ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«ΒΙΒΛΙΟΘΗΚΗ R ΓΙΑ SKYLINE OPERATOR»

Του φοιτητή
Κασαπίδη Βασίλειου
Αρ. Μητρώου: 134148

Επιβλέπων
Αντώνης Σιδηρόπουλος

Ημερομηνία 14/01/2021

Τίτλος Δ.Ε.: Βιβλιοθήκη R για Skyline operator
Κωδικός Δ.Ε.: 20197
Ονοματεπώνυμο φοιτητή: Κασαπίδης Βασίλειος
Ονοματεπώνυμο εισηγητή: Αντώνης Σιδηρόπουλος
Ημερομηνία ανάληψης Δ.Ε.: 09-11-2020
Ημερομηνία περάτωσης Δ.Ε.: 14-01-2021

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Περίληψη

Η R είναι μία γλώσσα προγραμματισμού η οποία προσφέρει πληθώρα επιλογών και δυνατοτήτων για στατιστικούς υπολογισμούς και ανάλυση δεδομένων καθώς και τη γραφική απεικόνιση τους. Είναι ανοιχτό λογισμικό και παρέχεται δωρεάν υπό τη γενική άδεια GNU. Για να προστεθούν νέες δυνατότητες, οι χρήστες μπορούν να εγκαταστήσουν πακέτα τα οποία θα φορτώνουν από βιβλιοθήκες μέσω του κώδικα. Τα πακέτα αυτά είναι διαθέσιμα μέσα από αποθετήρια όπως το CRAN, δηλαδή ιστοσελίδες μέσω των οποίων οι χρήστες μπορούν να κατεβάσουν και να εγκαταστήσουν πακέτα.

Στα πλαίσια της πτυχιακής εργασίας έχει αναπτυχθεί ένα πακέτο για τη δημιουργία ενός τελεστή Skyline. Με αυτόν τον τελεστή, ο χρήστης μπορεί να βρει μέσα από ένα σύνολο δεδομένων τα σημεία αυτά που δεν είναι χειρότερα σε όλες τις διαστάσεις από τα υπόλοιπα. Χαρακτηριστικό παράδειγμα είναι αυτό με την εύρεση ενός ξενοδοχείου το οποίο πρέπει να είναι όσο το δυνατόν πιο φθηνό και πιο κοντά στη θάλασσα. Τα ξενοδοχεία συγκρίνονται μεταξύ τους ως προς τις διαστάσεις τους, βρίσκονται αυτά τα οποία δεν «ξεπερνιούνται» από άλλα και αναπαρίστανται σε ένα δισδιάστατο γράφημα.

Το πακέτο που αναπτύχθηκε περιλαμβάνει τις απαραίτητες μεθόδους για την εκτέλεση μια πράξης Skyline, λαμβάνοντας υπόψη τους διάφορους ορισμούς της κυριαρχίας ενός σημείου, της ύπαρξης ή όχι ενός περιορισμένου Skyline το οποίο θα αποκλείει κάποιες τιμές αλλά και της πιθανής ύπαρξης NULL τιμών στο αρχικό σύνολο δεδομένων. Το αποτέλεσμα των μεθόδων αυτών είναι η εξαγωγή πολλαπλών Skyline αποτελεσμάτων, τα οποία αποτυπώνονται σε γράφημα με τη βοήθεια του πακέτου ggplot2.

“R LIBRARY FOR SKYLINE OPERATOR”

Bill Kasapidis

Abstract

R is a programming language which contains a wealth of options and capabilities used in statistical computing and data analysis, as well as their graphical presentation. It is an open source software and is freely available under the GNU General Public Licence. To add more functionalities, users can install packages which will be loaded from libraries via code. These packages are available through repositories such as CRAN, websites through which users can download and install packages.

For the purposes of this thesis paper, a package for the creation of a Skyline operator has been developed. With this operator, a user can find all the points from a dataset that are not inferior in all their dimensions to all others. The search for a hotel that is both cheap and close to the sea is a classic example. The hotels are all compared with one another, those that are not “dominated” by others are selected and are displayed on a two-dimensional graph.

The developed package contains all the methods required for the execution of a Skyline computation, while taking into account the various definitions of point dominance, the existence of a constrained Skyline and the possible existence of NULL values in the initial dataset. The result of these methods is the extraction of multiple Skylines, which are displayed in a graph with the help of the ggplot2 R package.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή μου Σιδηρόπουλο Αντώνη για την ευκαιρία που μου πρόσφερε να συμμετάσχω σε αυτήν την ενδιαφέρουσα εργασία, καθώς και όλους τους καθηγητές μου που με βοήθησαν να αποκτήσω τις απαραίτητες γνώσεις ώστε αυτή η πτυχιακή εργασία να έρθει στο πέρας της. Ειδικά ένα μεγάλο ευχαριστώ στους γονείς και στους φίλους μου, οι οποίοι ποτέ δεν έπαψαν να με ενθαρρύνουν.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

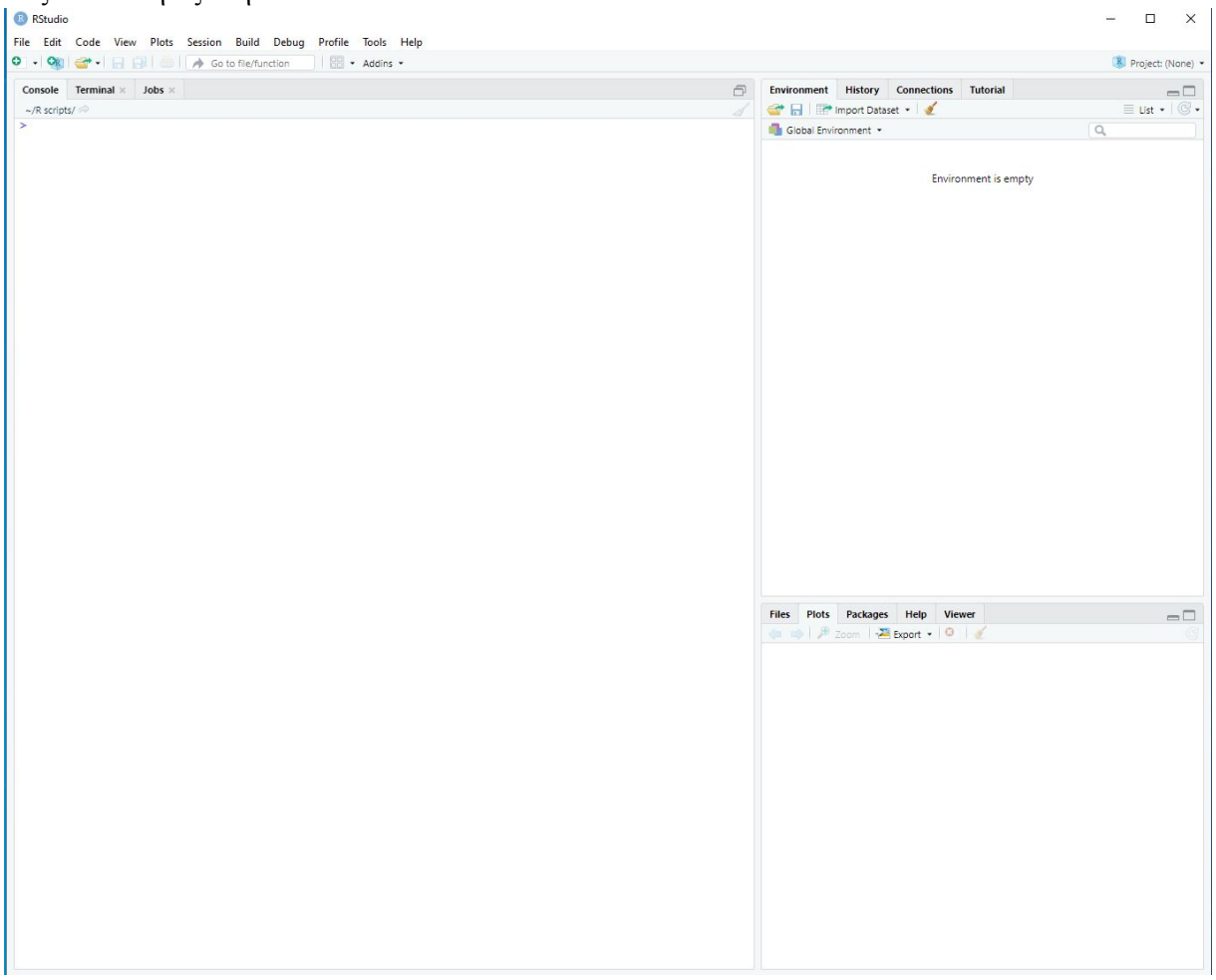
Κεφάλαιο 1ο: R

1.1 Εισαγωγή στην R

Η R είναι μία γλώσσα και περιβάλλον που χρησιμοποιείται κυρίως για στατιστικούς υπολογισμούς και την αναπαράσταση τους. Έχει πολλές ομοιότητες με την S, μια παρόμοια γλώσσα που αναπτύχθηκε στην Bell Laboratories, αλλά η R σήμερα θεωρείται διαφορετική υλοποίηση της S. Παρά τις διαφορές τους, μεγάλο τμήμα του κώδικα της S βρίσκεται και στην R [1].

Είναι μία γλώσσα ανοιχτού-κώδικα και μέρος του GNU Project [2]. Σχεδιάστηκε από τους Ross Ihaka and Robert Gentleman και υποστηρίζεται από μία κύρια ομάδα χρηστών (την R Core Team). Παρόλα αυτά, πολλοί χρήστες ανά τα χρόνια έχουν συμβάλλει με διάφορες βελτιώσεις στη γλώσσα και το περιβάλλον της.

Η R περιλαμβάνει πολλές και ποικίλες τεχνικές για διάφορους τύπους υπολογισμών, όπως γραμμική και μη γραμμική μοντελοποίηση, συσταδοποίηση, ταξινόμηση, στατιστικά τεστ, κλπ. Με την R, ο χρήστης μπορεί επίσης να αποτυπώσει το αποτέλεσμα μιας πράξης με την βοήθεια των γραφικών της δυνατοτήτων. Επιπροσθέτως, η R είναι επεκτάσιμη, με τη βοήθεια πακέτων που έχουν δημιουργηθεί και συντηρούνται από άλλους χρήστες. Τα πακέτα αυτά διανέμονται επίσης δωρεάν και προσθέτουν νέες δυνατότητες στην R.



