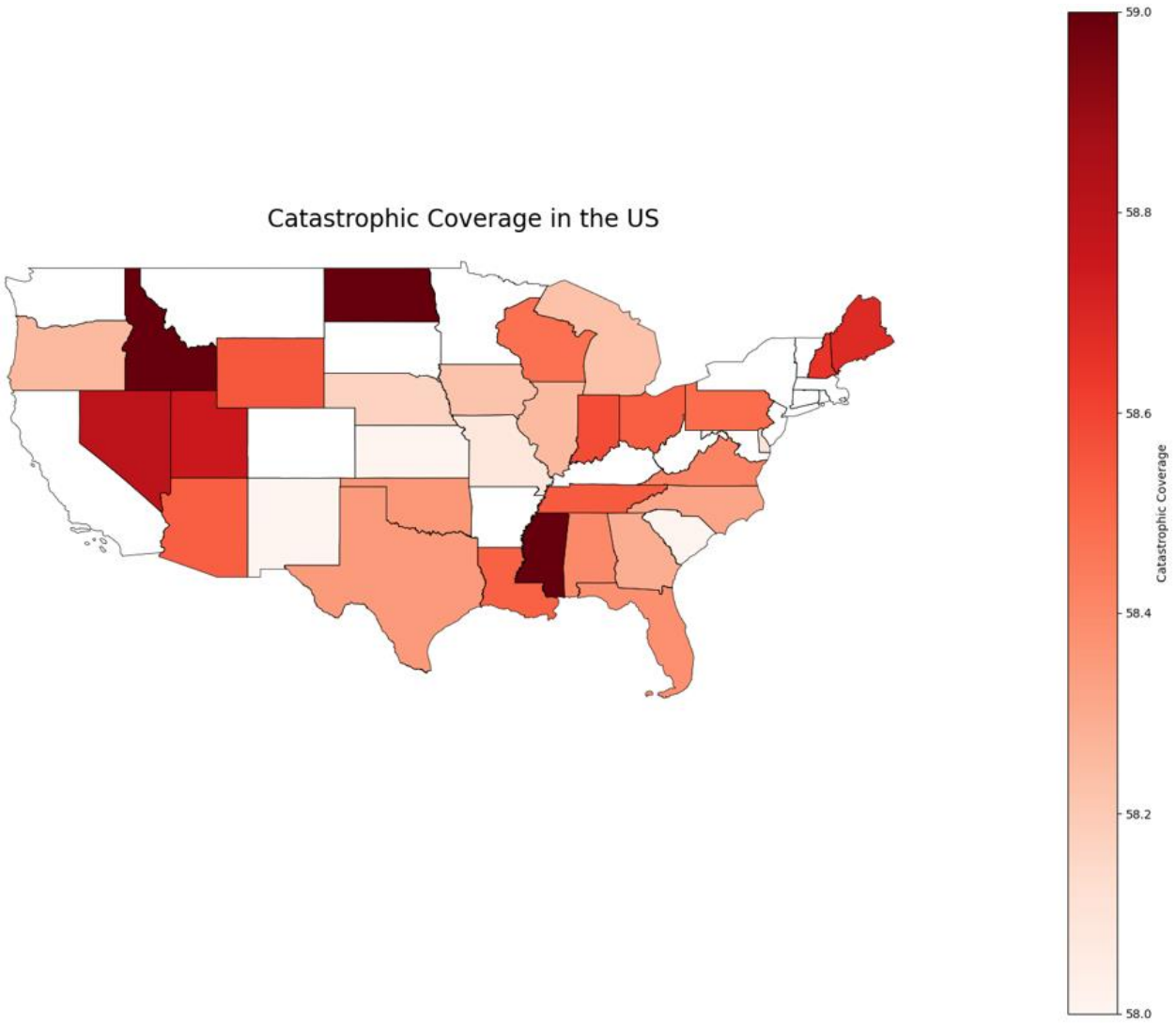
7.4.1 Κατηγοριοποίηση προγραμμάτων υγείας

Το αρχείο planAttributes περιέχει πληροφορίες για την στήλη “Issuer Actuarial value”, η οποία αναφέρεται στο ποσοστό που καταβάλλεται από ένα πρόγραμμα υγείας επί του ποσοστού του συνολικού επιτρεπόμενου κόστους των παροχών[59]. Δημιούργησα ένα υποσύνολο σύνολο δεδομένων, το οποίο περιέχει διαφορετικές βαθμίδες κάλυψης (Catastrophic, Bronze, Silver, Gold, Platinum) με βάση την Αναλογιστική αξία και τους κωδικούς των Αμερικάνικων πολιτειών (‘StateCode’) για το χρονικό διάστημα 2014-2016.

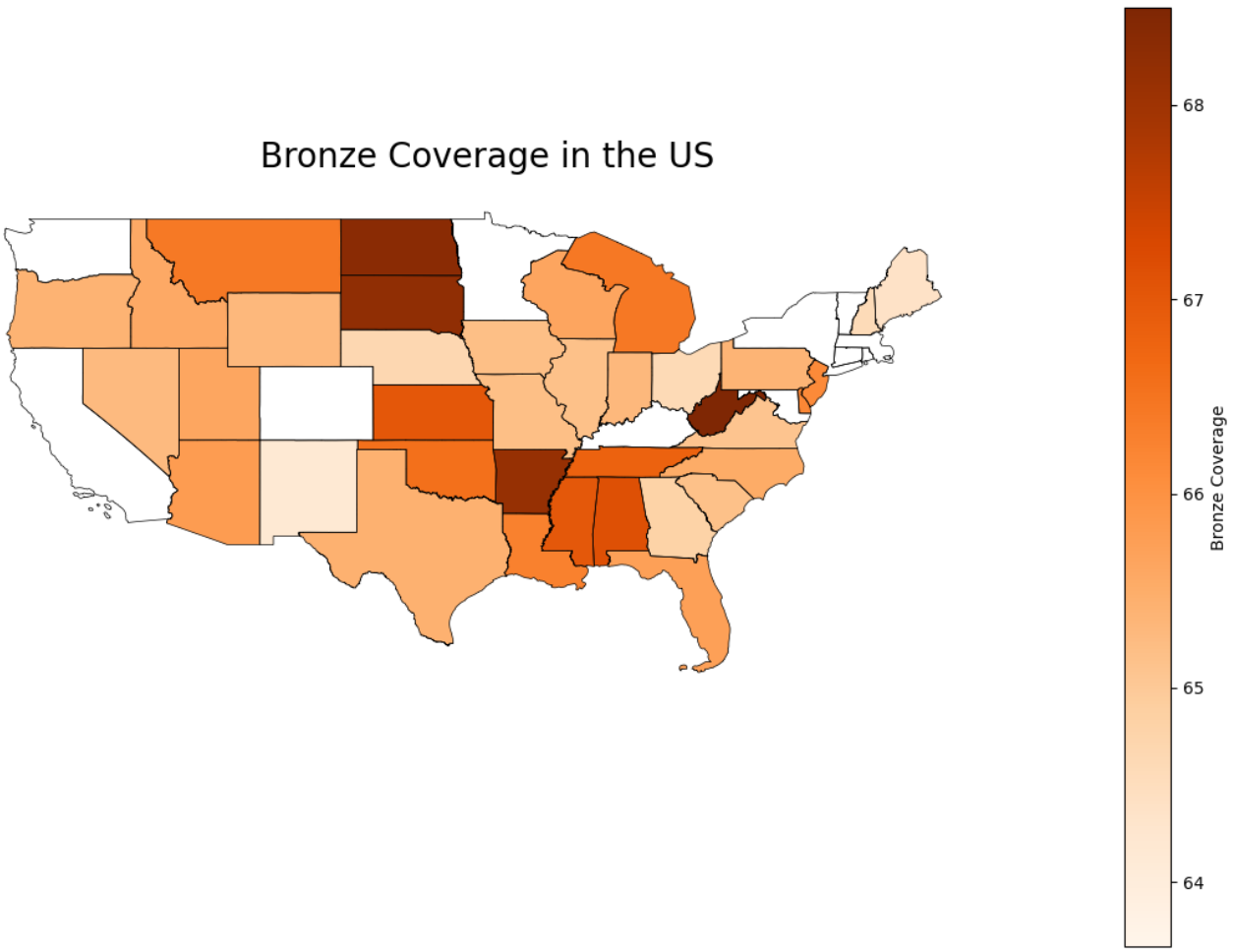
Κατηγορία	Αναλογιστική Αξία
Catastrophic	<60%
Bronze	60% - 70%
Silver	70% - 80%
Gold	80% - 90%
Platinum	90%>

Πίνακας 4: Στατιστικά στοιχεία κάθε κατηγορίας

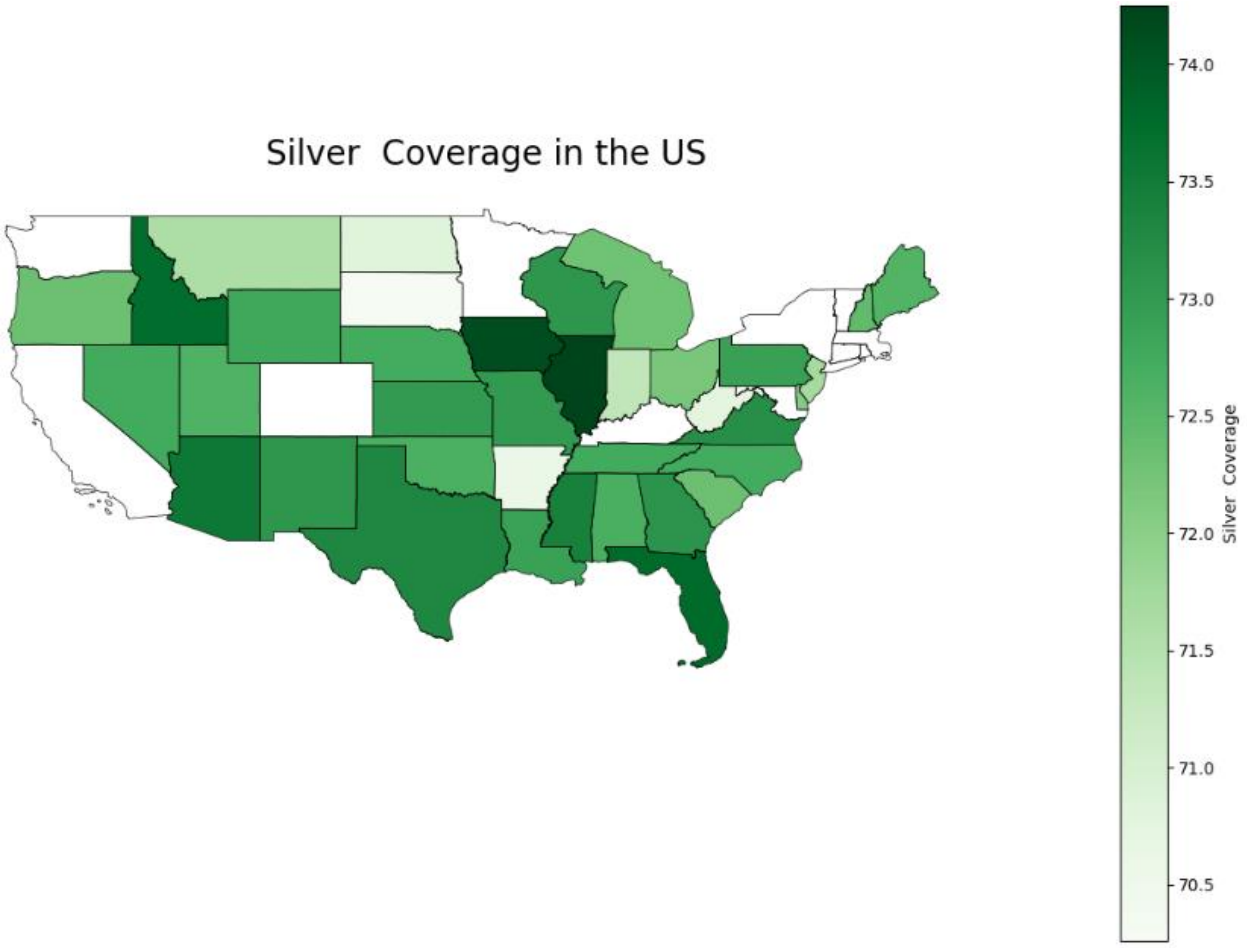
Αν το υποσύνολο δεδομένων που δημιουργήθηκε παρουσιαστεί σε μορφή πίνακα, θα είναι δυσνόητο. Με την χρήση των βιβλιοθηκών geopandas και matplotlib δημιούργησα για κάθε κατηγορία χάρτες, όπου εμφανίζονται τα ποσοστά κάθε κατηγορίας ανά πολιτεία. Για την χάρτες χρησιμοποίησα τις γεωγραφικές συντεταγμένες από το Εθνικό Ινστιτούτο Απογραφής Bureau**.** Σύνδεσα το υποσύνολο των κατηγοριών με αρχείο γεωχωρικής διανυσματικής μορφής.