Σχήμα 1.1: Το γραφικό περιβάλλον του RStudio

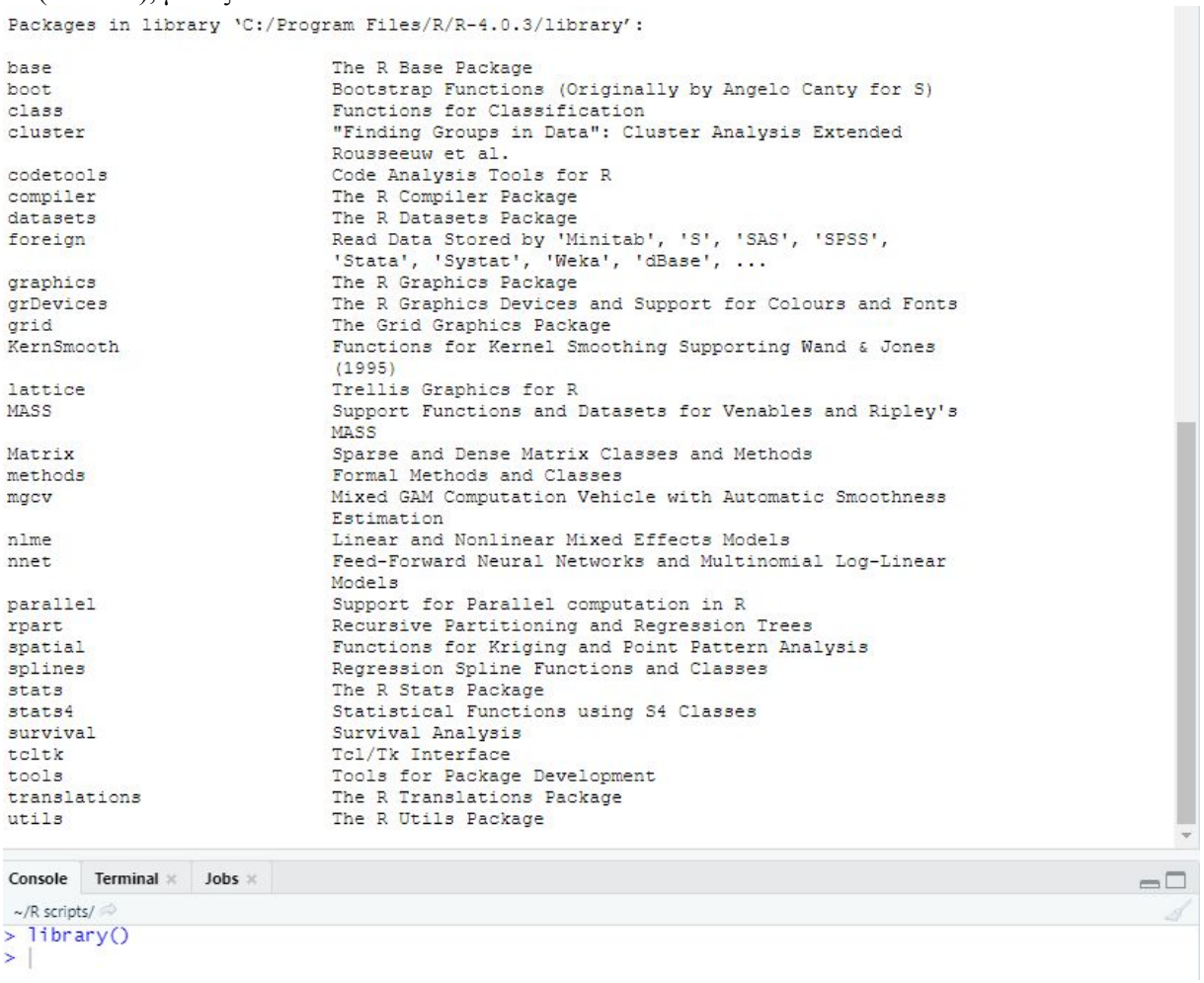
Η R περιέχει τη δική της διεπαφή γραμμής εντολών (command line interface) και μπορεί να

εγκατασταθεί σε μηχανήματα που τρέχουν πάνω σε Microsoft Windows, Mac ή Unix. Υπάρχουν και άλλα third-party περιβάλλοντα, όπως π.χ. το RStudio που χρησιμοποιήθηκε στα πλαίσια αυτής της πτυχιακής, το οποίο προσφέρει ένα πλουσιότερο περιβάλλον διεπαφής που είναι πιο φιλικό για το χρήστη.

1.2 Βιβλιοθήκες - Πακέτα R

Οι δημιουργοί της R έχουν διαφορετικό ορισμό για τις έννοιες «βιβλιοθήκη» και «πακέτο». Πακέτο είναι η συλλογή όλων των απαραίτητων αρχείων, όπως κώδικας, δεδομένα, άλλα βοηθητικά αρχεία, κλπ. τα οποία επεκτείνουν τις δυνατότητες της R. Η βιβλιοθήκη είναι η τοποθεσία από την οποία η R φορτώνει τα πακέτα. Πολλοί χρήστες χρησιμοποιούν τις δύο αυτές έννοιες αμφίδρομα, αν και τεχνικά αποτελούν δύο τελείως διαφορετικά πράγματα.

Η βασική εγκατάσταση της R περιλαμβάνει μερικά πακέτα τα οποία περιέχουν διάφορες λειτουργίες γραφικών, στατιστικές μεθόδους, υποστήριξη για παράλληλο υπολογισμό και μερικές συλλογές δεδομένων (datasets), μεταξύ άλλων.



Σχήμα 1.2 - Το αποτέλεσμα της πράξης library() η οποία εμφανίζει όλες τις φορτωμένες βιβλιοθήκες

Τα πακέτα ανεβάζονται σε αποθετήρια (repositories), μέσω των οποίων ο χρήστης μπορεί να αναζητήσει πληροφορίες για το κάθε ένα και να τα εγκαταστήσει. Το κυριότερο αποθετήριο είναι το CRAN [3], ή αλλιώς το Comprehensive R Archive Network. Αποτελεί μια συλλογή ιστοσελίδων οι οποίες περιλαμβάνουν τον ίδιο ακριβώς αριθμό και έκδοση αρχείων όπως για παράδειγμα τα

εκτελέσιμα αρχεία της R, documentation και πακέτα.

Η κύρια ιστοσελίδα εδρεύει στο Επιχειρηματικό και Οικονομικό Πανεπιστήμιο της Βιέννης και όλες οι άλλες ιστοσελίδες (mirrors) αντιγράφουν το περιεχόμενο της. Για τη χώρα μας, η ιστοσελίδα-«καθρέπτης» εδρεύει στο Πανεπιστήμιο της Κρήτης. Οι χρήστες προτρέπονται να επισκέπτονται την ιστοσελίδα που είναι πιο κοντά σε αυτούς για να γίνεται οικονομία σε πόρους δικτύου.

Πέραν από το CRAN, υπάρχουν και άλλα αποθετήρια [4], όπως το Bioconductor το οποίο περιέχει βιβλιοθήκες σχετικά με τον τομέα της γονιδιωματικής, και το R-Forge, στα οποία οι χρήστες μπορούν να συνεργαστούν για να αναπτύξουν τις βιβλιοθήκες τους. Παρόλα αυτά, το CRAN είναι το αποθετήριο που περιλαμβάνει τα πακέτα γενικού σκοπού που βοήθησαν στην ανάπτυξης της εργασίας αυτής.

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

Available CRAN Packages By Date of Publication

Date	Package	Title
2021-01-11	ActivityIndex	Activity Index Calculation using Raw 'Accelerometry' Data
2021-01-11	arcos	Load ARCOS Prescription Data Prepared by the Washington Post
2021-01-11	batchtools	Tools for Computation on Batch Systems
2021-01-11	briskaR	Biological Risk Assessment
2021-01-11	caTools	Tools: Moving Window Statistics, GIF, Base64, ROC AUC, etc
2021-01-11	cglasso	Conditional Graphical LASSO for Gaussian Graphical Models with Censored and Missing Values
2021-01-11	cld3	Google's Compact Language Detector 3
2021-01-11	cnum	Chinese Numerals Processing
2021-01-11	CompARE.design	Statistical Functions for the Design of Studies with Composite Endpoints
2021-01-11	crispRdesignR	Guide Sequence Design for CRISPR/Cas9
2021-01-11	crseEventStudy	A Robust and Powerful Test of Abnormal Stock Returns in Long-Horizon Event Studies
2021-01-11	cursr	Cursor and Terminal Manipulation
2021-01-11	dataprep	Efficient and Flexible Data Preprocessing Tools
2021-01-11	depigner	A Utility Package to Help you Deal with "Pignas"
2021-01-11	DOPE	Drug Ontology Parsing Engine
2021-01-11	DSWE	Data Science for Wind Energy
2021-01-11	eBsc	"Empirical Bayes Smoothing Splines with Correlated Errors"
2021-01-11	eList	List Comprehension and Tools
2021-01-11	ensembleTax	Ensemble Taxonomic Assignments of Amplicon Sequencing Data
2021-01-11	feisr	Estimating Fixed Effects Individual Slope Models
2021-01-11	freetypeharfbuzz	Deterministic Computation of Text Box Metrics
2021-01-11	frequency	Easy Frequency Tables
2021-01-11	fritools	Utilities for the Forest Research Institute of the State Baden-Wuerttemberg
2021-01-11	gfonts	Offline 'Google' Fonts for 'Markdown' and 'Shiny'
2021-01-11	ggnewscale	Multiple Fill and Colour Scales in 'ggplot2'
2021-01-11	glmnet	Lasso and Elastic-Net Regularized Generalized Linear Models
2021-01-11	gmvarkit	Estimate Gaussian Mixture Vector Autoregressive Model
2021-01-11	GPFDA	Gaussian Process Regression for Functional Data Analysis
2021-01-11	hermiter	Efficient Sequential and Batch Estimation of Univariate and Bivariate Probability Density Functions and Cumulative Distribution Functions along with Quantiles (Univariate) and Spearman's Correlation (Bivariate)
2021-01-11	imageData	Aids in Processing and Plotting Data from a Lemna-Tec Scananalyzer
2021-01-11	JMbayes2	Extended Joint Models for Longitudinal and Time-to-Event Data
2021-01-11	KSD	Goodness-of-Fit Tests using Kernelized Stein Discrepancy
2021-01-11	lfe	Linear Group Fixed Effects
2021-01-11	mlrCPO	Composable Preprocessing Operators and Pipelines for Machine Learning
2021-01-11	mvProbit	Multivariate Probit Models

Σχήμα 1.3 - Η σελίδα του CRAN με τα πιο πρόσφατα διαθέσιμα πακέτα

1.3 Εγκατάσταση πακέτων

Η διαδικασία εγκατάστασης πακέτων στα συστήματα Unix και Windows είναι η ίδια. Χρησιμοποιώντας την κονσόλα (CLI) της R, ο χρήστης μπορεί να χρησιμοποιήσει την εντολή install.packages() και να προσδιορίσει τα πακέτα που επιθυμεί να εγκαταστήσει.

```
install.packages(pkgs, lib, repos = getOption("repos"),
                 contriburl = contrib.url(repos, type),
                 method, available = NULL, destdir = NULL,
                 dependencies = NA, type = getOption("pkgType"),
                 configure.args = getOption("configure.args"),
                 configure.vars = getOption("configure.vars"),
                 clean = FALSE, Ncpus = getOption("Ncpus", 1L),
                 verbose = getOption("verbose"),
                 libs_only = FALSE, INSTALL_opts, quiet = FALSE,
                 keep_outputs = FALSE, ...)
```

Σχήμα 1.4 – Ο πλήρης ορισμός της install.packages

Για την μεγαλύτερη ευκολία του χρήστη, προτείνεται αντ'αυτού η εγκατάσταση των πακέτων μέσω του γραφικού περιβάλλοντος RGui που περιέχεται στην βασική εγκατάσταση της R. Με τον τρόπο αυτό ο χρήστης μπορεί να επιλέξει το CRAN mirror και τα πακέτα που επιθυμεί να εγκαταστήσει. Αν ο χρήστης δεν προσδιορίσει δικιά του βιβλιοθήκη, δηλαδή, ένα directory από το οποίο θα φορτώνονται τα πακέτα, η R θα τα εγκαταστήσει στην προεπιλεγμένη βιβλιοθήκη της.

Κατά την εκτέλεση της εντολής σε Windows λειτουργικά συστήματα, η R θα προσπαθήσει να βρει μία λίστα με τις δυαδικές εκδόσεις των πακέτων που ζήτησε ο χρήστης για την έκδοση R που τρέχει. Αν βρεθούν, τότε θα τα εγκαταστήσει, ή θα τα αναβαθμίσει στην τελευταία έκδοση. Τα δυαδικά πακέτα περιλαμβάνουν την 32-bit και τη 64-bit έκδοση του πακέτου, ανάλογα με την έκδοση του λειτουργικού του χρήστη.

Ένας άλλος τρόπος εγκατάστασης ενός πακέτου είναι μέσω ενός συμπιεσμένου zip αρχείου, από το οποίο η R θα εξάγει τα απαραίτητα αρχεία για την εγκατάσταση του.

1.4 Φόρτωση πακέτων

Όταν ο χρήστης επιθυμεί να χρησιμοποιήσει μία μέθοδο ή ένα σύνολο δεδομένων από ένα πακέτο, δεν αρκεί μονάχα να το έχει εγκατεστημένο. Στην αρχή κάθε R αρχείου, είναι απαραίτητο να φορτώσει το πακέτο το οποίο περιέχει τα αρχεία που χρειάζεται.

Για να γίνει αυτό, πρέπει να χρησιμοποιηθεί η εντολή “library(πακέτο)”. Η εντολή αυτή επαναλαμβάνεται για κάθε πακέτο που θέλει να φορτώσει ο χρήστης στο τρέχον αρχείο R. Αν δεν περαστούν ορίσματα στην εντολή library(), τότε θα τυπωθεί στην κονσόλα η λίστα με όλα τα ήδη φορτωμένα πακέτα.

1.5 Δημιουργία πακέτων R

Τα πακέτα αποτελούνται από ένα φάκελο ο οποίος περιέχει έναν αριθμό από υποφακέλους και αρχεία. Η R ορίζει πως ένα πακέτο μπορεί να περιέχει τους υποφακέλους R, data, demo, exec, inst, man, po, src, tests, tools και vignettes και τα αρχεία DESCRIPTION and NAMESPACE, INDEX, configure, cleanup, LICENSE, και NEWS.

Κάποιοι υποφάκελοι και αρχεία είναι προαιρετικά. Οι φάκελοι R, man και data κρίνονται αναγκαίοι καθώς περιέχουν τον κώδικα, το documentation και τα δεδομένα του πακέτου αντίστοιχα. Στο άλλο χέρι, οι φάκελοι vignettes, tests και το αρχείο NEWS περιλαμβάνουν βοηθητικές λειτουργίες και, ενώ καλό είναι να συμπεριληφθούν από τον δημιουργό, δεν είναι αναγκαία για την εγκατάσταση και λειτουργία του πακέτου.

Οι δημιουργοί της R προσφέρουν έναν εκτενή οδηγό [5] για το πως δημιουργείται ένα πακέτο και

ποιές είναι οι καλύτερες πρακτικές όσον αφορά τη χρησιμοποίηση μη-R κώδικα (όπως C++), του API της R, κλπ.

Για τους χρήστες που δεν επιθυμούν μία τόσο περίπλοκη διαδικασία και θα ήθελαν μια καλύτερη, πιο φιλική στο χρήστη εναλλακτική, το πρόγραμμα RStudio προσφέρει τη δυνατότητα δημιουργίας πακέτου με πολύ γρήγορο και απλό τρόπο, απλά επιλέγοντας το αρχείο με τις μεθόδους που ανέπτυξε και τον προσδιορισμό ενός φακέλου εγκατάστασης [6]. Το RStudio θα δημιουργήσει ένα directory με τους απαραίτητους φακέλους, επιτρέποντας στο χρήστη να δημιουργήσει τα απαραίτητα documentation files και να κάνει “build” το πακέτο, ολοκληρώνοντας έτσι τη δημιουργία του.

1.6 Ανέβασμα πακέτων σε αποθετήριο

Κάθε αποθετήριο έχει τη δικιά του διαδικασία και πολιτική ώστε να γίνει αποδεκτό το πακέτο ενός χρήστη. Για παράδειγμα, το CRAN απαιτεί από τον δημιουργό του πακέτου, μεταξύ άλλων, να προσδιορίσει ποιοί είναι οι συντηρητές του πακέτου, να δηλωθούν ξεκάθαρα οι άδειες και τα πνευματικά δικαιώματα και να ακολουθηθούν αυστηροί κανόνες για το πως θα πρέπει να είναι δομημένο το πακέτο (π.χ. το πηγαίο πακέτο απαγορεύεται να περιέχει εκτελέσιμο δυαδικό κώδικα).

Αντιθέτως, το R-Forge απαιτεί από τον χρήστη να δημιουργήσει λογαριασμό στην ιστοσελίδα του αποθετηρίου και τότε μπορεί να ανεβάσει το ημιτελές project του και να προσκαλέσει άλλους χρήστες να συνδράμουν στην ανάπτυξη του πακέτου.

Δεν είναι απαραίτητο για ένα πακέτο να ανέβει σε αποθετήριο για να μπορεί να αξιοποιηθεί από άλλους χρήστες. Το πακέτο μπορεί να διανεμηθεί ως ένα zip αρχείο και να φιλοξενηθεί σε άλλες σελίδες, όπως π.χ. τη GitHub σελίδα του δημιουργού.

Κεφάλαιο 2ο: Skyline

2.1 Πράξη Skyline

Η πράξη skyline προτάθηκε από τους Börzsönyi et al. [7] ως μία επέκταση της SQL για χρήση στις βάσεις δεδομένων. Με την πράξη skyline, ο χρήστης μπορεί να εμφανίζει τις γραμμές ενός πίνακα των οποίων οι τιμές δεν είναι χειρότερες από όλες τις άλλες. Το όνομα Skyline (γραμμή του ορίζοντα) βγαίνει από το σχήμα του γραφήματος που προκύπτει από την εφαρμογή της πράξης αυτής.

Στο σχήμα 2.1 φαίνεται η πράξη Skyline όπως έχει προταθεί από τους Börzsönyi et al. Όπου d1, … dm είναι οι διαστάσεις του πίνακα και το MIN, MAX, DIFF διευκρινίζει αν η τιμή σε εκείνη την διάσταση πρέπει να είναι μικρότερη, μεγαλύτερη ή και διαφορετική από όλες τις άλλες. Το DISTINCT θα αφαιρέσει τις διπλότυπες γραμμές από το τελικό αποτέλεσμα.

```
SELECT ... FROM ... WHERE ...
GROUP BY ... HAVING ...
SKYLINE OF [DISTINCT] d1 [MIN | MAX | DIFF],
                  ..., dm [MIN | MAX | DIFF]
ORDER BY ...
```

Σχήμα 2.1: Η προτεινόμενη σύνταξη της πράξης Skyline

Για να καταλάβουμε καλύτερα την χρήση μιας skyline πράξης, θα δούμε το παράδειγμα εύρεσης ξενοδοχείων. Έστω ότι πρόκειται να ταξιδέψουμε σε μία παραθαλάσσια πόλη για δουλειές, όπως, π.χ. η Θεσσαλονίκη. Χρειαζόμαστε ένα δωμάτιο ξενοδοχείου για την διανομή μας, αλλά θέλουμε ιδανικά το ξενοδοχείο να είναι κοντά στην θάλασσα. Παρόλα αυτά, μιας και τα ξενοδοχεία που είναι κοντά στην θάλασσα προσφέρουν την πιο όμορφη θέα, οι τιμές των δωματίων τους είναι μεγαλύτερες από ξενοδοχεία που είναι πιο μακριά από την θάλασσα.

Για να μάθουμε ποια ξενοδοχεία είναι τα καλύτερα, δηλαδή, ποια ξενοδοχεία είναι πιο κοντά στην θάλασσα και φθηνότερα από τα άλλα, μπορούμε να χρησιμοποιήσουμε την πράξη Skyline σε έναν θεωρητικό πίνακα SQL ο οποίος περιλαμβάνει την τιμή κάθε ξενοδοχείου και την απόσταση του από την θάλασσα. Ένα ξενοδοχείο θεωρείται καλύτερο από ένα άλλο όταν αυτό είναι καλύτερο ή ίσο σε όλες τις διαστάσεις (σε αυτό το παράδειγμα, τιμή και απόσταση) και καλύτερο σε τουλάχιστον μία διάσταση.

2.1.1 Υλοποίηση της Skyline σε SQL

Στην SQL, η εντολή που θα τρέχαμε για να υπολογιστούν τα καλύτερα ξενοδοχεία στην Θεσσαλονίκη είναι η εξής:

```
SELECT *
FROM hotels
WHERE City = 'Thessaloniki'
SKYLINE OF price MIN, distance MIN ;
```

Στην πράξη, η Skyline μπορεί να γραφτεί ως μία εμφωλευμένη εντολή SELECT.

```
SELECT *
FROM hotels h1
WHERE h1.City = 'Thessaloniki' AND NOT EXISTS (
        SELECT *
        FROM hotels h2
        WHERE h2.City = 'Thessaloniki' AND
        h2.price <= h1.price AND h2.distance <= h1.distance AND
        (h2.distance < h1.distance OR h2.price < h1.price)
) ;
```

Για να δουλέψει το παραπάνω παράδειγμα, ο πίνακας πρέπει να ταξινομηθεί ως προς ένα από τα γνωρίσματα του (π.χ. την τιμή). Έχοντας έναν ταξινομημένο δισδιάστατο πίνακα, η διαδικασία σύγκρισης των ξενοδοχείων σε ζευγάρια είναι πάρα πολύ εύκολη. Στο σχήμα 2.2 βλέπουμε έναν τέτοιο πίνακα με τρία ξενοδοχεία, ταξινομημένα ως προς τη τιμή τους με αύξουσα σειρά.