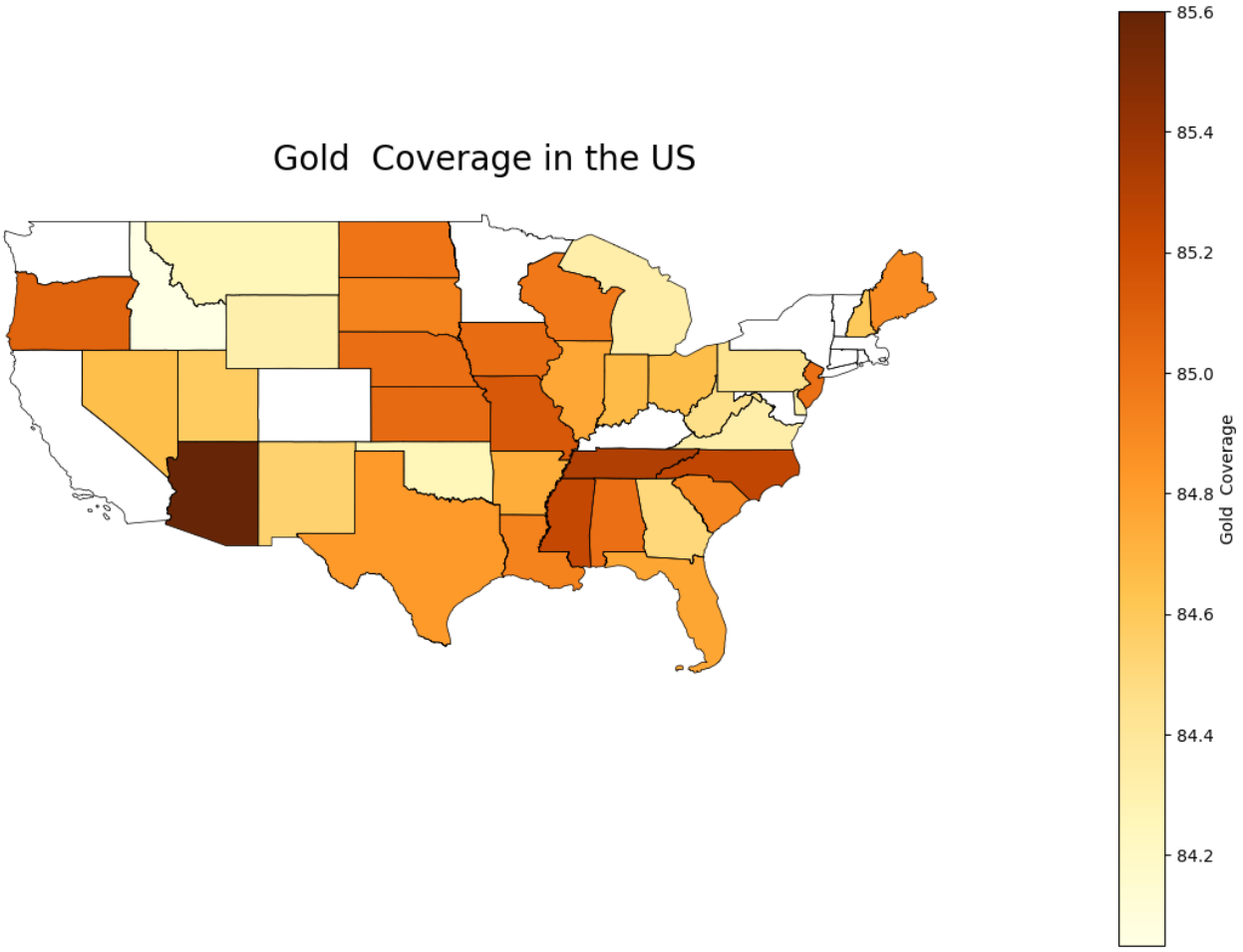
Εικόνα 27:Catastrophic category



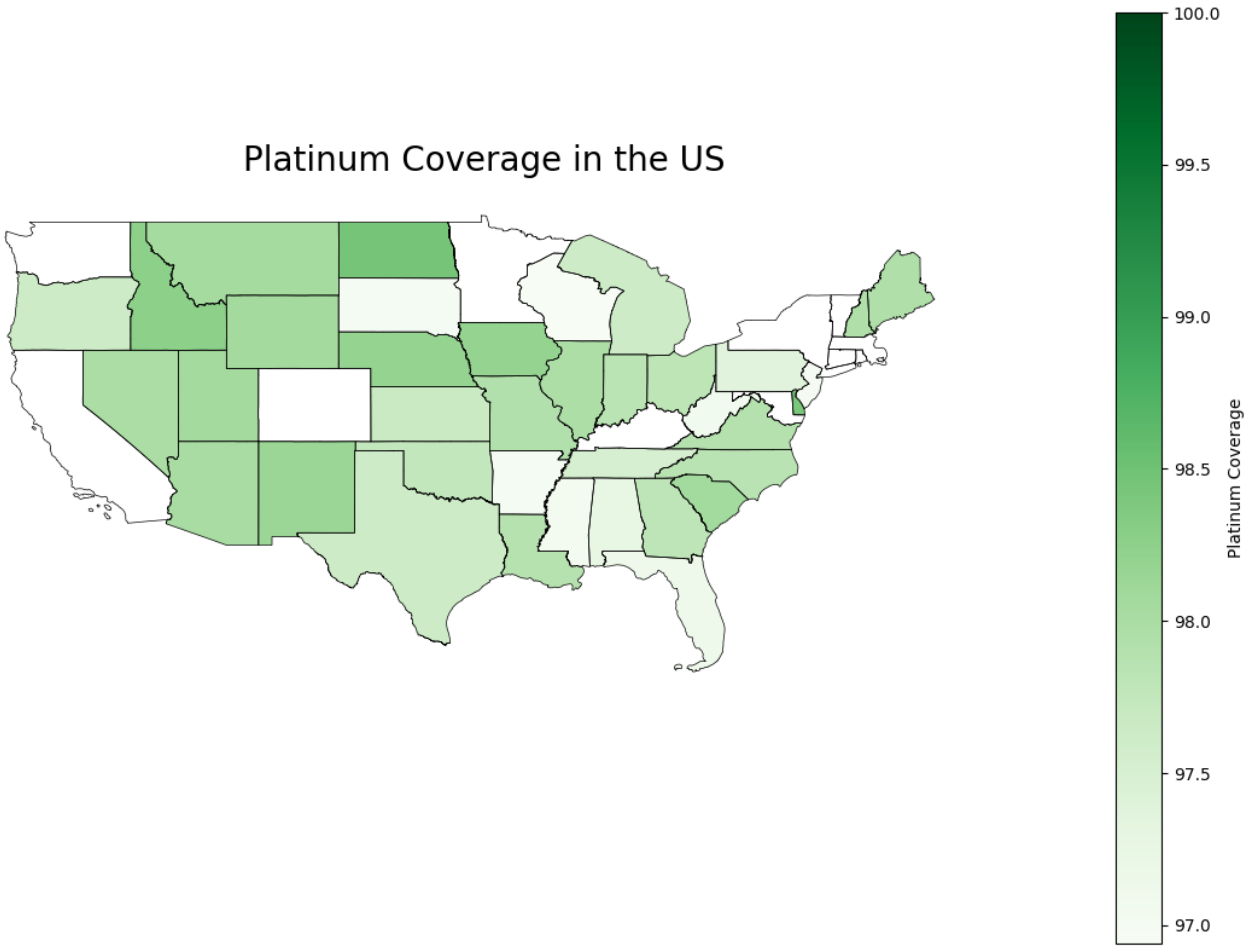
Εικόνα 28:Bronze category



Εικόνα 29:Silver category



Εικόνα 30:Gold category



Εικόνα 31:Platinum category

Πηγή έμπνευσης για την υλοποίηση της οπτικοποίησης των κατηγοριών προγραμμάτων υγείας αποτελεί το Kaggle notebook, στο οποίο πραγματοποιείται η δημιουργία αυτού του υποσυνόλου με την προγραμματιστική γλώσσα R.

Ο κώδικας για το 7.4.1 βρίσκεται στο παράρτημα.

7.4.2 Ασφαλιστικές εισφορές

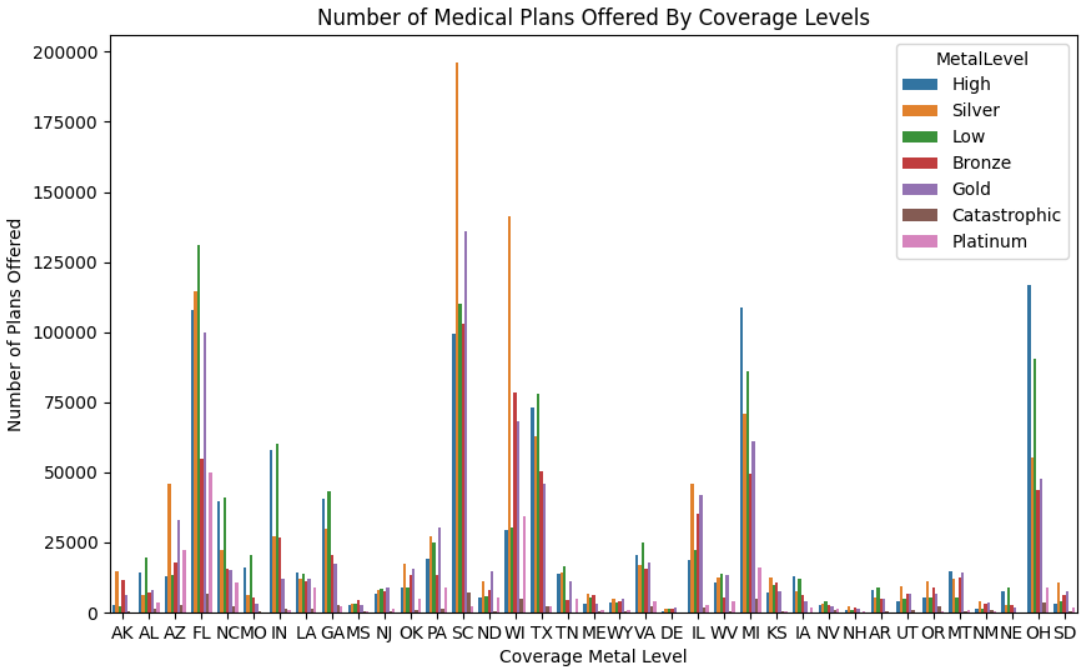
Το 2015 έγινε αισθητό στους πολίτες η αύξηση των ατομικών ασφαλιστικών υγειονομικών εισφορών. Με την χρήση της EDA απαντώνται τα ερωτήματα που αφορούν

a) το πλήθος των διαθέσιμών προγραμμάτων υγείας ανά πολιτεία
b) την σύγκριση προτίμησης των ατομικών ασφαλιστικών πακέτων με τα οδικά ασφαλιστικά προγράμματα ανά πολιτεία
c) την ατομική ασφαλιστική εισφορά ανά ηλικιακή ομάδα

d) την κατανομή των ατομικών ασφαλιστικών εισφορών ανά πρόγραμμα υγείας

Για την απάντηση των προαναφερθέντων ερωτημάτων αξιοποιείται το αρχείο Rate, το οποίο περιέχει πληροφορίες για την ηλικία και τις ατομικές ασφαλιστικές εισφορές. Στην στήλη της ηλικίας (“Age”) περιέχει την τιμή “Family Option” για τους πολίτες που εξαρτώνται από οικογενειακά ασφαλιστικά προγράμματα και δεν πληρώνουν εισφορές.

a) Με την συγχώνευση των κατηγοριών προγραμμάτων υγείας που δημιουργήθηκαν στο προηγούμενο ερώτημα μπορούμε να εντοπίσουμε το πλήθος των διαθέσιμων προγραμμάτων υγείας ανά πολιτεία.