Επειδή το δεύτερο ξενοδοχείο είναι χειρότερο συγκριτικά με το πρώτο και στις δύο διαστάσεις, μιας και είναι πιο ακριβό και πιο μακριά από την θάλασσα, η πράξη Skyline θα το αφαιρέσει από τα αποτελέσματα και το επόμενο ξενοδοχείο θα συγκριθεί με το πρώτο.

Το τρίτο ξενοδοχείο, παρόλο που είναι πιο ακριβό από το πρώτο, είναι παράλληλα πιο κοντά στη θάλασσα. Εφόσον η συνθήκη που θέλει τη μία γραμμή του πίνακα να είναι καλύτερη ή ίση από αυτήν που προηγείται σε όλες τις διαστάσεις δεν ικανοποιείται, τότε το Ξενοδοχείο 3 δεν διαγράφεται από τα αποτελέσματα και εισάγεται στο γράφημα της Skyline.

```
Ξ1, €15, 60μ
Ξ2, €20, 75μ
Ξ3, €25, 40μ
```

Σχήμα 2.2: Τα ξενοδοχεία του παραδείγματος, με αύξουσα ταξινόμηση ως προς τη τιμή

2.1.2 Περιορισμοί

Πρέπει να αναφερθεί ότι ο παραπάνω αλγόριθμος δεν μπορεί να λειτουργήσει σε πίνακες >2 διαστάσεων. Στο σχήμα 2.3 δίνεται ένα παράδειγμα του πίνακα των ξενοδοχείων σε 3 διαστάσεις – τιμή, απόσταση και ποιότητα (σε αστέρια).

Βλέπουμε πως το ξενοδοχείο Ξ3 νικιέται από το Ξ1, διότι το πρώτο είναι καλύτερο ή ίσο στη διάσταση της τιμής, απόστασης αλλά και ποιότητας. Παρόλα αυτά, τα 2 αυτά ξενοδοχεία δεν είναι σε διπλανές θέσεις, οπότε ο αλγόριθμος θα συγκρίνει το Ξ2 με το Ξ3 και όταν δεν θα θα ικανοποιηθεί η συνθήκη (το Ξ2 είναι πιο φθηνό και πιο κοντά στη θάλασσα, αλλά έχει μόνο 3 αστεράκια), το Ξ3 θα συμπεριληφθεί στο αποτέλεσμα της Skyline.

```
Ξ1, €10, 35μ, ****
Ξ2, €14, 50μ, ***
Ξ3, €17, 55μ, ****
```

Σχήμα 2.3: Τα ξενοδοχεία του παραδείγματος σε τρεις διαστάσεις, με αύξουσα ταξινόμηση ως προς τη τιμή

Για το σωστό υπολογισμό του αποτελέσματος σε πίνακες με παραπάνω από 2 διαστάσεις κριτηρίων, οι Börzsönyi et al. πρότειναν και άλλους αλγορίθμους, βασισμένους στην ίδια πρακτική των εμφωλευμένων εντολών (Block-nested-loops) και Διαίρει και Βασίλευε (Divide and Conquer) αλγορίθμων. Οι αλγόριθμοι αυτοί όχι μόνο μπορούν να διαχειριστούν δεδομένα με 3 και περισσότερες διαστάσεις, αλλά είναι πολύ πιο αποδοτικοί απ'τον τον βασικό αλγόριθμο.

2.2 Η πράξη Skyline στην R

Χρησιμοποιώντας την R, μπορούμε εύκολα να προγραμματίσουμε έναν αλγόριθμο για να εκτελούμε πράξεις Skyline. Παραμένοντας στο παράδειγμα των ξενοδοχείων, ο αλγόριθμος θα μας βοηθήσει να βρούμε ποια ξενοδοχεία είναι τα καλύτερα, να τα εκτυπώσουμε σε λίστα, αλλά και να τα αποτυπώσουμε σε ένα γράφημα. Αντίθετα με την SQL, στην R δεν έχουμε τον περιορισμό των διαστάσεων, καθώς μπορούμε να τον εφαρμόσουμε σε θεωρητικά άπειρες διαστάσεις. Και επειδή δεν απαιτείται ταξινόμηση των δεδομένων, η εφαρμογή του αλγορίθμου είναι λιγότερο ακριβή για πολύ μεγάλα datasets.

Εφόσον μετακινούμαστε από το κόσμο της τεχνολογίας των βάσεων δεδομένων, μιας και η Skyline αρχικά προτάθηκε σαν επέκταση της SQL, η ορολογία που αρμόζει περισσότερο στην R είναι οι μαθηματικές έννοιες πάνω στις οποίες βασίζεται η Skyline. Στα μαθηματικά, το αποτέλεσμα της πράξης Skyline ονομάζεται σύνολο Παρέτο (Pareto set) και περιλαμβάνει όλα τα στοιχεία τα οποία δεν είναι χειρότερα από όλα τα άλλα, δηλαδή είναι αποδοτικά κατά Παρέτο.

Πιο συγκεκριμένα, ο ορισμός λέει ότι ένα στοιχείο είναι αποδοτικό κατά Παρέτο όταν το στοιχείο αυτό δεν επιδέχεται άλλη βελτίωση. Αυτό σημαίνει πως αν δεν υπάρχει άλλο σημείο στο dataset το οποίο κρίνεται σαφώς καλύτερο, τότε το σημείο αυτό είναι αποδοτικό κατά Παρέτο και συμπεριλαμβάνεται στο σύνολο Παρέτο.

2.2.1 Ψευδοκώδικας

Δίνεται η απλούστερη έκδοση αυτού του αλγορίθμου σε ψευδοκώδικα:

```
1  ΔΙΑΒΑΣΕ hotels
2
3  paretoArray = []
4
5  ΓΙΑ i=1 ΜΕΧΡΙ hotels.size ΕΠΑΝΑΛΑΒΕ
6      j=i
7      ΑΝ existsBetter(hotels[j]) ΤΟΤΕ
8          paretoArray[j] = hotels[j]
9          ΓΙΑ j=i ΜΕΧΡΙ hotels.size ΕΠΑΝΑΛΑΒΕ
10             ΑΝ hotels[j] > paretoArray[j] ΤΟΤΕ
11                 paretoArray[j] = hotels[j]
12             ΤΕΛΟΣ ΑΝ
13         ΤΕΛΟΣ ΕΠΑΛΗΨΗΣ
14     ΤΕΛΟΣ ΑΝ
15 ΤΕΛΟΣ ΕΠΑΛΗΨΗΣ
16 ΕΜΦΑΝΙΣΕ paretoArray
17
```

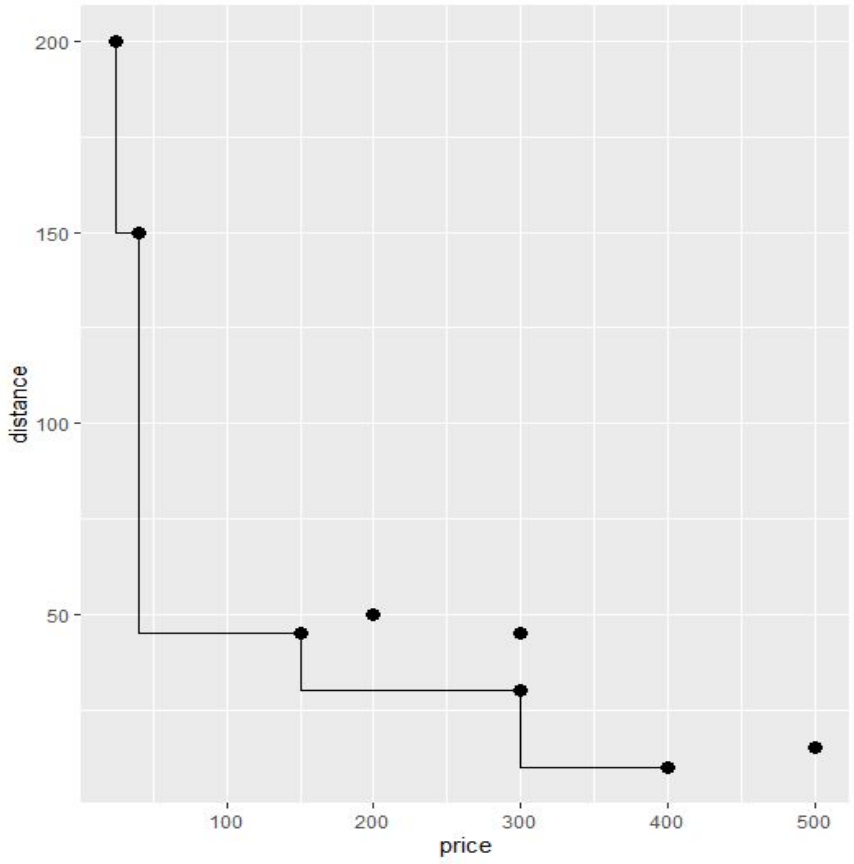
Σχήμα 2.4: Ψευδοκώδικας αλγορίθμου

Ξεκινάμε διαβάζοντας τον πίνακα των ξενοδοχείων και αρχικοποιούμε τον πίνακα που θα κρατάει τα ξενοδοχεία τα οποία είναι αποτελεσματικά κατά Παρέτο. Κάθε ξενοδοχείο που εξετάζεται, δηλαδή το υποψήφιο σημείο, πρέπει πρώτα να συγκρίνεται με όλα τα στοιχεία που βρίσκονται μέσα στον πίνακα Παρέτο (μέθοδος existsBetter). Αν υπάρχει σημείο μέσα στον τελευταίο πίνακα το οποίο υπερνικά το υποψήφιο, τότε το απορρίπτουμε και επιλέγουμε το επόμενο υποψήφιο σημείο.

Αν δεν υπάρχει σημείο στον πίνακα Παρέτο που να υπερνικά τον υποψήφιο, τότε το συγκρίνουμε με όλα τα άλλα στοιχεία του πίνακα των ξενοδοχείων για να βρούμε αν το υπερνικά κάποιο άλλο. Αν βρεθεί καλύτερη εναλλακτική, τότε αυτό το ξενοδοχείο γίνεται ο νέος υποψήφιος (γραμμή 11). Συνεχίζουμε μέχρι να ελεγχθούν όλα τα ξενοδοχεία μεταξύ τους.

Αξίζει να σημειώσουμε εδώ πως ο ψευδοκώδικας μας γνωρίζει από μόνος του τις προτιμήσεις των κριτηρίων, δηλαδή δεν του προσδιορίζουμε ότι ψάχνουμε τις χαμηλότερες τιμές και τις χαμηλότερες αποστάσεις από την θάλασσα. Αυτό γίνεται χάριν συντομίας. Το πως ορίζονται οι προτιμήσεις των κριτηρίων και η κυριαρχία των σημείων αναγράφεται με μεγαλύτερη λεπτομέρεια παρακάτω.