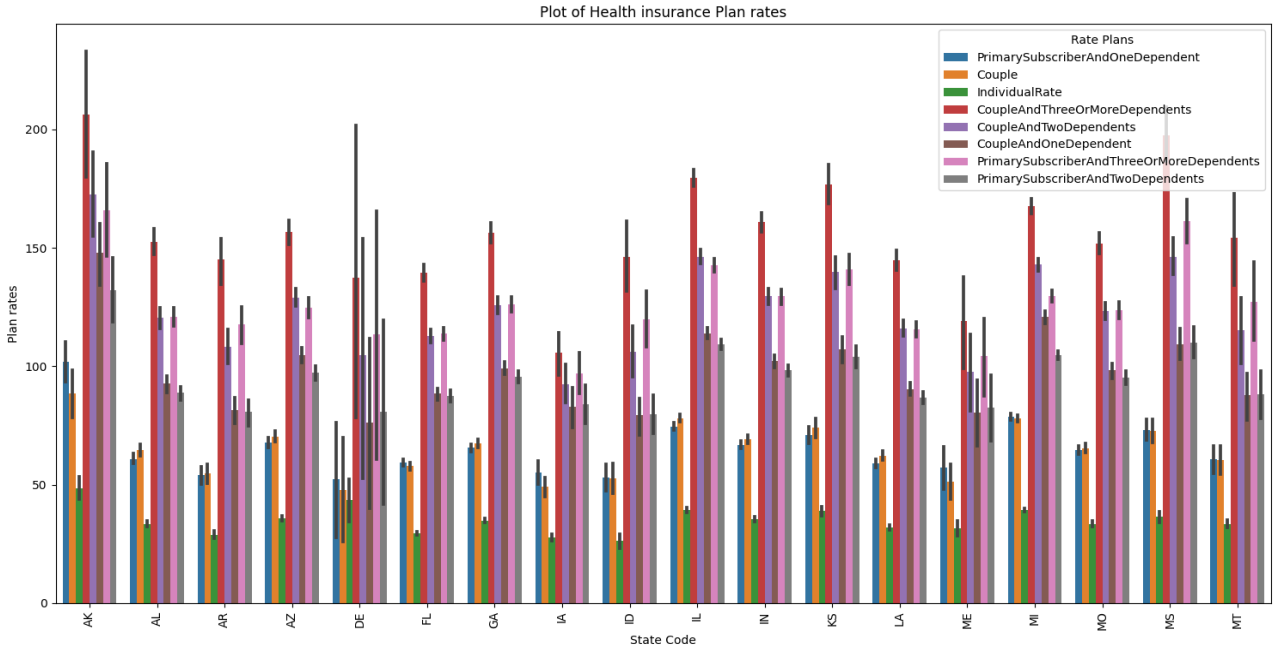


Εικόνα 32: Number of Medical Plans Offered By Coverage Levels

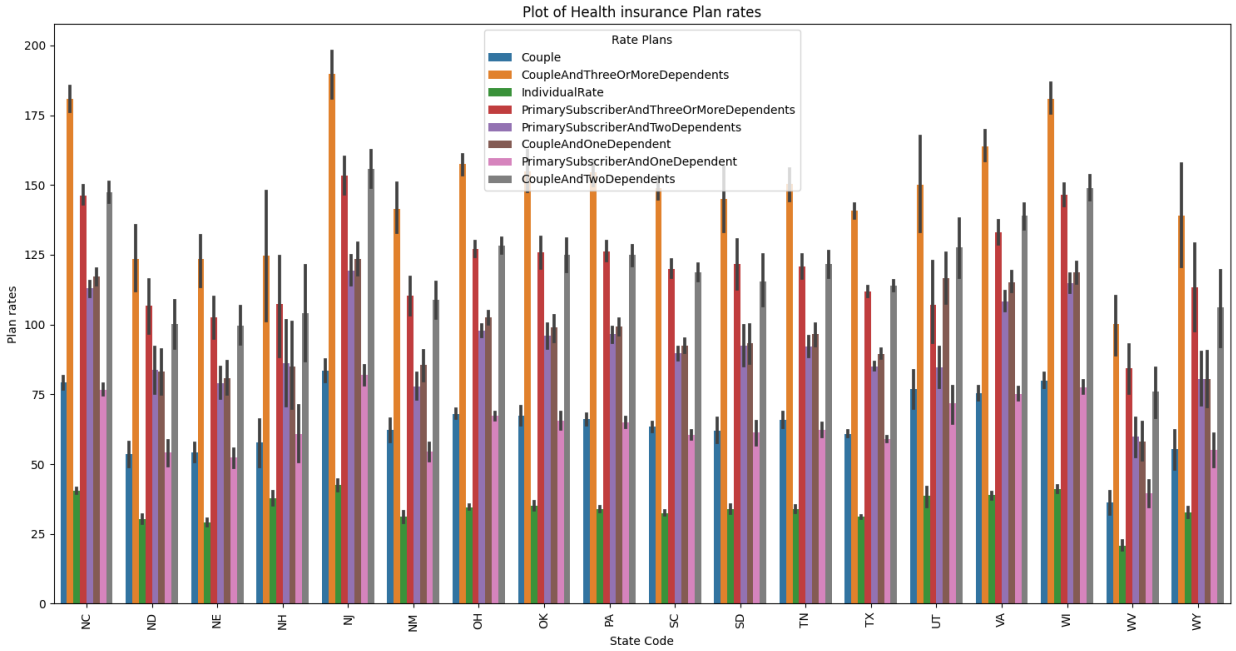
b) Το παραπάνω διάγραμμα έχει δημιουργηθεί με βάση το Actuarial Value και τις ατομικές εισφορές. Δεν εξάγονται πληροφορίες για τα ομαδικά προγράμματα υγείας.

Το αρχείο Benefits Cost Sharing περιλαμβάνει την στήλη “BenefitName”, όπου αναφέρει τις κωδικοποιημένες ονομασίες όλων των ομαδικών προγραμμάτων. Στους πολίτες όπου έχουν μόνο ατομικό ασφαλιστικό πρόγραμμα το value της στήλης είναι “InvidualRate’. Το περιεχόμενο της στήλης “BenefitName”, αποτελούν ξεχωριστά πεδία στο αρχείο Rate. Συνδυάζοντας αυτά τα 2 αρχείο δημιούργησα ένα νέο υποσύνολο το οποίο περιλαμβάνει τα πεδία: "Individual Rate", "Couple", "Primary Subscriber And One Dependent", "Primary Subscriber And Two Dependents", "Primary Subscriber And

Three Or More Dependents", "Couple And One Dependent", "Couple And Two Dependents", "Couple And Three Or More Dependents" και “State Code”. Το νέο υποσύνολο που δημιουργήθηκε, αποθηκεύτηκε σε αρχείο μορφής Parquet, για ταχύτερη επεξεργασία από τα υπολογιστικά συστήματα και για μικρότερη κατανάλωση αποθηκευτικού χώρου.

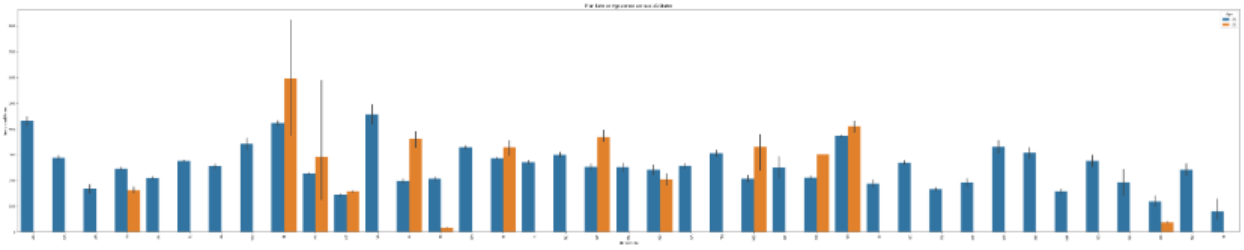


Εικόνα 33: Plans per state (1)

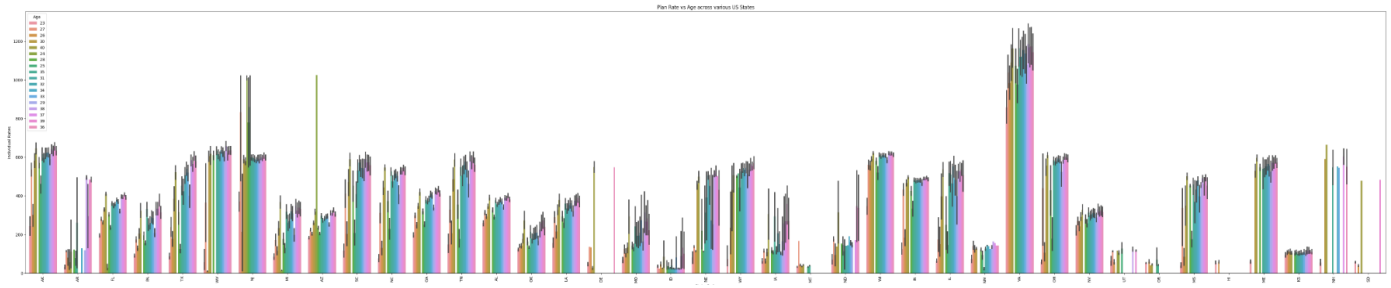


Εικόνα 34: Plans per state (2)

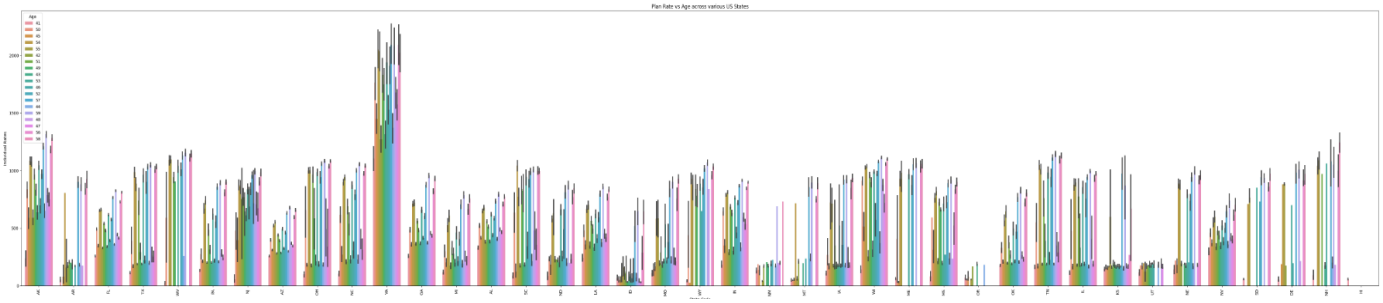
c) Οι ασφαλιστικές εισφορές δεν διαφέρουν μόνο ανάλογα την πολιτεία διαμονής, αλλά και την ηλιακή ομάδα καθώς καλύπτονται διαφορετικές ανάγκες. Στις παρακάτω εικόνες παρατίθενται γίνεται σύγκριση των ασφαλιστικών προγραμμάτων ανά ηλιακή ομάδα και πολιτεία διαμονής.



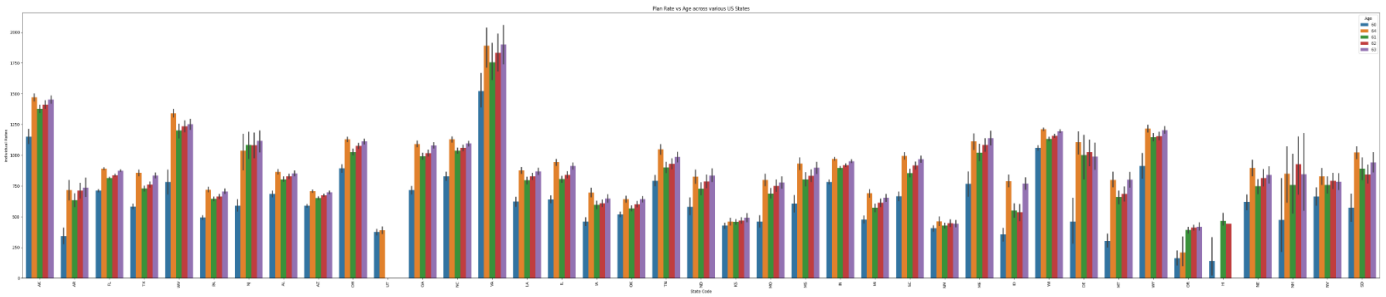
Εικόνα 35:Age 0-22



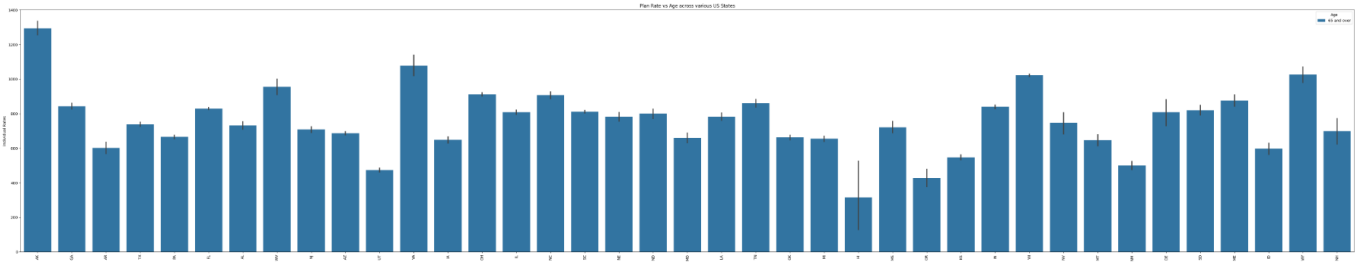
Εικόνα 36:Age 23-40



Εικόνα 37: Age 41-59

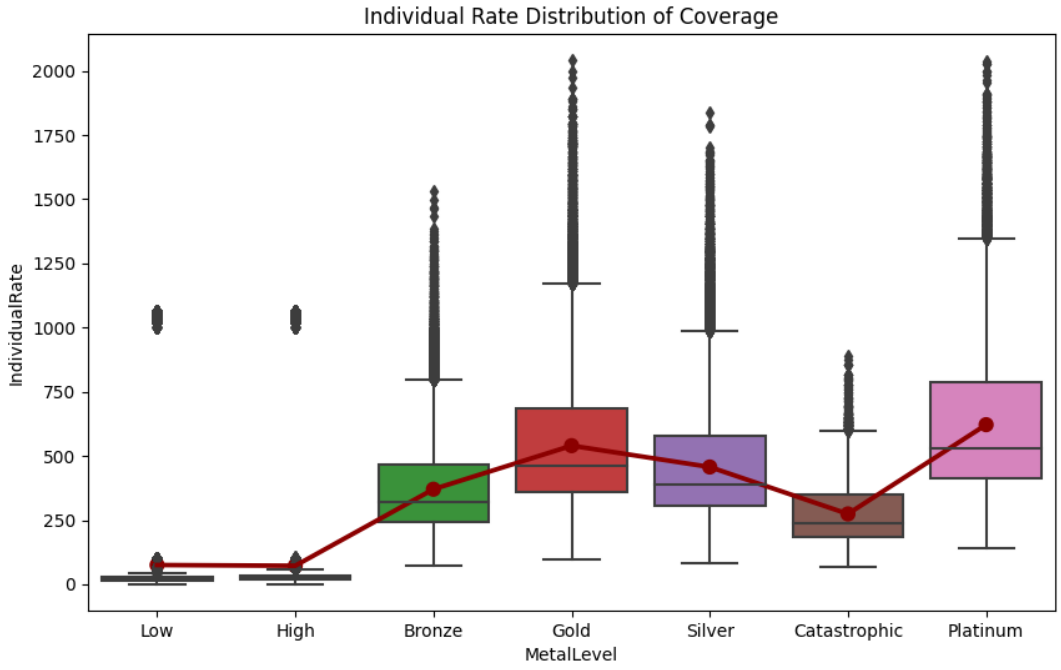


Εικόνα 38: Age 60-64



Εικόνα 39: Age 65+

d) Ενδιαφέρον στην ανάλυση των ασφαλιστικών εισφορών αποτελεί η τάση κατανομής ανά κατηγορία προγραμμάτων ασφάλισης.



Εικόνα 410:Individual Rate Distribution of Coverage

Πηγή έμπνευσης για την υλοποίηση αποτελεί το Kaggle notebook, στο οποίο πραγματοποιείται η δημιουργία αυτού του υποσυνόλου με την προγραμματιστική γλώσσα Python.
Ο κώδικας για το 7.4.2 βρίσκεται στο παράρτημα.