Κατά τη λήξη του αλγορίθμου, ο πίνακας pareto θα περιλαμβάνει όλα τα αποδοτικά κατά Παρέτο ξενοδοχεία. Τέλος, το αποτέλεσμα αποτυπώνεται σε ένα διάγραμμα.



Σχήμα 2.5: Το αποτέλεσμα του θεωρητικού αλγορίθμου

2.3 Dominance - Κυριαρχία σημείων

Μέχρι τώρα, χρησιμοποιούσαμε τον ίδιο ορισμό για την κυριαρχία ενός σημείου. Ένα σημείο κυριαρχεί ενός άλλου όταν είναι καλύτερο ή ίσο σε όλες τις διαστάσεις και καλύτερο σε τουλάχιστον μία διάσταση [8]. Στο παράδειγμα των ξενοδοχείων, θέλουμε το ξενοδοχείο s να είναι μικρότερο σε τιμή αλλά και απόσταση από τη θάλασσα από το σημείο t. Βασισμένοι στο παράδειγμα αυτό, θεωρούμε καλύτερο το σημείο s όταν αυτό έχει τιμές στις διαστάσεις τιμή και απόσταση μικρότερες

από το σημείο t. Οπότε, ο ορισμός **{1}** γράφεται ως εξής:

Ενα στοιχείο s κυριαρχεί το t ($s < t$) όταν

$$\forall i \in [0,d),\ s[i] \leq t[1] \text{ και } \exists\ j \in [0,d) \text{ ώστε } s[i] < t[i]$$

Αν θέλουμε μια πιο αυστηρή κυριαρχία στην Skyline πράξη, μπορούμε να ορίσουμε την κυριαρχία ενός σημείου έτσι ώστε να λαμβάνουμε υπόψιν μονάχα τα στοιχεία τα οποία είναι ξεκάθαρα καλύτερα από τα άλλα. Δηλαδή ο ορισμός **{2}** είναι ο:

Ένα στοιχείο s κυριαρχεί αυστηρά το t ($s << t$) όταν

$$\forall i \in [0,d),\ s[i] < t[i]$$

Αν θέλουμε να περιορίσουμε τις τιμές που εξετάζουμε γιατί δεν θέλουμε να τις συμπεριλάβουμε στο τελικό αποτέλεσμα μας, τότε ορίζουμε τα όρια (qL, qU). Όλες οι τιμές που θα εξετάζονται πρέπει να βρίσκονται εντός των ορίων. Το νέο περιορισμένο σύνολο δεδομένων **{3}** είναι το

$$s \in S : qL < s < qU$$

Έχοντας ένα σύνολο σημείων S και μία περιορισμένη περιοχή (qL, qU), μπορούμε να ορίσουμε **{4}** την πράξη Skyline ως εξής:

$$SKY(S, (qL,qU)) = \{ s \in S(qL,qU) : \nexists t \in S(qL,qU), t < s$$

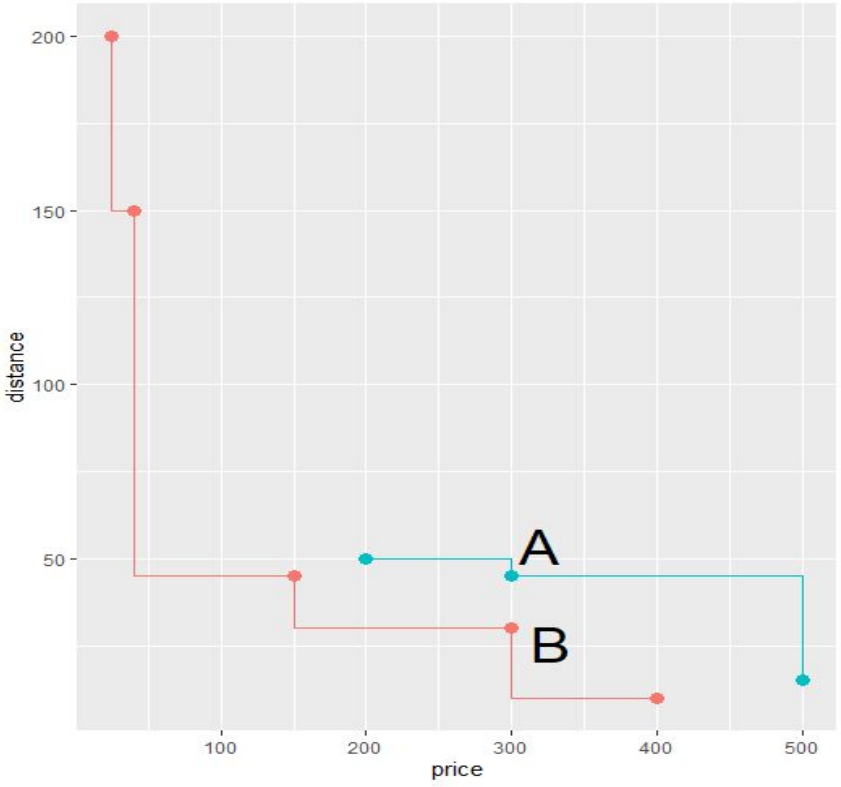
2.3.1 Διαφορά μεταξύ απλής και αυστηρής κυριαρχίας

Θα χρησιμοποιήσουμε το σύνολο δεδομένων του σχήματος 2.5 για να δείξουμε τη διαφορά μεταξύ των δύο ορισμών της κυριαρχίας των σημείων. Εφόσον πρόκειται για ξενοδοχεία, προτιμούνται οι χαμηλές τιμές και αποστάσεις από την θάλασσα.

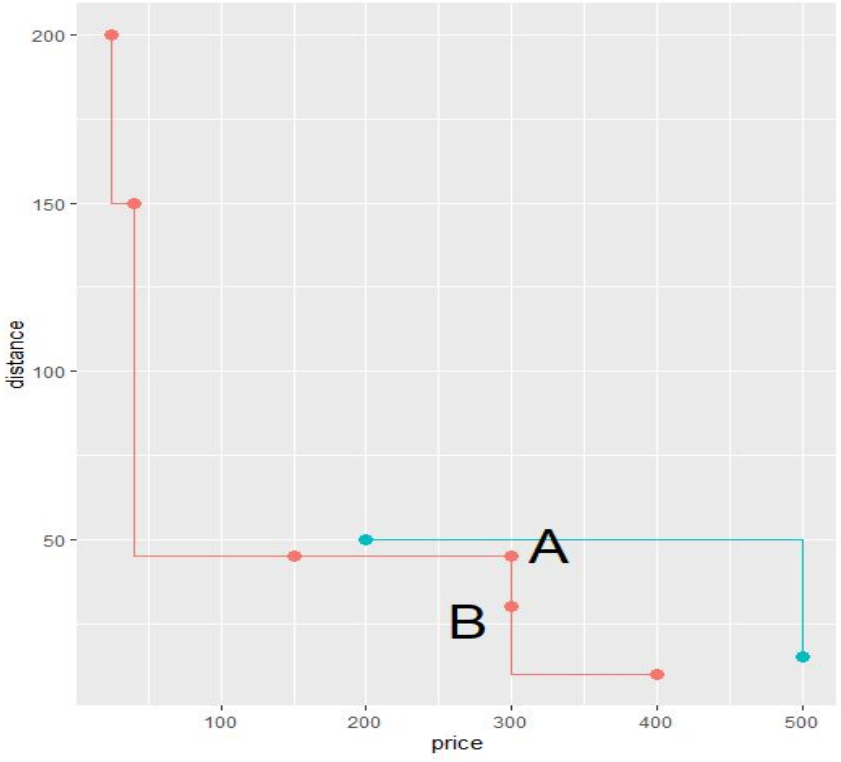
Στα σχήματα 2.6 και 2.7 φαίνονται τα αποτελέσματα δύο διαδοχικών πράξεων χρησιμοποιώντας την απλή και αυστηρή μέθοδο κυριαρχίας αντίστοιχα. Και στις δύο περιπτώσεις, τα σημεία χωρίζονται σε δύο σύνολα αποδοτικών κατά Παρέτο σημείων, τα οποία ενώνονται μεταξύ τους με γραμμή του ίδιου χρώματος. Παρατηρούμε ότι το σημείο Α στις συντεταγμένες 300,45 ανήκει σε διαφορετικά σύνολα Παρέτο ανάλογα με το πως ορίζουμε την κυριαρχία σημείων.

Με την απλή κυριαρχία, το σημείο Β (300,30) συγκρίνεται με το Α. Επειδή ισχύει $300_A = 300_B$ και $45_A > 30_B$, το Β είναι καλύτερο ή ίσο του Α, άρα το κυριαρχεί και παίρνει τη θέση του στο πρώτο σύνολο Παρέτο (πράσινη γραμμή).

Με την αυστηρή κυριαρχία, η σύγκριση του Α με Β έχει διαφορετικό αποτέλεσμα. Το Β δεν κυριαρχεί, καθώς δεν είναι αυστηρά καλύτερο του Α. Επομένως, το Α και το Β ενσωματώνονται στο ίδιο σύνολο Παρέτο.



Σχήμα 2.6: Πράξη Skyline με απλή κυριαρχία σημείων



Σχήμα 2.7: Πράξη Skyline με αυστηρή κυριαρχία σημείων

2.4 Διαχείριση NULL τιμών

Γενικότερα, είναι ευθύνη του χρήστη να προσέχει ότι το σύνολο δεδομένων που χρησιμοποιεί στον αλγόριθμο δεν περιέχει NULL τιμές. Προφανώς, αν στο παράδειγμα των ξενοδοχείων η τιμή ενός ξενοδοχείου δεν υπάρχει, αυτό θα προκαλέσει πρόβλημα στη διαδικασία εύρεσης κυρίαρχου σημείου.

Στην R, δεν μπορεί να γίνει έλεγχος μίας τιμής με το NULL, καθώς το δεύτερο υποδηλώνει έλλειψη τιμής. Για παράδειγμα, η λογική πράξη 1 != NULL (1 διάφορο του NULL) βγάζει πάντα FALSE, παρόλο που εμείς τις θεωρούμε ως διαφορετικές τιμές.

Όπως θα δούμε και παρακάτω, τα δεδομένα περνάνε μέσα στον αλγόριθμο μέσω ενός CSV αρχείου. Η R κατά το διάβασμα αυτού του αρχείου καταλαβαίνει πότε λείπει μία τιμή και στη θέση της βάζει την τιμή NA (Not Available), μία λογική σταθερά η οποία υποδηλώνει έλλειψη τιμής. Παρόλο που τεχνικά διαφέρει από τη NULL, θα τις εξισώσουμε καθώς και οι δύο δημιουργούν τα ίδια προβλήματα όταν συγκρίνονται με έναν αριθμό.

Μία επιλογή είναι κατά τον έλεγχο δύο εγγραφών στο σύνολο δεδομένων, να συγκριθούν μονάχα οι στήλες (διαστάσεις) που δεν περιέχουν NULL τιμές. Αυτή η προσέγγιση όμως αυξάνει την πολυπλοκότητα του αλγορίθμου αλλά και θα προκαλέσει πρόβλημα κατά τη δημιουργία του γραφήματος. Αν μία διάσταση αναπαριστάται στο γράφημα ως ένας άξονας και μία εγγραφή έχει NULL τιμή σε εκείνη τη διάσταση, τότε δεν θα μπορεί να αποτυπωθεί στο γράφημα.

Τέλος, οι NULL τιμές θα αλλοιώσουν το σύνολο των αποτελεσμάτων του αλγορίθμου. Ένα ξενοδοχείο μπορεί να κυριαρχεί ένα άλλο στις διαστάσεις της ποιότητας και στην απόσταση από τη θάλασσα, παρόλα αυτά, αν δεν μπορούμε να συγκρίνουμε πόσο ακριβά είναι επειδή το ένα έχει NULL τιμή, δεν είναι δυνατόν να γνωρίζουμε αν πρόκειται όντως για καλή επιλογή.

Για τους λόγους αυτούς, στην αρχή του αλγορίθμου θα ελέγχεται το σύνολο δεδομένων και θα διαγράφονται οι γραμμές αυτές που περιλαμβάνουν έστω και μία NULL τιμή.

2.5 Πολλαπλά σύνολα Παρέτο

Αν ο χρήστης το επιλέξει, ο αλγόριθμος εύρεσης ενός συνόλου αποδοτικών κατά Παρέτο στοιχείων μπορεί να επαναληφθεί. Η δεύτερη εκτέλεση του αλγορίθμου θα εντοπίσει τα σημεία που είναι καλύτερα από όλα τα άλλα, τα οποία όμως είναι χειρότερα από τα σημεία που επιλέχθηκαν κατά την πρώτη εκτέλεση του αλγορίθμου. Στην ουσία, βρίσκουμε τα δεύτερα καλύτερα σημεία.

Η υλοποίηση αυτής της λειτουργίας είναι απλή. Κάθε φορά που εκτελείται ο αλγόριθμος εύρεσης ενός συνόλου Παρέτο, όταν βρεθούν τα αποδοτικά κατά Παρέτο στοιχεία, θα αφαιρούνται από το αρχικό σύνολο δεδομένων. Αν το νέο, μικρότερο σύνολο δεδομένων έχει παραπάνω από μία γραμμή, τότε ο αλγόριθμος θα βρει τα δεύτερα πιο αποδοτικά στοιχεία. Η διαδικασία επαναλαμβάνεται μέχρι να τελειώσουν οι γραμμές στο σύνολο δεδομένων.

Τα σύνολα αυτά μπορούν να αποτυπωθούν στο ίδιο γράφημα και οι γραμμές που θα ενώνουν τα σημεία του ίδιου συνόλου Παρέτο θα έχουν το δικό τους χρώμα, ώστε να διαφοροποιούνται μεταξύ τους. Περισσότερες λεπτομέρειες για αυτή τη διαδικασία βρίσκονται στο κεφάλαιο 3.3.

Κεφάλαιο 3ο: Υλοποίηση Skyline στην R

3.1 Βασικές μέθοδοι

Παρακάτω γίνεται επεξήγηση των αλγορίθμων που περιέχονται στο πακέτο που θα επιτρέπει στο χρήστη να βρει τα αποδοτικά κατά Παρέτο σημεία από ένα σύνολο δεδομένων, αλλά και να χειριστεί NULL τιμές, θα θέσει Skyline όρια και θα επιλέξει ποια μέθοδο κυριαρχίας σημείων επιθυμεί.

- *sanitizeData (dataset)*

  Δέχεται σαν όρισμα ένα σύνολο δεδομένων. Το ελέγχει για γραμμές με NULL τιμές, τις διαγράφει αν υπάρχουν και το επιστρέφει.

- *setConstraint (dataset, dimension, lower, upper)*

  Αν ο χρήστης επιθυμεί τις γραμμές του συνόλου δεδομένων στις οποίες μία διάσταση πρέπει να ανήκει μεταξύ 2 ορίων (βλ. κεφάλαιο ταδε), πρέπει να χρησιμοποιήσει την μέθοδο αυτή και σαν ορίσματα να περάσει το σύνολο δεδομένων, το όνομα της διάστασης και το κάτω και άνω όριο. Η μέθοδος θα επιστρέψει το σύνολο δεδομένων χωρίς τις γραμμές που περιλαμβάνουν τιμές εκτός των ορίων για τη διάσταση που επισημάνθηκε. Αν ο χρήστης επιθυμεί να θέσει όρια και για άλλη διάσταση, μπορεί να ξανακαλέσει την μέθοδο περνώντας σαν όρισμα το ανανεωμένο dataset.

- *calculatePareto(dataset, preferences, strict)*

  Η μέθοδος δέχεται ως όρισμα ένα σύνολο δεδομένων, ένα διάνυσμα επιλογών (low ή high) και μία προαιρετική boolean μεταβλητή strict. Με τη χρήση των βοηθητικών μεθόδων *betterInAll* και *betterInAtLeastOne* (κεφάλαιο 3.2), υπολογίζεται η κυριαρχία των σημείων. Αν ο χρήστης θέσει την strict ως TRUE, θα κληθεί η βοηθητική μέθοδος *betterStrict* που θα υπολογίσει το σύνολο Παρέτο χρησιμοποιώντας αυστηρή κυριαρχία σημείων. Στο τέλος, η μέθοδος επιστρέφει ένα διάνυσμα το οποίο περιέχει τα αποδοτικά κατά Παρέτο σημεία του dataset βάσει των προτιμήσεων που δόθηκαν.

- *recalculate(dataset, pareto)*

  Αν ο χρήστης επιθυμεί να υπολογίσει πολλαπλά σύνολα Παρέτο, τότε πρέπει να χρησιμοποιήσει αυτήν τη μέθοδο η οποία δέχεται σαν ορίσματα το σύνολο δεδομένων και το διάνυσμα με τα αποδοτικά κατά Παρέτο σημεία. Ο σκοπός είναι να διαγραφούν από το σύνολο δεδομένων οι γραμμές που είναι κοινές με το διάνυσμα pareto. Αν δεν γίνει αυτό, τότε η δεύτερη κλήση της *calculatePareto* θα επιστρέψει το ίδιο αποτέλεσμα, καθώς θα δεχτεί σαν όρισμα το ίδιο σύνολο δεδομένων και θα επιστραφεί ένα διάνυσμα με τα ίδια ακριβώς σημεία.

3.2 Βοηθητικές μέθοδοι

Στο πακέτο περιλαμβάνονται και άλλες, βοηθητικές μέθοδοι οι οποίες βοηθούν τις άλλες μεθόδους για συγκεκριμένους υπολογισμούς.

- *existsBetter (candidate,pareto,preferences)*

  Με τη βοήθεια αυτής της μεθόδου ο αλγόριθμος θα προσπαθήσει να βρει ένα καλύτερο σημείο στον πίνακα pareto, ο οποίος περιέχει αποδοτικά κατά Παρέτο σημεία, βάσει των

προτιμήσεων του χρήστη. Αν βρεθεί καλύτερο σημείο, τότε ο υποψήφιος θα απορριφθεί και η διαδικασία θα επαναληφθεί για τον επόμενο. Για την αυστηρή κυριαρχία, χρησιμοποιείται η παρόμοια μέθοδος *existsBetterStrict*.

- *betterInAll (candidate,newCandidate,preferences)*

  Σύμφωνα με το πρώτο σκέλος του ορισμό της απλής κυριαρχίας σημείων ({1} στο κεφάλαιο 2.3), ένα σημείο πρέπει να είναι τουλάχιστον καλύτερο ή ίσο σε όλες τις διαστάσεις από το σημείο με το οποίο συγκρίνεται. Αν ο νέος υποψήφιος που εξετάζεται είναι χειρότερος σε έστω και μία διάσταση από τον υποψήφιο, η μέθοδος τερματίζει πρόωρα στέλνοντας FALSE. Αυτό γίνεται για να αποφευχθούν περιττοί έλεγχοι.

- *betterInAtLeastOne (candidate,newCandidate,preferences)*

  Σύμφωνα με το δεύτερο σκέλος του ορισμού της απλής κυριαρχίας σημείων ({1} στο κεφάλαιο 2.3), ένα σημείο πρέπει να είναι καλύτερο σε τουλάχιστον μία διάσταση από το άλλο. Αν βρεθεί έστω και μία διάσταση στην οποία το νέο υποψήφιο σημείο είναι καλύτερο από το άλλο, τότε η μέθοδος επιστρέφει TRUE και τερματίζει πρόωρα, χωρίς να κάνει νέους ελέγχους.

- *betterInAllStrict(candidate,newCandidate,preferences)*

  Σύμφωνα με τον ορισμό της αυστηρής κυριαρχίας σημείων ({2} στο κεφάλαιο 2.3), ένα σημείο πρέπει να είναι καλύτερο σε όλες τις διαστάσεις από το άλλο. Αν το νέο υποψήφιο σημείο είναι χειρότερο ή ίσο από το άλλο, δηλαδή δεν είναι αυστηρά καλύτερο, τότε η μέθοδος επιστρέφει FALSE.

3.3 Δημιουργώντας το γράφημα Παρέτο

Μόλις ο χρήστης βρει ένα ή περισσότερα σύνολα Παρέτο, μπορεί να προχωρήσει στη δημιουργία ενός γραφήματος. Αυτό θα γίνει με τη βοήθεια του πακέτου 'ggplot2' το οποίο δίνει στο χρήστη πολλές επιλογές σχετικά με την εμφάνιση διαφορετικών συνόλων δεδομένων.