7.4.3 Οδοντιατρική περίθαλψη

Η οδοντιατρική περίθαλψη περιλαμβάνει ένα ευρύ φάσμα στους τρόπους αντιμετώπισης και ποικίλει κατά τη διάρκεια της ζωής από την πρώιμη ζωή έως την τρίτη ηλικία, είναι αναπόσπαστο κομμάτι της γενικής υγείας και υποστηρίζει τα άτομα να συμμετέχουν στην κοινωνία και να αξιοποιήσουν τις δυνατότητές τους. Σύμφωνα με άρθρα στα ΜΜΕ, οι ΗΠΑ ανήκουν στις χώρες με αυξημένη πολυπλοκότητα στην ασφαλιστική κάλυψη για την στοματική υγιεινή. Σε κάθε πολίτη γεννούνται τα εξής ερωτήματα:

a) Ποιοι τύποι οδοντιατρικών υπηρεσιών είναι πιο διαδεδομένοι και πώς συγκρίνονται ως προς τη σχετική συχνότητά τους στο σύνολο δεδομένων;
b) Πώς επηρεάζονται οι μηνιαίες οικονομικές ασφαλιστικές εισφορές από το κάπνισμα σε ανάλογα με την ηλικία και τα μηνιαία έξοδα σε καπνιστικά είδη;

Για την απάντηση των ερωτημάτων χρειάζεται να αποκτηθεί η γνώση για τις οδοντιατρικές υπηρεσίες που καλύπτονται από τις ασφαλιστικές εταιρίες. Τα δεδομένα για την στοματική περίθαλψη βρίσκονται στο αρχείο Benefit Cost Share και αποτελούν τα values της στήλης “BenefitName”. Παράλληλα χρειαζόμαστε την πληροφορία για το πλήθος των προγραμμάτων ασφάλισης υγείας που καλύπτουν αυτές τις υπηρεσίες, από τις στήλες του αρχείου Plan Attributes .

BenefitName	Covering Plans
Dental Check-Up f...	35413
Initial Oral Exam...	6
Periodic Oral Exa...	6
Teeth Cleaning - ...	6
Prophylaxis (clea...	2
Prophylaxis (Clea...	18
Prophylaxis (Clea...	8

Εικόνα 11:Routine exams

BenefitName	Covering Plans
Basic Dental Care...	9831
Basic Dental Care...	33662
Routine Dental Se...	9888
Specified Non-Rou...	85
Specified Non-Rou...	3049

Εικόνα 412: Basic Dentalcare

BenefitName	Covering Plans
Major Dental Care...	33590
Major Dental Care...	7726
Specified Non-Rou...	85
Specified Non-Rou...	43
Specified Non-Rou...	3049

Εικόνα 43: Major Dentalcare

BenefitName	Covering Plans
Dental X-Rays	57
X-rays and Diagno...	65221
X-rays of Entire ...	12
Bitewing X-rays -...	27
Bitewing X-rays -...	12
Panoramic / Compl...	7
X-rays of Entire ...	20
Routine X-Rays	4
Bite Wing X-Ray -...	6
Dental X-rays	10
Routine Exams and...	2
Routine Exams and...	8
Routine Exams and...	10
Routine Exams an...	8

Εικόνα 44: X-Ray

BenefitName	Covering Plans
Topical Flouride	63
Fluoride Treatmen...	27
Topical Fluoride	4
Fluoride Treatments	4

Εικόνα 45: Fluoride treatment

BenefitName	Covering Plans
Removal of Fixed ...	38
Removal of Fixed ...	27
Extractions	67
Extractions (Simp...	32
Removal of teeth ...	16
Removal of oral t...	16
Surgical Extractions	4
Simple Extractions	4
Removal of Impact...	6
Removal of Uncomp...	6
Removal of Impact...	6
Surgical Extracti...	4
Surgical Extracti...	16
Surgical Extracti...	4
Removal of Fxed S...	2

Εικόνα 46:Extractions

BenefitName	Covering Plans
Root Canal Therap...	58
Root Canal Therap...	32
Endodontic Proced...	17
Retreatment of pr...	16
Root canal therap...	16
Endodontic Proced...	17
Endodontics	4
Single Root Canal...	6
Bi-Root Canal - C...	6
Molar Root Canal ...	6
Endodontics - Adult	8
Root Canal Therap...	4
Endodontics - Child	20
Root canal Therap...	3

Εικόνα 47:Root Canals

BenefitName	Covering Plans
Sealants - Child	85
Sealants	71

Εικόνα 48:Sealants

BenefitName	Covering Plans
Periodontal and O...	61
Periodontal Maint...	64
Periodontal Root ...	56
Root Scaling and ...	32
Periodontal Maint...	81
Periodontal Proce...	17
Periodontal - Oth...	17
Periodontal Maint...	20
Periodontal Scali...	17
Periodontal Root ...	10
Periodontal Scali...	6
Periodontal - Adult	8
Periodontal - Child	20

Εικόνα 49:Periodontics

BenefitName	Covering Plans
Retrograde Fillin...	32
Fillings - Child	32
Fillings	59
Restorative Services	25
Major Restorative...	5
Posterior Composi...	28
Posterior Composi...	22
Posterior Composi...	33
Posterior Composi...	22
Amalgam and compo...	16
Fillings/Amalgam	4
Cavities - Amalga...	6
Cavities - Amalga...	6
Restorative Denti...	20
Restorative Denti...	8
Posterior Composi...	2

Εικόνα 51:Fillings

BenefitName	Covering Plans
Orthodontia - Adult	2012
Occlusal Adjustments	52
Non-Medically Nec...	237
Orthodontia - Child	34851
Occlusal Guard - ...	32
Non-Medically Nec...	274
Occlusal Adjustment	4
Non-medically nec...	2
Corrective Orthod...	15
Non-Medically Nec...	6
Non Medically Nec...	4
Non-Medically Nec...	2
occlusal Adjustments	3
Occlusal Guards -...	49
Occlusal guard - ...	16
Non-Medically Nec...	4
Cosmetic Orthodon...	28
Orthognatic Surgery	12
Ocalusal Adjustments	8
Cosmetic Orthodon...	8

only showing top 20 rows

Εικόνα 50: Orthodontia

BenefitName	Covering Plans
intial Placement ...	4
Initial Placement...	16
Denture Reline or...	23
Denture relining ...	32
Complete and Part...	5
Denture placement...	32
Denture repair - ...	32
Bridges - Child	52
Denture Adjustments	61
Immediate Dentures	61
Denture Reline an...	38
Initial Placement...	24
immediate Dentures	3
Addition of teeth...	16
One relining or r...	16
Bridges -Child	16
Denture adjustmen...	16
Dentures -Child	16
Initial Placement...	15
Denture repair -C...	16

only showing top 20 rows

Εικόνα 52:Dentures or Partials

BenefitName	Covering Plans
Gingivectomy and ...	32
Gingivectomy - Pe...	6

Εικόνα 53: Specialist Gum Procedures

BenefitName	Covering Plans
Dental Anesthesia...	32
Dental Anesthesia	40873
Dental Anesthesia...	112
Dental Anesthesia...	46
Dental Anesthesia...	16

Εικόνα 54:Dental Anesthesia

BenefitName	Covering Plans
Crown Build-up - ...	32
Cast metal, Stain...	16
Crown build-up -C...	16
Prefab Resin Crown	1
Prefabricated Cro...	16
Crowns - Child	20
Crowns - Adult	4

Εικόνα 55:Crowns

BenefitName	Covering Plans
Cosmetic Orthodon...	28
Cosmetic Orthodon...	8
Cosmetic Orthodon...	53
Cosmetic Orthodon...	16
Cosmetic Orthodon...	6
Cosmetic Orthodontia	6

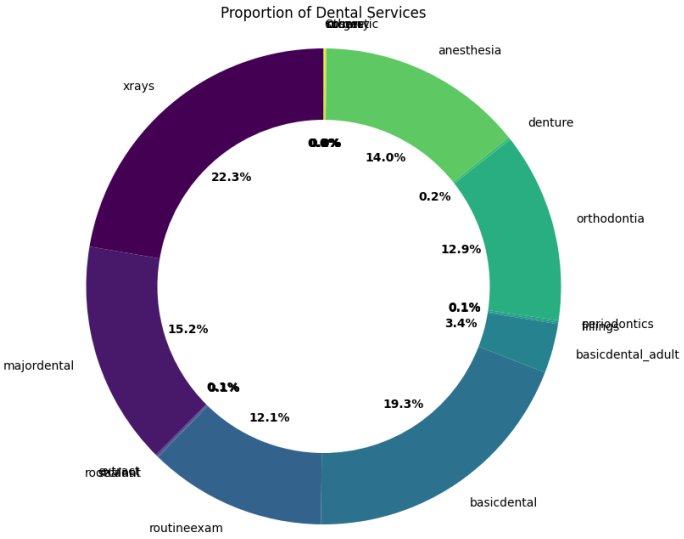
Εικόνα 56:Cosmetic Dentistry

BenefitName	Covering Plans
Osseous Surgery -...	32
Periodontal and O...	61
Complex Oral Surgery	65
Coimplex Oral Sur...	2
Orthognatic Treat...	4
Oral Surgery	69

Εικόνα 57: Dental Surgery

Οι ξεχωριστοί πίνακες που προέκυψαν αποτελούν μέρος ενός νέου υποσυνόλου, το οποίο συνιστά το data framework/πηγή για την απάντηση των δύο ερωτημάτων.

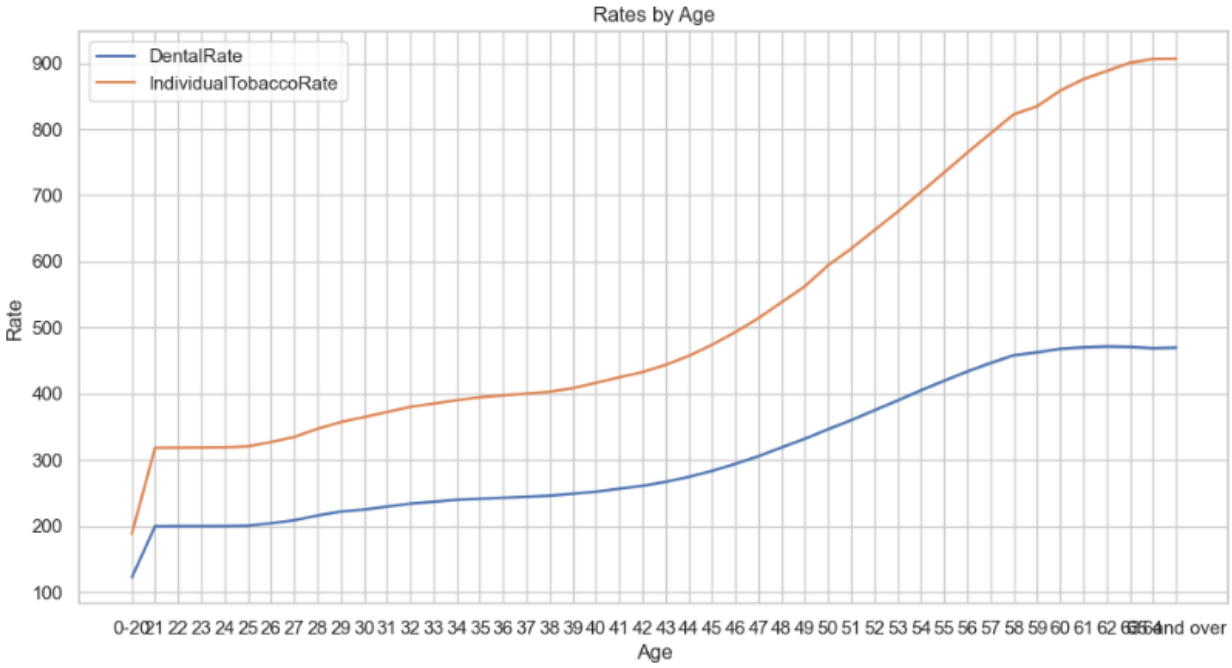
a) Η διερευνητική ανάλυση για την στοματική περίθαλψη εξήγαγε τις προτεραιότητες των πολιτών των Ηνωμένων Πολιτειών Αμερικής για την χρονική περίοδο 2014 – 2016. Στο παρατιθέμενο κυκλικό διάγραμμα παρατηρούμε ότι οι πολίτες προέβησαν κυρίως σε βασικές εξετάσεις και ακτινογραφίες με συνολικό ποσοστό μεγαλύτερο από 57%, ενώ οι αισθητικές οδοντιατρικές επεμβάσεις αποτελούν λιγότερο από 2%.



Εικόνα 58: Dental Services

Πηγή έμπνευσης για την υλοποίηση αποτελεί το Kaggle notebook, στο οποίο πραγματοποιείται η δημιουργία αυτού του υποσυνόλου με την προγραμματιστική γλώσσα R.

b) Το κάπνισμα αρκετά συχνά συνδέεται άμεσα με στοματικές ασθένειες, όπως περιοδοντίτιδα και καταστροφή της επιφάνειας της οδοντοστοιχίας, έχουν υψηλό κόστος για την θεραπεία τους. Στο παρακάτω διάγραμμα παρατηρούμε ότι τα ατομικά έξοδα για καπνικά είδη είναι σχεδόν διπλάσια από τις οδοντιατρικές εισφορές. Ιδιαίτερο ενδιαφέρον αποτελεί για την ηλικιακή ομάδα 40+, όπου τα οδοντιατρικά έξοδα εμφανίζουν τάση για σταθεροποίηση (40-45 ετών) με μικρή αύξηση της τάξης 1%-2% , ενώ τα καπνικά έξοδα παρουσιάζουν εκθετική αύξηση.



Εικόνα 59: Rate by age

Ο κώδικας για το 7.4.3 βρίσκεται στο παράρτημα.

Επίλογος

Καθώς η εποχή της επιστήμης των Big Data ανατέλλει, οι προεκτάσεις τους σε διάφορους τομείς διερευνώνται και εκτιμώνται συνεχώς. Αυτή η πτυχιακή είχε ως στόχο να εξετάσει τόσο τις θεωρητικές πτυχές των Μεγάλων Δεδομένων, όσο και τις πρακτικές μέσω της υλοποίησης εφαρμογής Data Analysis στον τομέα της υγειονομική ασφαλιστική αγορά των Ηνωμένων Πολιτείων Αμερική. Οι τεχνικές ανάλυσης Περιγραφή-Διερεύνηση-Πρόγνωση και το Apache Spark εξυπηρέτησαν στον εξαγωγή χρήσιμων πληροφοριών, οι οποίες απαντούν σε επιζήμια ερωτήματα είτε για τους υπεύθυνους πολιτικής στρατηγικής (κράτος, ιδιωτικές ασφαλιστικές εταιρίες) είτε για τους απλούς πολίτες. Τα διαγράμματα και οι εξατομικευμένοι πίνακες συμβάλλουν στην ταχύτερη και μαζικότερη κατανόηση των πολύπλοκων raw δεδομένων. Οι γνώσεις που δημιουργούνται μπορούν να συνεισφέρουν στον σχεδιασμό πιο δίκαιων ασφαλιστικών συμβολαίων, στη μείωση της απάτης και να οδηγήσουν σε καλύτερα μοντέλα αξιολόγησης κινδύνου. Η αξιοποίηση των Big Data τεχνολογιών, διαπερνά τα μεγέθη των δεδομένων και οι δυνατότητες άμεσης ανάλυσης δεδομένων σε πραγματικό χρόνο χρήζουν περαιτέρω εξερεύνηση.