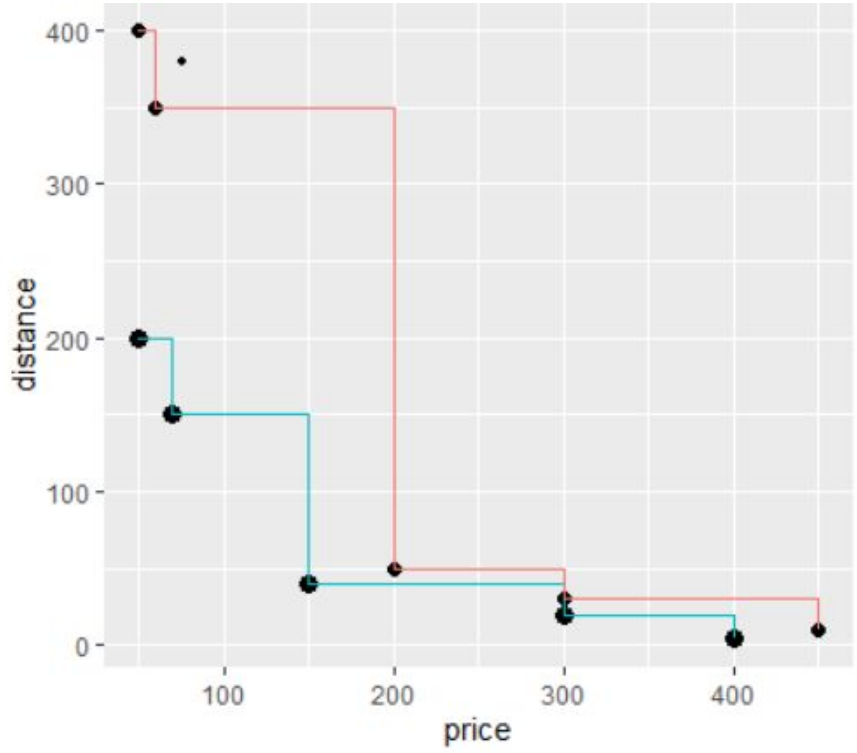
```
1  plot1 <- ggplot(dataset, aes(x = ΑΞΟΝΑΣ1, y = ΑΞΟΝΑΣ2)) +
2    geom_point(data = pareto1, size = 3) +
3    geom_point(data = pareto2, size = 2) +
4    geom_point(data = dataset, size = 1) +
5    geom_step(data=pareto1,direction = "hv",aes(colour="red"))+
6    geom_step(data=pareto2,direction = "hv",aes(colour="blue"))
7  plot(plot1)
8
```

Σχήμα 3.1 - Δημιουργία γραφήματος για δύο Skyline πράξεις

Στο σχήμα 3.1 δίνεται ένα παράδειγμα δημιουργίας ενός γραφήματος για 2 σύνολα Παρέτο. Η διαδικασία ξεκινάει με τη δημιουργία της μεταβλητής plot1 στην οποία ανατίθεται η μέθοδος ggplot με τα ορίσματα που φαίνονται στο σχήμα. Στην πρώτη σειρά περνιούνται τα ορίσματα dataset, δηλαδή το πλήρες σύνολο δεδομένων, και aes όπου ορίζονται τα ονόματα των δύο αξόνων. Στις γραμμές 2-4 με το όρισμα geom_point δημιουργούνται κουκίδες στο γράφημα συγκεκριμένου μεγέθους. Οι γραμμές 2 και 3, συγκεκριμένα, θα δημιουργήσουν κουκίδες για τα πιο δύο σύνολα Παρέτο, ενώ στη γραμμή 3 θα προστεθούν τα εναπομείναντα σημεία (αν υπάρχουν). Στις γραμμές 5 και 6 προστίθενται τα τελευταία δύο ορίσματα τα οποία δημιουργούν γραμμές διαφορετικού χρώματος που ενώνουν τα

σημεία των ίδιων συνόλων.

Τέλος, στη γραμμή 7 η μέθοδος plot δέχεται σαν όρισμα το plot1 και εμφανίζει το γράφημα (σχήμα 3.2).



Σχήμα 3.2 - Skyline γράφημα για δύο σύνολα Παρέτο τα οποία περιλαμβάνουν τα πιο αποδοτικά σημεία ξενοδοχείων.

3.4 Παράδειγμα

Παρατίθεται ένα παράδειγμα στο οποίο χρησιμοποιούνται όλες οι κύριοι μέθοδοι για να βρεθούν τρία σύνολα Παρέτο από ξενοδοχεία στις διαστάσεις τιμή, απόσταση από τη θάλασσα, και ποιότητα ξενοδοχείου. Προφανώς, προτιμούμε τα ξενοδοχεία τα οποία έχουν χαμηλή τιμή και απόσταση από τη θάλασσα και υψηλή ποιότητα.

Το φόρτωμα του συνόλου δεδομένων ξεκινά με τη μέθοδο *read.csv* η οποία δέχεται σαν όρισμα το αρχείο "test3d.csv", το οποίο περιλαμβάνει όλα τα ξενοδοχεία με τις αντίστοιχες τιμές τους χωρισμένα με κόμμα και θα τα αποθηκεύσει στη μεταβλητή *ds*. Στη συνέχεια, η μέθοδος *sanitizeData* θα πάρει σαν όρισμα το *ds* και θα αφαιρέσει τυχόν γραμμές που περιλαμβάνουν τυχόν NULL τιμές και θα αποθηκεύσει το αποτέλεσμα σε μία νέα μεταβλητή *myData*. Ο λόγος για τον οποίο γίνεται αυτό είναι διότι το σύνολο δεδομένων θα συρρικνωθεί με επανειλημμένους υπολογισμούς συνόλων Παρέτο και χρειαζόμαστε μία μεταβλητή που να περιλαμβάνει όλα τα σημεία για να δημιουργηθεί σωστά το γράφημα. Για τον υπολογισμό πολλαπλών συνόλων Παρέτο, λοιπόν, συνιστάται η χρήση δύο διαφορετικών μεταβλητών για την αποθήκευση του dataset.

```
1  ds <- read.csv(file = "test3d.csv")
2  myData <- sanitizeData(ds)
3
4  preferences <- c("low","low","high")
5
6
7  pareto1 <- calculatePareto(myData,preferences)
8
9  myData <- recalculate(myData,pareto1)
10
11 pareto2 <- calculatePareto(myData,preferences)
12
13 myData <- recalculate(myData,pareto2)
14
15 pareto3 <- calculatePareto(myData,preferences)
16
17 plot1 <- ggplot(ds, aes(x = price, y = distance)) +
18   geom_point(data = pareto1, size = 3) +
19   geom_point(data = pareto2, size = 3) +
20   geom_point(data = pareto3, size = 3) +
21   geom_point(data = ds, size = 1) +
22   geom_step(data=pareto1,direction = "hv",aes(colour="red"))+
23   geom_step(data=pareto2,direction = "hv",aes(colour="blue")) +
24   geom_step(data=pareto3,direction = "hv",aes(colour="green"))
25 plot(plot1)
26
```

Σχήμα 3.3 - Ο κώδικας του παραδείγματος

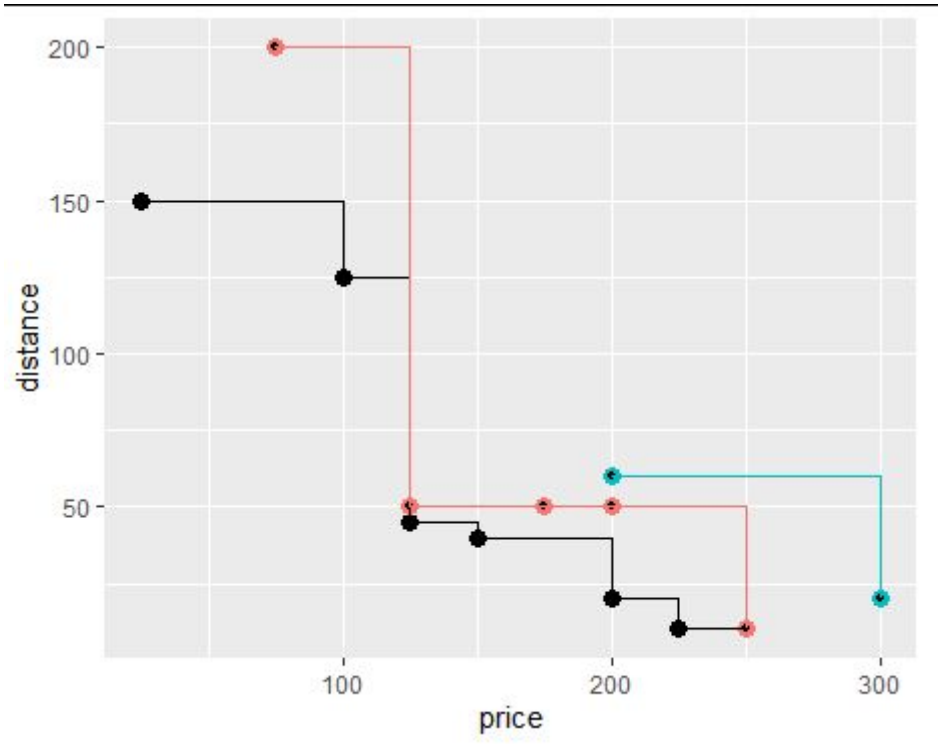
Στη συνέχεια, δημιουργείται το διάνυσμα *preferences* το οποίο περιέχει τις προτιμήσεις του χρήστη. Ο χρήστης πρέπει να προσέχει να προσθέσει τις προτιμήσεις με την σειρά που εμφανίζονται στο σύνολο δεδομένων. Το csv αρχείο (Σχήμα 3.4) έχεις τις στήλες με τη σειρά "τιμή-απόσταση-ποιότητα" και αυτό αντικατοπτρίζεται στη δημιουργία της μεταβλητής.

Στη γραμμή 7 δημιουργείται η μεταβλητή *pareto1* στο οποίο θα επιστραφεί το διάνυσμα με το πρώτο σύνολο Παρέτο (χρησιμοποιώντας απλή κυριαρχία σημείων) και στη γραμμή 9 το *myData* το προσωρινό σύνολο δεδομένων θα προετοιμαστεί για μετέπειτα υπολογισμούς συνόλων Παρέτο.

Τέλος, όπως υποδεικνύεται στο κεφάλαιο 3.3, καλείται η μέθοδος ggplot του πακέτου ggplot2 για τη δημιουργία και προβολή του γραφήματος (Σχήμα 3.5). Στον X και Y άξονα αποτυπώνονται οι διαστάσεις τιμή και απόσταση αντίστοιχα.

```
price,distance,quality
250,10,5
200,50,5
225,10,4
200,20,5
300,20,3
150,40,4
200,60,5
175,50,4
125,50,3
125,45,4
25,150,2
75,200,1
100,125,3
250,10,3
```

Σχήμα 3.4 - Το csv αρχείο που περιλαμβάνει ξενοδοχεία και 3 διαστάσεις: τιμή, απόσταση από τη θάλασσα και ποιότητα.



Σχήμα 3.5 - Το αποτέλεσμα του παραδείγματος

3.5 Απόδοση αλγορίθμου

Χρησιμοποιώντας την εντολή *Sys.time()* πριν και μετά την εκτέλεση του αλγορίθμου και υπολογίζοντας την διαφορά μεταξύ των δύο χρόνων, είναι δυνατόν να βρεθεί ο χρόνος εκτέλεσης συγκεκριμένων σεναρίων. Κάθε σενάριο περιλαμβάνει την ανάγνωση των αρχικών δεδομένων, την εύρεση και διαγραφή NULL τιμών και την εκτέλεση μιας ή πολλαπλών πράξεων Skyline.

Παρακάτω παρατίθεται μία συλλογή από διάφορα σενάρια και ο χρόνος εκτέλεσης του κάθε ενός. Σημειώνεται πως τα σενάρια έτρεξαν σε περιβάλλον RStudio 64-bit σε υπολογιστή εξοπλισμένο με επεξεργαστή AMD Ryzen 5 3600 (6-πύρηνος, βασικός χρονισμός 3.6 GHz) και 16 GB μνήμης. Προφανώς, ο χρόνος εκτέλεσης δύναται να διαφέρει από σύστημα σε σύστημα.

A/A	Dataset	Διαστάσεις	Αριθμός Skylines	Χρόνος περαίωσης
1	hotels (14 εγγραφές)	3	3	0.196552 secs
2	mtcars(32 εγγραφές)	5	8	0.2730601 secs
3	iris (150 εγγραφές)	2	13	2.900663 secs
4	iris (150 εγγραφές)	3	10	3.303755 secs

Πινακας 3.1: Δοκιμές σε μικρά datasets

3.5.1 Απόδοση σε μεγάλα datasets

Τα δεδομένα που χρησιμοποιούνται για αυτή τη δοκιμή είναι το QS World University Rankings 2020 το οποίο περιλαμβάνει βαθμολογίες από το 0 έως το 100 για διάφορες πτυχές πανεπιστημίων. Θα βρεθούν όλα τα Skylines, χρησιμοποιώντας απλή και αυστηρή κυριαρχία σημείων. Μιας και πρόκειται για βαθμολογίες, προτιμάμε τις υψηλότερες τιμές.

A/A	Dataset	Διαστάσεις	Αριθμός Skylines	Χρόνος περαίωσης
1	QS World University Rankings 2020(498 εγγραφές)	6	8	45.84535 secs
2	QS World University Rankings 2020(498 εγγραφές)	6	7(αυστηρή κυριαρχία)	47.44852 secs
3	QS World University Rankings 2020(1019 εγγραφές με 521 NULL)	6	8	46.95145 secs
4	QS World University Rankings 2020(1019 εγγραφές με 521 NULL)	6	7(αυστηρή κυριαρχία)	48.33546 secs

Πινακας 3.2: Δοκιμές σε μεγάλα datasets (με απλή και αυστηρή κυριαρχία σημείων)

3.6 Βελτίωση αλγορίθμου

Παρατηρείται πως σε μεγάλο πλήθος δεδομένων η απόδοση του αλγορίθμου μειώνεται δραστικά. Σε μια προσπάθεια να μειωθούν τα βήματα που απαιτούνται για την εύρεση των καλύτερων σημείων, θα αλλαχθούν οι βοηθητικές μέθοδοι *betterInAtLeastOne* και *betterInAll* για την απλή κυριαρχία, καθώς και η *betterStrict* που χρησιμοποιείται για την αυστηρή κυριαρχία. Η λογική πίσω από τους αλγορίθμους παραμένει η ίδια. Κατά την σύγκριση 2 σημείων, αν σε έστω και μία διάσταση το ένα σημείο αποδειχθεί χειρότερο του άλλου, τότε απορρίπτεται. Οι μεγαλύτερες αλλαγές έγιναν στην δομή αυτών των μεθόδων ώστε να περιοριστούν οι αχρείαστες κλήσεις άλλων μεθόδων.

Για την επίτευξη καλύτερου χρόνου, δημιουργήθηκαν οι μέθοδοι *isBetter* και *isBetterStrict* (για την απλή και αυστηρή μέθοδο κυριαρχίας αντίστοιχα) οι οποίες είναι μικρότερες σε έκταση και δύναται να αποφέρουν τα ίδια αποτελέσματα σε μειωμένο χρόνο.

A/A	Dataset	Διαστάσεις	Αριθμός Skylines	Χρόνος περαίωσης
1	QS World University Rankings 2020(498 εγγραφές)	6	8	37.51747 secs
2	QS World University Rankings 2020(498 εγγραφές)	6	7(αυστηρή κυριαρχία)	33.45454 secs
3	QS World University Rankings 2020(1019 εγγραφές με 521 NULL)	6	8	37.57565 secs
4	QS World University Rankings 2020(1019 εγγραφές με 521 NULL)	6	7(αυστηρή κυριαρχία)	34.85833 secs

Πινακας 3.3: Δοκιμές σε μεγάλα datasets με τη βοήθεια των βελτιωμένων μεθόδων

Κεφάλαιο 4ο: Συμπεράσματα - Προτάσεις βελτίωσης

Αναλύοντας τα αποτελέσματα των σεναρίων στο κεφάλαιο 3.5 εξάγεται το συμπέρασμα πως ο χρόνος περάτωσης των υπολογισμών για την εξαγωγή όλων των Skylines εξαρτάται από τον αριθμό των διαστάσεων και από το πλήθος των γραμμών σε κάθε σύνολο δεδομένων. Αυτό φυσικά ήταν και το αναμενόμενο. Από τον πίνακα 3.1 βλέπουμε πως το σενάριο 4 έχει περίπου 10 φορές περισσότερες γραμμές από το σενάριο 1 και ο χρόνος περάτωσης είναι περίπου 17 φορές μεγαλύτερος για τον ίδιο αριθμό διαστάσεων.

Από τον πίνακα 3.2 εξάγουμε το δεύτερο συμπέρασμα το οποίο αποδεικνύει πως το υπολογιστικό φόρτο μεταξύ της απλής και της αυστηρής κυριαρχίας των σημείων ενός dataset είναι αμελητέο. Η διαδικασία εξαγωγής όλων των γραμμών του dataset που περιείχαν NULL τιμές είναι επίσης πολύ φθηνή διαδικασία, μιας και οι χρόνοι περάτωσης των σεναρίων 1 - 3 και 2 - 4 διαφέρουν ελάχιστα μεταξύ τους.

Για να βελτιωθεί η ταχύτητα της διαδικασίας, αναπτύχθηκαν μέθοδοι για την εύρεση των βέλτιστων σημείων οι οποίες περιέχουν λιγότερες κλήσεις σε άλλες μεθόδους και λιγότερες προσπελάσεις στα πεδία των datasets. Οι νέες μέθοδοι κατάφεραν να επιταχύνουν το αποτέλεσμα, με τον τελικό χρόνο να έχει μειωθεί για ~10 δευτερόλεπτα.

Εν κατακλείδι, το πακέτο που δημιουργήθηκε στα πλαίσια της πτυχιακής αυτής εργασίας είναι ικανό να εξάγει όλα τα πιθανά Skylines από ένα σύνολο δεδομένων, χωρίς να εξαρτάται από εξωτερικές βιβλιοθήκες (πέραν της ggplot2 που χρησιμοποιείται για τη προαιρετική δημιουργία γραφημάτων). Με τη χρησιμοποίηση του πακέτου, ο χρήστης μπορεί να απομακρύνει τυχόν NULL τιμές ή και να θέσει όρια τιμών σε όποιο dataset χρησιμοποιεί. Ο χρήστης μπορεί επίσης να επιλέξει μεταξύ δύο ορισμών για το πως τα σημεία των δεδομένων υπερνικά το ένα το άλλο. Με την απλή κυριαρχία σημείων, επικρατούν τα σημεία τα οποία είναι καλύτερα ή ίσα σε όλες τις διαστάσεις τους από όλα τα άλλα, ενώ με την αυστηρή κυριαρχία επικρατούν τα σημεία που είναι καλύτερα σε όλες τις διαστάσεις.

Ο κώδικας του πακέτου έχει χώρο για βελτιώσεις, παρόλα αυτά. Μπορεί να επιτευχθεί μεγαλύτερη μείωση στο χρόνο περάτωσης της εύρεσης των skylines με την προσθήκη μιας προσωρινής στήλης στο σύνολο δεδομένων. Η στήλη αυτή θα δείχνει αν ένα σημείο του συνόλου δεδομένων βρίσκεται σε Skyline και αν ναι, σε ποιο. Η πρόσβαση στην τιμή αυτής της στήλης είναι πολύ πιο γρήγορη από το να προσπαθήσουμε να βρούμε αν ένα υποψήφιο σημείο έχει ήδη μπει στο διάνυσμα των καλύτερων σημείων με επαναλαμβανόμενες προσπελάσεις σε διανύσματα. Με αυτή τη μεθοδολογία μπορεί να επιτευχθεί ραγδαία επιτάχυνση του αλγορίθμου.

Επιπλέον, οι μοναδικές επιλογές για τον ορισμό της προτίμησης τιμών για τα σημεία είναι “high” ή “low”. Οι μέθοδοι του πακέτου μπορούν να επεκταθούν ώστε να δέχονται σαν προτιμήσεις και boolean τιμές (TRUE/FALSE). Για παράδειγμα, ο χρήστης θα μπορούσε να θέσει ως προτίμηση για την επιλογή των καλύτερων ξενοδοχείων αυτά τα οποία είναι φτηνά (προτίμηση “low”), προσφέρουν υψηλή ποιότητα (προτίμηση “high”), αλλά και που επιτρέπουν τα κατοικίδια ζώα (προτίμηση “TRUE”).

Τέλος, δεν υπάρχουν τρόποι διαχείρισης διπλότυπων εγγραφών. Ο χρήστης που είναι εξοικειωμένος με την R και τις διάφορες λειτουργίες της μπορεί μόνος του να φροντίσει ώστε το dataset που

χρησιμοποιεί να έχει distinct τιμές (π.χ. με τη βοήθεια του δημοφιλές πακέτου dplyr). Παρόλα αυτά, η μέθοδος *sanitizeData* η οποία είναι υπεύθυνη για την απομάκρυνση NULL τιμών μπορεί να επεκταθεί ώστε να διαγράφει τις διπλότυπες γραμμές και να διευκολύνει τον χρήστη.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1].https://www.r-project.org/about.html

[2].https://directory.fsf.org/wiki/R

[3].https://cran.r-project.org/doc/FAQ/R-FAQ.html#What-is-CRAN_003f

[4].https://www.oreilly.com/library/view/r-in-a/9781449358204/ch04s04.html

[5].https://cran.r-project.org/doc/manuals/r-release/R-exts.html

[6].https://support.rstudio.com/hc/en-us/articles/200486488-Developing-Packages-with-RStudio

[7].http://www.cs.ucr.edu/~ravi/CS236Papers/skyline-operator.pdf

[8].https://openproceedings.org/2015/conf/edbt/paper-192.pdf

ΠΑΡΑΡΤΗΜΑ Α : Πηγαίος Κώδικας

Μπορείτε να κατεβάσετε το πακέτο για το Skyline operator, καθώς και να δείτε τον πηγαίο κώδικα στη σελίδα μου στο GitHub.