Κώδικας εφαρμογής

7.3

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *
import os
import pandas as pd
import numpy as np
from pyspark.sql import functions as F

#Create Spark Session
spark = SparkSession.builder.appName("DescriptiveAnalysis")\
.config("spark.hadoop.hadoop.home.dir", "C:diplomatiki") \
.getOrCreate()

#ELT
#Load BenefitsCostSharing
df_costShare = spark.read.csv("raw_data/BenefitsCostSharing.csv", header=True, inferSchema=True)

#Remove empty or null rows
df_no_nulls = df_costShare.dropna()
for column in df_no_nulls.columns:
    df_no_nulls = df_no_nulls.filter(col(column) != "")
num_rows = df_costShare.count()

for column in df_costShare.columns:
    dtype = [dtype for name, dtype in df_costShare.dtypes if name == column][0]
    if dtype == 'double':
        df_costShare = df_costShare.withColumn(column, F.when(F.col(column).isNull(),
float(np.nan)).otherwise(F.col(column)))

total_records = df_costShare.count()
print(f"Total records in file: {total_records}")
unique_benefit_count = df_costShare.select("BenefitName").distinct().count()
print(f"Unique benefits present in the file: {unique_benefit_count}")

# Filter DataFrame for years 2014-2016
df_filtered_years = df_costShare.filter((df_costShare["BusinessYear"] >= 2014) &
(df_costShare["BusinessYear"] <= 2016))
#Generate summary statistics
summary_df_filtered_years = df_filtered_years.describe()
# Export the DataFrame to a Parquet file
summary_df_filtered_years.write.mode("overwrite").parquet("output_files/summary_filtered_years.parqu
et")

#Analyze all the Features
# Get list of all columns
all_columns = df_costShare.columns

# Select first 32 columns
selected_columns = all_columns[0:32]

# Loop through each selected column
for i, cn in enumerate(selected_columns):
print(i, cn)
df_costShare.select(cn).describe().show()
print("-" * 40)

# Group by 'StateCode' and then perform aggregations to describe 'BenefitName'
cost_per_state =
df_costShare.groupBy('StateCode').agg(F.countDistinct('BenefitName').alias('Distinct_Benefit_Count')
)
```

```
# Sort the DataFrame by 'StateCode'
cost_per_state = cost_per_state.orderBy('StateCode')

# Export the DataFrame to a Parquet file
cost_per_state.write.mode("overwrite").parquet("output_files/cost_per_state.parquet")

#Load Plan Attributes
df_planAttributes = spark.read.csv("raw_data/PlanAttributes.csv", header=True, inferSchema=True)

#Remove empty or null rows
df_no_nulls = df_planAttributes.dropna()
for column in df_no_nulls.columns:
df_no_nulls = df_no_nulls.filter(col(column) != "")

for column in df_planAttributes.columns:
dtype = [dtype for name, dtype in df_planAttributes.dtypes if name == column][0]
if dtype == 'double':
df_costShare = df_planAttributes.withColumn(column, F.when(F.col(column).isNull(),
float(np.nan)).otherwise(F.col(column)))
total_records = df_planAttributes.count()
print(f"Total records in file: {total_records}")
unique_planAttridute_count = df_planAttributes.select("BenefitPackageId").distinct().count()
print(f"Unique Plan Attributes present in the file: {unique_planAttridute_count}")


# Get list of all columns
all_columns = df_planAttributes.columns
# Select first 30 columns
selected_columns = all_columns[0:30]

# Loop through each selected column
for i, cn in enumerate(selected_columns):
    print(i, cn)
    df_costShare.select(cn).describe().show()
    print("-" * 40)
#Load Rate csv
df_rate=spark.read.csv("raw_data/Rate.csv", header=True, inferSchema=True)
#Remove empty or null rows
df_no_nulls = df_rate.dropna()
for column in df_no_nulls.columns:
    df_no_nulls = df_no_nulls.filter(col(column) != "")

for column in df_rate.columns:
    dtype = [dtype for name, dtype in df_rate.dtypes if name == column][0]
    if dtype == 'double':
        df_rate = df_rate.withColumn(column, F.when(F.col(column).isNull(),

float(np.nan)).otherwise(F.col(column)))
total_records = df_rate.count()

print(f"Total records in file: {total_records}")
unique_rate_count = df_rate.select("FederalTIN").distinct().count()
print(f"Unique rate present in the file: {unique_rate_count}")

# Get list of all columns
all_columns = df_rate.columns
# Select first 6 columns
selected_columns = all_columns[0:7]
for i, cn in enumerate(selected_columns):
    print(i, cn)
    df_costShare.select(cn).describe().show()
    print("-" * 40)
spark.stop()
```

7.4.1

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *
import os
import pandas as pd
import numpy as np
#Create Spark Session
spark = SparkSession.builder \
    .appName("ExploratoryAnalysis") \
    .getOrCreate()
#Plan Attriubutes CSV
df_planAttributes = spark.read.csv("raw_data/PlanAttributes.csv", header=True, inferSchema=True)
#Categorization of customers based on the market value
# Categories:
#Bronze - 60%, Silver - 70%, Gold - 80%, Platinum - 90%, Catastrophic - below 60% sold to individual
from pyspark.sql import functions as F
df_IssuerValue= df_planAttributes[df_planAttributes['IssuerActuarialValue'] != '']
df_IssuerValue = df_planAttributes.select('StateCode', 'IssuerId', 'ServiceAreaId',
'IssuerActuarialValue', 'MarketCoverage')
df_IssuerValue = df_IssuerValue.withColumn("IssuerActuarialValue",
F.regexp_replace(F.col("IssuerActuarialValue"), "%", ""))
df_IssuerValue = df_IssuerValue.withColumn("IssuerActuarialValue",
F.col("IssuerActuarialValue").cast("int"))

#Categories Gold,Bronze,Silver,Platinum
df_bronze = df_IssuerValue.filter((F.col("IssuerActuarialValue") >= 60) &
(F.col("IssuerActuarialValue") < 70))
df_bronze = df_bronze.groupBy("StateCode").agg(round(mean("IssuerActuarialValue"),
2).alias("Bronze"))

df_Silver = df_IssuerValue.filter((F.col("IssuerActuarialValue") >= 70) &
(F.col("IssuerActuarialValue") < 80))
df_Silver =
df_Silver.groupBy("StateCode").agg(round(mean("IssuerActuarialValue"),2).alias("Silver"))

df_Gold= df_IssuerValue.filter((F.col("IssuerActuarialValue") >= 80) &
(F.col("IssuerActuarialValue") < 90))
df_Gold= df_Gold.groupBy("StateCode").agg(round(mean("IssuerActuarialValue"),2).alias("Gold"))

df_Platinum= df_IssuerValue.filter((F.col("IssuerActuarialValue") >= 90) &
(F.col("IssuerActuarialValue") <= 100))
df_Platinum=
df_Platinum.groupBy("StateCode").agg(round(mean("IssuerActuarialValue"),2).alias("Platinum"))

df_catastrophic= df_IssuerValue.filter(col("IssuerActuarialValue") < 60)
df_catastrophic =
df_catastrophic.groupBy("StateCode").agg(round(mean("IssuerActuarialValue"),2).alias("Catastrophic")
)

#Comdine all the Catergories in order to make a new dataset
from functools import reduce
from pyspark.sql import DataFrame
def join_dfs(df1, df2, join_column_name="StateCode"):
    return df1.join(df2, on=join_column_name, how="outer")

all_tiers = [df_bronze, df_Silver,df_Gold,df_Platinum, df_catastrophic]
df_categories= reduce(join_dfs, all_tiers)

# Remove_List contains US regions like the Counry Colombia
remove_list = [
    "30751MT0560005", "30751MT0560006", "30751MT0560014", "30751MT0560015",
    "30751MT0560020", "30751MT0560021"]
```

```
# Filter out the rows with the values in the remove_list from the StateCode column
df_categories = df_categories.filter(~col("StateCode").isin(remove_list))
df_categories.show()

#Graphic Map of US
import geopandas as gpd
import matplotlib.pyplot as plt

#Join categories with map
df_usGeo=spark.read.csv("geo_data/US_GeoCode.csv", header=True, inferSchema=True)

# Rename the 'StateCode' column in df_usGeo to 'region'
df_usGeo = df_usGeo.withColumnRenamed("StateCode", "region")

# Rename the 'StateCode' column in df_categories to 'region'
df_categories = df_categories.withColumnRenamed("StateCode", "region")

# Group by 'region' and compute average longitude and latitude
statename = df_usGeo.groupBy("region").agg(avg("longitude").alias("long"),
avg("latitude").alias("lat"))

# Join the two DataFrames on the 'region' column
mapdata = df_categories.join(df_usGeo, on="region", how="left_outer")
mapdata_pd = mapdata.toPandas()

# Loading data and the US states shapefile
us_states_detailed = gpd.read_file("geo_data/tl_2022_us_state.shp")
mapdata_gdf = gpd.GeoDataFrame(mapdata_pd, geometry=gpd.points_from_xy(mapdata_pd.longitude,
mapdata_pd.latitude))

# Ensure the CRS matches before spatial join
# Set the initial CRS for mapdata_gdf
mapdata_gdf.crs = "EPSG:4326"

#Transformation in order to match the US states shapefile's CRS
mapdata_gdf = mapdata_gdf.to_crs(us_states_detailed.crs)
# Join the data using a spatial join
merged_data = gpd.sjoin(us_states_detailed, mapdata_gdf, how="left", predicate="intersects")

# Catastrophic Map
fig, ax = plt.subplots(figsize=(20, 15))  # Adjust the size here for a bigger map

merged_data.boundary.plot(ax=ax, color='black', linewidth=0.5)
merged_data.plot(column='Catastrophic', ax=ax, legend=True, cmap='Reds', legend_kwds={'label':
"Catastrophic Coverage"})

# Bounds to focus on the contiguous US
ax.set_xlim(-130, -60)
ax.set_ylim(20, 50)

# Centering the title
ax.set_title("Catastrophic Coverage in the US", fontsize=20, pad=20)

# Removing the axis for a cleaner look
ax.axis('off')
plt.savefig("graphs/Catastrophic_Coverage.png")
plt.show()

# Platinum Map
fig, ax = plt.subplots(figsize=(15, 10))  # Adjust the size here for a bigger map
merged_data.boundary.plot(ax=ax, color='black', linewidth=0.5)
merged_data.plot(column='Platinum', ax=ax, legend=True, cmap='Greens', legend_kwds={'label':
"Platinum Coverage"})
```

```
# Bounds to focus on the contiguous US
ax.set_xlim(-130, -60)
ax.set_ylim(20, 50)
# Centering the title
ax.set_title("Platinum Coverage in the US", fontsize=20, pad=20)

# Removing the axis for a cleaner look
ax.axis('off')
plt.savefig("graphs/Platinum_Coverage.png")
plt.show()
# Bronze Map
fig, ax = plt.subplots(figsize=(15, 10))  # Adjust the size here for a bigger map

merged_data.boundary.plot(ax=ax, color='black', linewidth=0.5)
merged_data.plot(column='Bronze', ax=ax, legend=True, cmap='Oranges', legend_kwds={'label': "Bronze
Coverage"})

# Bounds to focus on the contiguous US
ax.set_xlim(-130, -60)
ax.set_ylim(20, 50)

# Centering the title
ax.set_title("Bronze Coverage in the US", fontsize=20, pad=20)

# Removing the axis for a cleaner look
ax.axis('off')
plt.savefig("graphs/Bronze_Coverage.png")
plt.show()


# Gold Map
fig, ax = plt.subplots(figsize=(15, 10))  # Adjust the size here for a bigger map

merged_data.boundary.plot(ax=ax, color='black', linewidth=0.5)
merged_data.plot(column='Gold', ax=ax, legend=True, cmap='YlOrBr', legend_kwds={'label': "Gold
Coverage"})

# Bounds to focus on the contiguous US
ax.set_xlim(-130, -60)
ax.set_ylim(20, 50)

# Centering the title
ax.set_title("Gold  Coverage in the US", fontsize=20, pad=20)

# Removing the axis for a cleaner look
ax.axis('off')
plt.savefig("graphs/Gold_Coverage.png")
plt.show()
```

7.4.2a

```
#Rate by coverage level
#Rate csv
df_rate=spark.read.csv("raw_data/Rate.csv", header=True, inferSchema=True)
#Subset of 2015 individual rate
df_rate2015 = df_rate.filter((df_rate.BusinessYear == 2015) & (df_rate.Age != "Family Option") &
(df_rate.IndividualRate < 9000))
#df_rate2015.show(5)
# Subset the planAttributes and clean PlanId
df_benefit = df_planAttributes.select(df_planAttributes.columns[114],
df_planAttributes.columns[103],df_planAttributes.columns[161], df_planAttributes.columns[165],
df_planAttributes.columns[169])
df_benefit = df_benefit.withColumn("PlanId", F.substring(df_benefit.PlanId, 1, 14))
df_rate2015 = df_rate2015.withColumn("PlanId", df_rate2015.PlanId.cast("string"))
df_benefit.columns
# Clean MOOP columns

def clean_moop(column):
    return regexp_replace(regexp_replace(F.col(column), ",", ""), "\\$", "").cast("double")

df_benefit = df_benefit.withColumn("TEHBInnTier1IndividualMOOP",
clean_moop("TEHBInnTier1IndividualMOOP"))
df_benefit = df_benefit.withColumn("TEHBInnTier2IndividualMOOP",
clean_moop("TEHBInnTier2IndividualMOOP"))
df_benefit = df_benefit.withColumn("TEHBOutOfNetIndividualMOOP",
clean_moop("TEHBOutOfNetIndividualMOOP"))

# Aggregate the benefit table
benefittouse = df_benefit.groupBy("PlanId", "MetalLevel")\
.agg(
    F.mean("TEHBInnTier1IndividualMOOP").alias("innettier1moop"),
    F.mean("TEHBInnTier2IndividualMOOP").alias("innettier2moop"),
    F.mean("TEHBOutOfNetIndividualMOOP").alias("outnetmoop")
)
# Join the benefit to the rates
df_planrates = df_rate2015.join(benefittouse, "PlanId", "inner")

# Group by state and metal level, then aggregate
bystatecoverage = df_planrates.groupBy("StateCode", "MetalLevel").agg(
F.countDistinct("PlanId").alias("PlanOffered"),
F.mean("IndividualRate").alias("MeanIndRate"),
F.expr("percentile(IndividualRate, 0.5)").alias("MedianIndRate")  # median calculation
).orderBy(F.desc("PlanOffered"))



# Exclude dental coverage
medicalonly = bystatecoverage.filter((bystatecoverage.MetalLevel != "High") &
(bystatecoverage.MetalLevel != "Low"))

import matplotlib.pyplot as plt
import seaborn as sns

medicalonly = df_planrates

# Aggregate data
plan_counts = medicalonly.groupBy("StateCode", "MetalLevel").count().toPandas()

plt.figure(figsize=(10,6))
sns.barplot(x='StateCode', y='count', hue='MetalLevel', data=plan_counts, errorbar=None)
plt.title("Number of Medical Plans Offered By Coverage Levels")
plt.xlabel("Coverage Metal Level")
plt.ylabel("Number of Plans Offered")
```

```
plt.savefig("graphs/Num_Plans_Offered.png")
plt.show()

#Determination the plans offered  across various US states
from pyspark.sql import functions as F  # Make sure to import functions as F

df_benefit = spark.read.csv("raw_data/BenefitsCostSharing.csv", header=True, inferSchema=True)
num_fields = len(df_benefit.columns)
print(f"Number of fields: {num_fields}")

# Remove empty or null rows
df_no_nulls = df_benefit.dropna()
for column in df_no_nulls.columns:
    df_no_nulls = df_no_nulls.filter(F.col(column) != "")  # Use F.col here

num_rows = df_benefit.count()
print(f"The CSV file has {num_rows} rows.")

# Subset for planRateBenefit
df_planRateBenefit = df_rate\
.filter((df_rate["IndividualRate"] < 9999) & (df_rate["Age"] != "Family Option"))\
.select("BusinessYear", "StateCode", "IssuerId", "PlanId", "Age", "IndividualRate",
"IndividualTobaccoRate")

#df_planRateBenefit.show()

df_StateCarrier = df_planRateBenefit.groupBy("StateCode", "IssuerId", "PlanId")\
.agg(count("StateCode").alias("count"))

df_StateCarrier.show()

#Graph
dataBenefitsName = df_benefit.select("BusinessYear", "IssuerId", "StandardComponentId", "StateCode",
"BenefitName")
dataBenefitsName.show(5)
```

7.4.2b

```
# Subset for VariousRatesPerState
VariousRatesPerState = df_rate\
.filter(
    (df_rate["Couple"].isNotNull()) |
    (df_rate["PrimarySubscriberAndOneDependent"].isNotNull()) |
    (df_rate["PrimarySubscriberAndTwoDependents"].isNotNull()) |
    (df_rate["PrimarySubscriberAndThreeOrMoreDependents"].isNotNull()) |
    (df_rate["CoupleAndOneDependent"].isNotNull()) |
    (df_rate["CoupleAndTwoDependents"].isNotNull()) |
    (df_rate["CoupleAndThreeOrMoreDependents"].isNotNull()))\
.select("StateCode", "IndividualRate", "Couple", "PrimarySubscriberAndOneDependent",
         "PrimarySubscriberAndTwoDependents", "PrimarySubscriberAndThreeOrMoreDependents",
         "CoupleAndOneDependent", "CoupleAndTwoDependents", "CoupleAndThreeOrMoreDependents")
#VariousRatesPerState.show()


TotalRatePerState = VariousRatesPerState.groupBy(
    "StateCode", "IndividualRate", "Couple", "PrimarySubscriberAndOneDependent",
    "PrimarySubscriberAndTwoDependents", "PrimarySubscriberAndThreeOrMoreDependents",
    "CoupleAndOneDependent", "CoupleAndTwoDependents", "CoupleAndThreeOrMoreDependents")\
.agg(count("StateCode").alias("count"))

TotalRatePerState.show()

# Melt operation
melted_columns = ["IndividualRate", "Couple", "PrimarySubscriberAndOneDependent",
                  "PrimarySubscriberAndTwoDependents", "PrimarySubscriberAndThreeOrMoreDependents",
                  "CoupleAndOneDependent", "CoupleAndTwoDependents",
"CoupleAndThreeOrMoreDependents"]
df_melt = TotalRatePerState.select("StateCode", F.expr('stack(' + str(len(melted_columns)) + ', ' +
', '.join(["'" + x + "', " + x for x in melted_columns]) + ') as (RateType, RateValue)'))
df_melt.show(5)

# PlanRatevsAge subset
PlanRatevsAge = df_rate.filter(
    (df_rate["IndividualRate"] < 9999) &
    (df_rate["StateCode"].isNotNull()) &
    (df_rate["Age"].isNotNull()) &
    (df_rate["IndividualRate"].isNotNull())).select("StateCode", "Age", "IndividualRate")


# Filter data for different age groups and drop duplicates based on 'IndividualRate'
TotRatevsAge1 = PlanRatevsAge.filter(PlanRatevsAge["Age"] <= 22).dropDuplicates(["IndividualRate"])
TotRatevsAge2 = PlanRatevsAge.filter((PlanRatevsAge["Age"] >= 23) & (PlanRatevsAge["Age"] <=
40)).dropDuplicates(["IndividualRate"])
TotRatevsAge3 = PlanRatevsAge.filter((PlanRatevsAge["Age"] >= 41) & (PlanRatevsAge["Age"] <=
59)).dropDuplicates(["IndividualRate"])
TotRatevsAge4 = PlanRatevsAge.filter(PlanRatevsAge["Age"] == "Family
Option").dropDuplicates(["IndividualRate"])
TotRatevsAge5 = PlanRatevsAge.filter((PlanRatevsAge["Age"] >= 60) & (PlanRatevsAge["Age"] <=
64)).dropDuplicates(["IndividualRate"])
TotRatevsAge6 = PlanRatevsAge.filter(PlanRatevsAge["Age"] == "65 and
over").dropDuplicates(["IndividualRate"])

# Convert a PySpark DataFrame to Pandas for plotting
pdf = df_melt.toPandas()

# Sort the DataFrame by 'StateCode'
pdf = pdf.sort_values(by='StateCode')

# Split the state codes into two halves
unique_states = sorted(pdf['StateCode'].unique())
```

```
half_length = len(unique_states) // 2

first_half_states = unique_states[:half_length]
second_half_states = unique_states[half_length:]
plt.xticks(rotation=90)
plt.legend(title='Rate Plans')
plt.tight_layout()
plt.savefig("graphs/Rate_Plans_part2.png")
plt.show()

# Plot the first half
plt.figure(figsize=(15, 8))
sns.barplot(data=pdf[pdf['StateCode'].isin(first_half_states)], x='StateCode', y='RateValue',
hue='RateType', order=first_half_states)
plt.title("Plot of Health insurance Plan rates")
plt.ylabel("Plan rates")
plt.xlabel("State Code")
plt.xticks(rotation=90)
plt.legend(title='Rate Plans')
plt.tight_layout()
plt.savefig("graphs/Rate_Plans_part1.png")
plt.show()
# Plot the second half
plt.figure(figsize=(15, 8))
sns.barplot(data=pdf[pdf['StateCode'].isin(second_half_states)], x='StateCode', y='RateValue',
hue='RateType', order=second_half_states)
plt.title("Plot of Health insurance Plan rates")
plt.ylabel("Plan rates")
plt.xlabel("State Code")
plt.legend(title='Rate Plans')
plt.tight_layout()
plt.savefig("graphs/Rate_Plans_part2.png")
plt.show()
```

7.4.2c

```
#Graphs per age
pdf1 = TotRatevsAge1.toPandas()
pdf2 = TotRatevsAge2.toPandas()
pdf3 = TotRatevsAge3.toPandas()
pdf4 = TotRatevsAge4.toPandas()
pdf5 = TotRatevsAge5.toPandas()
pdf6 = TotRatevsAge6.toPandas()
 #Function for creating all the plots be group age
def plot_rate_vs_age(data, title):
    plt.figure(figsize=(50, 10))
    sns.barplot(data=data, x='StateCode', y='IndividualRate', hue='Age')
    plt.ylabel("Individual Rates")
    plt.xlabel("State Code")
    plt.title(title)
    plt.xticks(rotation=90)
    plt.legend(title='Age')
    plt.tight_layout()
    plt.show()

#Age 0-22
plot_rate_vs_age(pdf1, "Plan Rate vs Age across various US States")

#Age 23-40
plot_rate_vs_age(pdf2, "Plan Rate vs Age across various US States")

#Age 41 to 59
plot_rate_vs_age(pdf3, "Plan Rate vs Age across various US States")

#Age 60 to 64
plot_rate_vs_age(pdf5, "Plan Rate vs Age across various US States")
```

```
#Age 65 and over
plot_rate_vs_age(pdf6, "Plan Rate vs Age across various US States")

#Family edition
plot_rate_vs_age(pdf4, "Plan Rate vs Age across various US States")
```

7.4.2d

```
#Premium Distribution by Coverage Levels in 2015
#Due to limit memory I only presented the 20% of the data
sample_fraction = 0.2
planrates_sampled = df_planrates.sample(False, sample_fraction)

# Identify columns of type TimestampType
timestamp_cols = [f.name for f in planrates_sampled.schema.fields if isinstance(f.dataType,
TimestampType)]

# Convert to StringType
for col in timestamp_cols:
    planrates_sampled = planrates_sampled.withColumn(col, planrates_sampled[col].cast(StringType()))

# Convert the Spark dataframe to Pandas
planrates_pd = planrates_sampled.toPandas()

# Convert the string date columns back to datetime64[ns] in Pandas
for col in timestamp_cols:
    planrates_pd[col] = planrates_pd[col].astype('datetime64[ns]')

plt.figure(figsize=(10,6))
sns.boxplot(x='MetalLevel', y='IndividualRate', data=planrates_pd)
sns.pointplot(x='MetalLevel', y='IndividualRate', data=planrates_pd, errorbar=None,
estimator=np.mean, color='darkred')
plt.title("Individual Rate Distribution of Coverage")
plt.savefig("graphs/Individual_Rate.png")
plt.show()
```

7.4.3

```
#DentalCare System
#Prepare Plan Attributes with Dental data
#Keep only the columns with dental data
selected_columns = (
    df_planAttributes.columns[0:4] +
    [df_planAttributes.columns[5], df_planAttributes.columns[6]] +
    df_planAttributes.columns[9:15] +
    df_planAttributes.columns[16:21] +
    df_planAttributes.columns[24:28] +
    df_planAttributes.columns[29:50] +
    df_planAttributes.columns[54:125]
)
df_cleanPlan=df_planAttributes.select(selected_columns)

#Remove
rm_cols=["OutOfCountryCoverageDescription","OutOfServiceAreaCoverageDescription",
"DiseaseManagementProgramsOffered","PlanEffictiveDate", "PlanExpirationDate",
"OutOfCountryCoverage",
        "MedicalDrugMaximumOutofPocketIntegrated",
"FirstTierUtilization","SecondTierUtilization","MEHBInnTier1IndividualMOOP","MEHBInnTier1FamilyMOOP"
,"MEHBInnTier2IndividualMOOP",

"MEHBInnTier2FamilyMOOP","MEHBOutOfNetIndividualMOOP","MEHBOutOfNetFamilyMOOP","MEHBCombInnOonIndivi
dualMOOP",

"MEHBCombInnOonFamilyMOOP","DEHBInnTier1IndividualMOOP","DEHBInnTier1FamilyMOOP","DEHBInnTier2Indivi
dualMOOP","DEHBInnTier2FamilyMOOP","DEHBOutOfNetIndividualMOOP",

"DEHBOutOfNetFamilyMOOP","DEHBCombInnOonIndividualMOOP","DEHBCombInnOonFamilyMOOP","TEHBInnTier1Indi
vidualMOOP",

"TEHBInnTier1FamilyMOOP","TEHBInnTier2IndividualMOOP","TEHBInnTier2FamilyMOOP","TEHBOutOfNetIndividu
alMOOP",

"TEHBOutOfNetFamilyMOOP","TEHBCombInnOonIndividualMOOP","TEHBCombInnOonFamilyMOOP","MEHBDedInnTier1I
ndividual",

"MEHBDedInnTier1Family","MEHBDedInnTier1Coinsurance","MEHBDedInnTier2Individual","MEHBDedInnTier2Fam
ily","MEHBDedInnTier2Coinsurance",

"MEHBDedOutOfNetIndividual","MEHBDedOutOfNetFamily","MEHBDedCombInnOonIndividual","MEHBDedCombInnOon
Family","DEHBDedInnTier1Individual",
         "DEHBDedInnTier1Family","DEHBDedInnTier1Coinsurance",
"DEHBDedInnTier2Individual","DEHBDedInnTier2Family",

"DEHBDedInnTier2Coinsurance","DEHBDedOutOfNetIndividual","DEHBDedOutOfNetFamily","DEHBDedCombInnOonI
ndividual",

"DEHBDedCombInnOonFamily","TEHBDedInnTier1Individual","TEHBDedInnTier1Family","TEHBDedInnTier1Coinsu
rance",

"TEHBDedInnTier2Individual","TEHBDedInnTier2Family","TEHBDedInnTier2Coinsurance","TEHBDedOutOfNetInd
ividual",

"TEHBDedOutOfNetFamily","TEHBDedCombInnOonIndividual","TEHBDedCombInnOonFamily","SBCHavingaBabyDeduc
tible",

"SBCHavingaBabyCopayment","SBCHavingaBabyCoinsurance","SBCHavingaBabyLimit","SBCHavingDiabetesDeduct
ible",
        "SBCHavingDiabetesCopayment","SBCHavingDiabetesCoinsurance","SBCHavingDiabetesLimit",
        "FormularyId", "MedicalDrugDeductiblesIntegrated", "InpatientCopaymentMaximumDays",
        "SpecialtyDrugMaximumCoinsurance", "IsNoticeRequiredForPregnancy"]
```

```
df_cleanPlan=df_planAttributes.drop(*rm_cols)

df_cleanPlan=df_cleanPlan.withColumn("plan_id_short", F.expr("substring(PlanId, 1, length(PlanId) -
3)"))
# Filter to get dental_only_list and not_dental_only_list
dental_only_list = df_cleanPlan.filter(df_cleanPlan.DentalOnlyPlan ==
"Yes").select("PlanId",df_cleanPlan.columns[2], df_cleanPlan.columns[3], df_cleanPlan.columns[4],
df_cleanPlan.columns[11], df_cleanPlan.columns[33])
# Count unique PlanId
print(dental_only_list.select("PlanId").distinct().count())
#Prepare 'Benefits of Cost Sharing' with Dental data
#Load BenefitsCostSharing
df_costShare = spark.read.csv("raw_data/BenefitsCostSharing.csv", header=True, inferSchema=True)

column_positions = {column: index for index, column in enumerate(df_costShare.columns)}
print(column_positions)
all_columns = df_costShare.columns
select_bnf_cols =df_costShare.columns[0:8] + df_costShare.columns[9:31]
df_cleanbnf=df_costShare.select(select_bnf_cols)
df_cleanbnf.printSchema()

#Linked benefits to dentalcare

#General examination

# Add a new column "routineexam" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    "routineexam",
    F.when(
        (F.col("BenefitName").rlike("Prophylaxis|Oral Exam|Cleaning|Dental Check-Up for Children"))
&
        (F.col("IsCovered") == "Covered"),
        1
    ).otherwise(None)
)

# Summary table
general_exams = df_cleanbnf.filter(F.col("routineexam") > 0).groupBy("BenefitName").count()
general_exams = general_exams.withColumnRenamed("count", "Covering Plans")
general_exams.show()

#Basic Denatalcare

# Add a new column "basicdental" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    "basicdental",
    F.when(
        (F.col("BenefitName").rlike("Basic Dental Care|Routine Dental Services")) &
        (F.col("IsCovered") == "Covered"),
        1
    ).otherwise(
        F.when(
            (F.col("BenefitName").rlike("Non-Routine")) &
            (F.col("IsCovered") == "Covered"),
            0
        ).otherwise(None)
    )
)

# Summary table
basic_dental = df_cleanbnf.filter(F.col("basicdental") > 0).groupBy("BenefitName").count()
basic_dental = basic_dental.withColumnRenamed("count", "Covering Plans")
basic_dental.show()
```

```
#Basic Dental Adult

if 'basicdental_adult' not in df_cleanbnf.columns:
    df_cleanbnf = df_cleanbnf.withColumn('basicdental_adult', F.lit(0))

# Your existing code
df_cleanbnf = df_cleanbnf.withColumn(
    "basicdental_adult",
    F.when(
        (F.col("BenefitName").like("%Basic Dental Care - Adult%")) & (F.col("IsCovered") ==
"Covered"),
        1
    ).otherwise(F.col("basicdental_adult"))
)
# Summary table
basicdental_adult = df_cleanbnf.filter(F.col("basicdental_adult") >
0).groupBy("BenefitName").count()
basicdental_adult = basic_dental.withColumnRenamed("count", "Covering Plans")
basicdental_adult.show()

#Major Dentalcare

# Add a new column "majordental" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    "majordental",
    F.when(
        (F.col("BenefitName").rlike("Major Dental Care|Non-Routine")) &
        (F.col("IsCovered") == "Covered"),
        1
    ).otherwise(None)
)

# Summary table
major_dental = df_cleanbnf.filter(F.col("majordental") > 0).groupBy("BenefitName").count()
major_dental = major_dental.withColumnRenamed("count", "Covering Plans")
major_dental.show()
#X-Rays
# Add a new column "xrays" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    "xrays",
    F.when(
        (F.col("BenefitName").rlike("X-ray|X-Ray|X ray|X Ray")) &
        (F.col("IsCovered") == "Covered"),
        1
    ).otherwise(
        F.when(
            F.col("BenefitName") == "X-rays and Diagnostic Imaging",
            0
        ).otherwise(None)
    )
)

# Summary table
xrays = df_cleanbnf.filter(F.col("xrays") > 0).groupBy("BenefitName").count()
xrays = xrays.withColumnRenamed("count", "Covering Plans")
xrays.show()

#Fluoride Treatment

# Add a new column "fluoride" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    'fluoride',
    F.when(
```

```
        (F.col('BenefitName').rlike('Flouride|Fluoride|fluoride')) & (F.col('IsCovered') ==
'Covered'),
        1
    ).otherwise(None)
)

# Summary table
fluoride = df_cleanbnf.filter(F.col("fluoride") > 0).groupBy("BenefitName").count()
fluoride = fluoride.withColumnRenamed("count", "Covering Plans")
fluoride.show()

#Extranctions

# Add a new column "extract" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    'extract',
    F.when(
        (F.col('BenefitName').rlike('Extraction|extraction|removal|Removal')) & (F.col('IsCovered')
== 'Covered') & (F.col('BenefitName') != 'Breast Implant Removal'),
        1
    ).otherwise(None)
)
# Summary table
extract = df_cleanbnf.filter(F.col("extract") > 0).groupBy("BenefitName").count()
extract = extract.withColumnRenamed("count", "Covering Plans")
extract.show()

#Root Canals

# Add a new column "rootcanal" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    'rootcanal',
    F.when(
        (F.col('BenefitName').rlike('Root Canal|root canal|Root canal|Endodonti')) &
(F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
rootcanal = df_cleanbnf.filter(F.col('rootcanal') > 0).groupBy('BenefitName').count()
rootcanal = rootcanal.withColumnRenamed('count', 'Covering Plans')
rootcanal.show()

#Sealants

# Add a new column "sealant" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    'sealant',
    F.when(
        (F.col('BenefitName').rlike('Sealant|sealant')) & (F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
sealant = df_cleanbnf.filter(F.col('sealant') > 0).groupBy('BenefitName').count()
sealant = sealant.withColumnRenamed('count', 'Covering Plans')
sealant.show()

# Fillings

# Add a new column "fillings" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    'fillings',
    F.when(
```

```
        (F.col('BenefitName').rlike('restorative|Restorative|Amalgam|Filling')) &
(F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
fillings = df_cleanbnf.filter(F.col('fillings') > 0).groupBy('BenefitName').count()
fillings = fillings.withColumnRenamed('count', 'Covering Plans')
fillings.show()

# Periodontics

# Add a new column "periodontics" based on conditions
df_cleanbnf = df_cleanbnf.withColumn(
    'periodontics',
    F.when(
        (F.col('BenefitName').rlike('Periodont|periodont|Scaling|Planing')) & (F.col('IsCovered') ==
'Covered'),
        1
    ).otherwise(None)
)
# Summary table
periodontics = df_cleanbnf.filter(F.col('periodontics') > 0).groupBy('BenefitName').count()
periodontics = periodontics.withColumnRenamed('count', 'Covering Plans')
periodontics.show()

#Orthodontia

df_cleanbnf = df_cleanbnf.withColumn(
    'orthodontia',
    F.when(
        (F.col('BenefitName').rlike('Orthognatic|orthodont|Orthodont|Occlusal|Ocalusal|occlusal')) &
(F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
orthodontia = df_cleanbnf.filter(F.col('orthodontia') > 0).groupBy('BenefitName').count()
orthodontia = orthodontia.withColumnRenamed('count', 'Covering Plans')
orthodontia.show()

#Dentures or Partials

df_cleanbnf = df_cleanbnf.withColumn(
    'denture',
    F.when(
        (F.col('BenefitName').rlike('Denture|denture|Bridge|Dentiures')) & (F.col('IsCovered') ==
'Covered'),
        1
    ).otherwise(None)
)
# Summary table
denture = df_cleanbnf.filter(F.col('denture') > 0).groupBy('BenefitName').count()
denture = denture.withColumnRenamed('count', 'Covering Plans')
denture.show()
#Specialist Gum Procedures
df_cleanbnf = df_cleanbnf.withColumn(
    'gums',
    F.when(
        (F.col('BenefitName').rlike('Gingivec')) & (F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
```

```
gums = df_cleanbnf.filter(F.col('gums') > 0).groupBy('BenefitName').count()
gums = gums.withColumnRenamed('count', 'Covering Plans')
gums.show()

#Dental Anesthesia

df_cleanbnf = df_cleanbnf.withColumn(
    'anesthesia',
    F.when(
        (F.col('BenefitName').rlike('Dental Anesthesia')) & (F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
anesthesia = df_cleanbnf.filter(F.col('anesthesia') > 0).groupBy('BenefitName').count()
anesthesia = anesthesia.withColumnRenamed('count', 'Covering Plans')
anesthesia.show()

#Crowns

df_cleanbnf = df_cleanbnf.withColumn(
    'crown',
    F.when(
        (F.col('BenefitName').rlike('Crown')) & (F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
crown = df_cleanbnf.filter(F.col('crown') > 0).groupBy('BenefitName').count()
crown = crown.withColumnRenamed('count', 'Covering Plans')
crown.show()

#Cosmetic Dentistry

df_cleanbnf = df_cleanbnf.withColumn(
    'cosmetic',
    F.when(
        (F.col('BenefitName').rlike('Cosmetic Ortho')) & (F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
cosmetic = df_cleanbnf.filter(F.col('cosmetic') > 0).groupBy('BenefitName').count()
cosmetic = cosmetic.withColumnRenamed('count', 'Covering Plans')
cosmetic.show()

#Dental Surgery

df_cleanbnf = df_cleanbnf.withColumn(
    'surgery',
    F.when(
        (F.col('BenefitName').rlike('Oral Surgery|Orthognatic Treatment/Surgery|Osseous Surgery')) &
(F.col('IsCovered') == 'Covered'),
        1
    ).otherwise(None)
)
# Summary table
surgery = df_cleanbnf.filter(F.col('surgery') > 0).groupBy('BenefitName').count()
surgery = surgery.withColumnRenamed('count', 'Covering Plans')
surgery.show()

#Dental benefits summary
ben_summary = df_cleanbnf.groupBy('PlanId', 'IssuerId').agg(
    F.sum('xrays').alias('xrays'),
```

```
    F.sum('fluoride').alias('fluoride'),
    F.sum('majordental').alias('majordental'),
    F.sum('extract').alias('extract'),
    F.sum('rootcanal').alias('rootcanal'),
    F.sum('sealant').alias('sealant'),
    F.sum('routineexam').alias('routineexam'),
    F.sum('basicdental').alias('basicdental'),
    F.sum('basicdental_adult').alias('basicdental_adult'),
    F.sum('fillings').alias('fillings'),
    F.sum('periodontics').alias('periodontics'),
    F.sum('orthodontia').alias('orthodontia'),
    F.sum('denture').alias('denture'),
    F.sum('gums').alias('gums'),
    F.sum('anesthesia').alias('anesthesia'),
    F.sum('crown').alias('crown'),
    F.sum('cosmetic').alias('cosmetic'),
    F.sum('surgery').alias('surgery')
)
ben_summary.show(5)

#Create Dental Data file
# Create a new column 'plan_id_short' using substring function
df_planAttributes_filtered = df_planAttributes.withColumn("plan_id_short",
substring(F.col("PlanId"), 1, -4))

# Define the join keys
join_keys = ["PlanId", "IssuerId"]

# Perform the merge using the join keys
dental_data = df_planAttributes_filtered.join(ben_summary, on=join_keys, how="inner")

# Show the merged data
output_path = "output_files/dental_data.csv"
dental_data.write.csv(output_path, header=True, mode="overwrite")

#Rates are the monthly payments regardless of whether you go see your dentist or not.
rate=spark.read.csv("raw_data/Rate.csv", header=True, inferSchema=True)
```

7.3.4a

```
 # Select required columns
rate = rate.select([rate.columns[i] for i in [0, 1, 2, 3, 5, 7, 8, 9, 10, 11, 12, 13,
                                              14, 15, 16, 17, 18, 19, 20, 21, 22]])

# Filter out invalid rates
rate = rate.filter((rate.IndividualRate < 1000) & (rate.IndividualRate > 0))

# Merge with dental_data
rate = rate.join(dental_data.select("plan_id_short", "DentalOnlyPlan"),
                 rate.PlanId == dental_data.plan_id_short, "left")
#Proportion of the Dental Services
ben_summary_pd = ben_summary.toPandas()

# Summing up each column
numeric_cols = ben_summary_pd.select_dtypes(include=['number']).columns
sums = ben_summary_pd[numeric_cols].sum()


# Removing non-numeric fields if any
sums = sums[sums.index.isin(['xrays', 'fluoride', 'majordental', 'extract', 'rootcanal', 'sealant',
                             'routineexam', 'basicdental', 'basicdental_adult', 'fillings',
'periodontics',
                             'orthodontia', 'denture', 'gums', 'anesthesia', 'crown', 'cosmetic',
'surgery'])]

# Combine small categories into 'Others'
threshold = 100  # Define your own threshold
mask = sums <= threshold
tail = sums.loc[mask]
sums = sums.loc[~mask]
sums['Others'] = tail.sum()

# Define colors
colors = plt.cm.viridis(np.linspace(0, 1, len(sums)))

# Create pie chart
fig, ax = plt.subplots(figsize=(10, 8))
wedges, texts, autotexts = ax.pie(sums, labels=sums.index, autopct='%1.1f%%', colors=colors,
startangle=90, wedgeprops=dict(width=0.3))

# Improve readability
ax.legend(wedges, sums.index, title='Categories', loc='best', bbox_to_anchor=(1, 0, 0.5, 1))
plt.setp(autotexts, size=10, weight='bold')

# Add title
ax.set_title('Proportion of Dental Services')

# Ensure the pie chart is a circle
ax.axis('equal')

# Show the pie chart
plt.savefig("graphs/Dental Services.png")
plt.show()
```

7.4.3b

```
#Tobacco&AgeVsdDentalRate

# Filter out 'Family Option' from the 'Age' column
filtered_rate = rate.filter(F.col("Age") != "Family Option")
# Select only the columns you're going to use
selected_columns = ["Age", "IndividualRate","IndividualTobaccoRate"]
filtered_rate = filtered_rate.select(*selected_columns)
# Convert the Spark DataFrame to a Pandas DataFrame
filtered_rate_pd = filtered_rate.toPandas()

# Set the aesthetics for the plots
sns.set(style="whitegrid")

# Create a figure and axis objects
fig, ax = plt.subplots(figsize=(12, 6))

# Plot IndividualRate against Age
sns.lineplot(x='Age', y='IndividualRate', data=filtered_rate_pd, label='DentalRate', ax=ax)

# Plot IndividualTobaccoRate against Age
sns.lineplot(x='Age', y='IndividualTobaccoRate', data=filtered_rate_pd,
label='IndividualTobaccoRate', ax=ax)

# Set the title and labels for the plot
ax.set_title('Rates by Age')
ax.set_xlabel('Age')
ax.set_ylabel('Rate')

# Display the legend
ax.legend()

# Show the plot
plt.show()
```

Συντομογραφίες

BD	Big Data
DB	Βάση δεδομένων
RDBMS	Σχεσιακή βάση δεδομένων
NoSQL	Not Only SQL
NRDBMS	Μη-Σχεσιακή βάση δεδομένων
RDD	Ανθεκτικό Κατανεμημένο Σύνολο Δεδομένων
ML	Machine Learning

Ευρετήριο Διαγραμμάτων

Ευρετήριο Εικόνων

Βιβλιογραφία

[1] Manning P. (2013): Big Data in History (Palagrave Pivot) 2013th Edition, Palgrave Pivot

[2] Zennaro Marco (December 2016): "Developing the ICU ecosystem to harness IoTs" Intro to Big Data https://www.itu.int/en/ITU-D/Regional-Presence/AsiaPacific/SiteAssets/Pages/Events/2016/Dec-2016-IoT/IoTtraining/BigData%20-Zennaro.pdf

[3] Marr Benard (Febuary 25,2015): A brief history of big data everyone should read https://www.weforum.org/agenda/2015/02/a-brief-history-of-big-data-everyone-should-read/

[4] Maria Cristina Mariani, Osei Kofi Tweneboah, Maria Pia Beccar-Varela (2021): Data Science in theory and practice 2nd Edition, Willey

[5] Indicate Team: What is Data Volumehttps://www.indicative.com/resource/volume-of-data/

[6] Indicate Team: What is Data Velocity https://www.indicative.com/resource/data-velocity/

[7] Indicate Team: What is Data Variety https://www.indicative.com/resource/data-variety/

[8] Indicate Team: What is Data Veracityhttps://www.indicative.com/resource/data-veracity/

[9] Smallcombe Mark (August 20,2020): The 7Vs of Big Datahttps://www.integrate.io/blog/7-vs-big-data/#volume

[10] Oracle: What is Big Data? https://www.oracle.com/a/ocom/docs/what-is-big-data-ebook-4421383.pdf

[11] Jacquelyn Bulao (May 2,2022): How much data is created every day in 2022? https://techjury.net/blog/how-much-data-is-created-every-day/#gref

[12] Jens-Petter Sandvik, Katin Franke, Habtanum Abie, Ande Arnes (2022): Quantifying data volatility of IoT forensics with examples from Contik OS, 9th Annual DFRWS Europe Conference

[13] Tableau: What Is Data Visualization?Definition,Examples,And Learning Resourceshttps://www.tableau.com/learn/articles/data-visualization#:~:text=Data%20visualization%20is%20the%20graphical,outliers%2C%20and%20patterns%20in%20data.

[14] Rock Content (June 1,2020): What is Big Data Visualization? https://rockcontent.com/blog/big-data-visualization/#:~:text=Big%20Data%20visualization%20describes%20data,easy%20to%20understand%20and%20interpret.

[15] Enterprice Big Data Framewort (January 9,2019): Three different data structureshttps://www.bigdataframework.org/data-types-structured-vs-unstructured-data/

[16]Cyntia Harvey (August 9,2018): Big Data Pros and Conshttps://www.datamation.com/big-data/big-data-pros-and-cons/

[17] Oracle: What is NoSQL?https://www.oracle.com/database/nosql/what-is-nosql/

[18]TechTerms (August 27,2013): NoSQLhttps://techterms.com/definition/nosql

[19]GeekforGeeks (July 7,2022): Types of NoSQL Databases https://www.geeksforgeeks.org/types-of-nosql-databases/

[20]IBM Cloud Education (August 6,2019): NoSQL Databases https://www.ibm.com/cloud/learn/nosql-databases

[21]MongoDB:Types of NoSQL Databases https://www.mongodb.com/scale/types-of-nosql-databases

[22]https://www.geeksforgeeks.org/features-of-cassandra/?ref=gcse

[23] Jeff Carpenter, Eben Hewitt (April 2020): Cassandra: The Definitive Guide, 3rd Edition, O'Reilly Media

[24] C.Y. Kan (December 23, 2014): Cassandra Data Modeling and Analysis, Packt Publishing

[25]GeekforGeeks(May 10, 2022): Features of Cassandrahttps://cassandra.apache.org/doc/latest/cassandra/cql/

[26]Apache Cassandra : The Cassandra Query Language (CQL) https://www.programsbuzz.com/article/apache-spark-history

[27]Shiksha Dahiya (July 21, 2021): Apache Spark History https://sql-stack.com/2018/12/07/apache-sparks-history/

[28]LFEIOCK (December 7, 2018):Apache Spark's Historyhttps://www.knowledgehut.com/tutorials/apache-spark-tutorial/apache-spark-evolution

[29] Jule S. Damji, Brooke Wenig, Tathagata Das, Denny Lee (July 16,2020): Learning Spark Lightning-Fast Data Analytics 2nd Edition, O'Reilly Media

[30] James Peebles(July 11, 2022): Apache Spark Ecosystem and Spark Components https://www.projectpro.io/article/apache-spark-ecosystem-and-spark-components/219#mcetoc_1faffpsntj

[31]Big Data Solutions Blog(January 4,2019)**:** Limitations of Apache Sparkhttps://bigdatasolutionss.blogspot.com/2019/01/limitations-of-apache-spark.html

[32]Talend : Big Data in Government https://www.talend.com/resources/big-data-in-government/#:~:text=Big%20data%20in%20government%20is%20the%20influx%20of%20data%20from,and%20manage%20the%20public%20sector.

[33] Mashooque A. Memon1, Safeeullah Soomro2, Awais K. Jumani3 and Muneer A. Kartio3, Big Data Analytics and Its Applications

[34] Ruth Dmonte, Asher Dmello (January 1,2017): Big Data in Sports (vol.6), International Journal of Engineering Research &Techonology

[35]Mallika Rangaia (December 26,2020):Applications of Big Data in Sports Industry https://www.analyticssteps.com/blogs/applications-big-data-sports-industry

[36] Maris Mirovic, Mario Milicevic, Ines Obradovic (March 2018): Big Data in the Maritime Industry

[37] Sjoukje A. Osinga, Dilli Paudel, Spiros A. Mauzakitis, Ioannis N. Athanasiadis (January 2022): Big data in agriculture: Between opportunity and solution

[38]Market Trends (January7, 2021): The impact of Big Data in agriculture https://www.analyticsinsight.net/the-impact-of-big-data-in-agriculture/

[39] Nana Terra (January 27, 2021): Data Science in energy- what is your big data talent plan? https://www.airswift.com/blog/data-science-in-energy

[40] Ravi V Angadi, P.S Venkataraman, Suresh Babu Daram (January 17, 2020): Role of Big Data Analytics in Power system Application

[41] altexsoft (June 7, 2021): Hadoop Vs Spark: Main Big Data Tools Explained https://www.altexsoft.com/blog/hadoop-vs-spark/

[42]Soumyaa Rawat (August 12, 2021): NoSQL Database: Characteristics, Types and Applications https://www.analyticssteps.com/blogs/nosql-database-characteristics-types-and-applications

[43]Robert Buda (August 27, 2021): Beyond "Fast and Simple»: Top 5 Use Cases for NoSQL Database Technology https://www.budaconsulting.com/fast-and-simple-use-cases-for-nosql-database-technology/

[44] George Lawton (February 17, 2022): Hadoop vs Spark: An in-depth big data framework comparisonhttps://www.techtarget.com/searchdatamanagement/feature/Hadoop-vs-Spark-Comparing-the-two-big-data-frameworks

[45] Rajat Mark Grover, Ted Malaska, Jonathan Seidman, Gwen Shapira (July 2015): Hadoop Application Architectures, O'Reilly Media

[46] Douglas Edline (June 2018): Hadoop and Spark Fundamentals, Addison-Wesley Professional

[47] Scholar Nest (February 2022): Spark programming in Python for Beginners with Apache Spark 3, Packt Publishing

[48] Ευάγγελος Πεφάνης (May 7, 2022): Διαδικτυακό σεμινάριο: Ενιαίο περιβάλλον ανάπτυξης εφαρμογών Python σε Pseudo Cluster με χρήση WSL2, Hadoop,Yarn-Spark.https://www.youtube.com/watch?v=GQTQm9N4DE4&t=1824s

[49] Khalifa Mohamed (2018): Health Analytics Types, Functions and Levels: A Review of Literature

[50] Erin McNemar (August 2, 2021): What Are the Benefits of Predictive Analytics in Healthcare? https://healthitanalytics.com/news/what-are-the-benefits-of-predictive-analytics-in-healthcare

[51] Mary K. Patt (December 13, 2021): Predictive analytics in healthcare: 12 valuable use cases https://www.techtarget.com/searchbusinessanalytics/tip/Predictive-analytics-in-healthcare-12-valuable-use-cases

[52] Aaron Tan (June 15, 2021): How healthcare organisations are tapping data analytics https://trimurl.co/tbVxgG

[53] Nadejda Alkhaldi (May 24, 2021): Big Data in healthcare: where does the value come from? https://itrexgroup.com/blog/big-data-in-healthcare-examples-problems-benefits/

[54]Adrea Bastek (July 19, 2021): Data sharing in healthcare: ethical considerations for a remote era of research https://medcitynews.com/2021/07/data-sharing-in-healthcare-ethical-considerations-for-a-remote-era-of-research/

[55] Ben Lutkevich (March 10, 2023): Healthcare strategies: Machine learning for treatment planning https://www.techtarget.com/whatis/podcast/Healthcare-strategies-Machine-learning-for-treatment-planning

[56] Richard Herschel, Virginia M. Miori (May 2017): Ethics & Big Data, Elsevier

[57] Sofia Segkouli, Giuseppe Fico, Cecilia Vera-Muñoz, Mario Lecumberri, Antonis Voulgaridis, Andreas Triantafyllidis, Pilar Sala, Stefano Nunziata, Nadia Campanini, Enrico Montanari, Suzanne Morton, Alexandre Duclos,9 Francesca Cocchi, Mario Diaz Nava, Trinidad de Lorenzo, Eleni Chalkia, Matina Loukea, Juan Bautista Montalvá Colomer, George E. Dafoulas, Sergio Guillén, María Teresa Arredondo Waldmeyer, and Konstantinos Votis (May 2022): Ethical Decision Making in Iot Data Driven Research: A case of a Large-Scale Pilot https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9141539/

[58] R. Lakshmana Kumar, R. Indrakumari, B. Balamurugan, Achyut Shankar, R. Lakshmana Kumar, R. Indrakumari, B. Balamurugan, Achyut Shankar (2021): Exploratory Data Analytics for Healthcare, CRC Press

[59] Cornell Law School:

https://www.law.cornell.edu/cfr/text/45/156.20#:~:text=Actuarial%20value%20(AV)%20means%20the,total%20allowed%20costs%20of%20benefits.

[60] Ashwin Belle, Raghuram Thiagarajan, S. M. Reza Soroushmehr, Fatemeh Navidi, Daniel A. Beard and Kayvan Najarian (16 June 2015): Big Data Analytics in Health Care for Hindawi Publishing Corporation BioMed Research International Volume 2015, Article ID 370194, 16 pages http://dx.doi.org/10.1155/2015/370194

[61] Chayakrit Krittanawong, Tao Sun and Eyal Herzog (28 July 17): Big Data and Genome Editing Technology: A New Paradigm of Cardiovascular Genomics

[62] Shreyas Nopany, Prof. Manonmani S (4 July 2021): Applications of Big Data Analytics in Healthcare Management Systems

[63] Wullianallur Raghupathi1, Viju Raghupathi (7 Febuary 2014): Big data analytics in healthcare: promise and potential for Raghupathi and Raghupathi Health Information Science and Systems, http://www.hissjournal.com/content/2/1/3

[64] Alejandro Giorgetti, Andrew Harrison, Aleksey Kochetov, Jens Krüger, Qi Ma, Hiroshi Matsuno, Chanchal K. Mitra, Josch K. Pauling, Chris Rawlings, Florentino Fdez-Riverola, Paolo Romano, Richard Röttger, Alban Shoshi, Siomar de Castro Soares, Jan Taubert, Andreas Tauch, Malik Yousef, Stephan Weise, Keywan Hassani-Pak (1 January 2017): Journal of Integrative Bioinformatics, ISSN: 1613-4516 ,Torsten Krüger

[65] Costa FF. (19 April 2014): Big data in biomedicine. Drug Discov: 433-40. doi: 10.1016/j.drudis.2013.10.012. Epub 2013 Oct 29. PMID: 24183925.

[66] Kornelia Batko,Andrzej Ślęzak (2022): The use of Big Data Analytics in healthcare for Batko and Ślęzak Journal of Big Data = 9:3 https://doi.org/10.1186/s40537-021-00553-4