ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Application of Machine Learning Techniques on Traffic Data for Customer's Segmentation, Churn Prediction and Customer's Lifetime Value Evaluation

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Βασίλειος Γ. Κιούσης

Επιβλέπων : Παναγιώτης Τσανάκας
Καθηγητής Ε.Μ.Π

Αθήνα, Μάρτιος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Application of Machine Learning Techniques on Traffic Data for Customer's Segmentation, Churn Prediction and Customer's Lifetime Value Evaluation

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Βασίλειος Γ. Κιούσης

Επιβλέπων : Παναγιώτης Τσανάκας
Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την
3η Μαρτίου 2022.

.....	.....	.....
Παναγιώτης Τσανάκας Καθηγητής Ε.Μ.Π	Δημήτριος Κουτσούρης Καθηγητής Ε.Μ.Π	Γεώργιος Ματσόπουλος Καθηγητής Ε.Μ.Π

Αθήνα, Μάρτιος 2022

.....
Βασίλειος Γ. Κιούσης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Στο κλάδο των επιχειρήσεων, η διατήρηση και απόκτηση νέων πελατών αποτελούν έναν από τους σημαντικότερους τομείς ανάπτυξης για μια κερδοφόρα μελλοντική πορεία. Για να το κάνει αυτό, μια επιχείρηση χρειάζεται να διαθέτει μετρικές όπως ο προσδιορισμός την αξίας των πελατών έτσι ώστε να έχει πληρέστερη εικόνα για τη δυναμική και τη κατανομή του πελατειακού κοινού της, να εφαρμόζει κατάλληλους διαχωρισμούς των τελευταίων βάσει των αναγκών του αλλά και της ίδιας, όπως επίσης και να χρησιμοποιεί μοντέλα πρόβλεψης για πιθανή μελλοντική διακοπή της σχέσης τους.

Έτσι, η ανάγκη για ένα συστηματικό τρόπο αξιολόγησης αποφάσεων που αφορούν σχέσεις εταιρίας-πελατών και περικλείουν τις παραπάνω ενέργειες, οδηγεί στην υιοθέτηση συστημάτων Διαχείρισης Πελατειακών Σχέσεων. Ο στόχος της παρούσας διπλωματικής εργασίας ήταν η ανάπτυξη μέρους ενός συστήματος Διαχείρισης Πελατειακών Σχέσεων για δεδομένα που αφορούσαν χρήστες ηλεκτρονικού εισιτηρίου σε οδικά δίκτυα. Πιο συγκεκριμένα το σύστημα αυτό συλλέγει, επεξεργάζεται, αναλύει δεδομένα χρηστών της Α.Ε. Αττικές Διαδρομές και εφαρμόζοντας σε αυτά τεχνικές αλγορίθμων μηχανικής μάθησης, εξειδικεύεται στο προσδιορισμό των εξής τριών μεγεθών: Στην τμηματοποίηση των πελατών με χρήση 2 μεθόδων και σύγκριση τους, στη πρόβλεψη μετάπτωσης πελατών σε αδράνεια βάσει επιλογής κατάλληλου αλγορίθμου μηχανικής μάθησης και τέλος, στο προσδιορισμό της αξίας των πελατών, διαδικασία η οποία χρησιμοποιεί στοιχεία από ανάλυση επιβίωσης αλλά και τις προηγούμενες διαδικασίες.

Λέξεις κλειδιά:

Σύστημα Διαχείρισης Πελατειακών Σχέσεων, ηλεκτρονικό εισιτήριο, οδικά δίκτυα, ανάλυση δεδομένων, μηχανική μάθηση, αλγόριθμοι, τμηματοποίηση πελατών, πρόβλεψη μετάπτωσης σε αδράνεια, αξία ζωής πελατών.

Abstract

In the business sector, retaining and acquiring new customers is one of the most important areas regarding growth for a company's profitable future. To accomplish this, firms need to possess metrics such as determining customer's lifetime value, so that they have better understanding of the dynamics and distribution of their customer audience, to apply appropriate segregations of the latter based on its needs and the company's , as well as to use prediction models for possible future termination of their relationship.

Thus, the need for a systematic way of evaluating decisions that concern company-customer relationships and include the above actions, leads to the adoption of Customer Relationship Management systems. The aim of this diploma thesis was to develop part of a Customer Relationship Management system for data relating to e-ticket users on road networks. In concrete terms, this system collects, processes, analyzes data of the firm's Αττικές Διαδρομές A.E users, and by applying machine learning algorithm techniques to them, it specializes in identifying the following three variables: Customer segmentation utilizing 2 methods and comparing them, churn prediction based on the selection of an appropriate machine learning algorithm and finally, evaluation of customer's lifetime value, a process that derives from survival analysis as well as is built upon previous procedures.

Keywords:

customer relationship management system, e-pass ticket, road networks, data analysis, machine learning, algorithms, customer's segmentation, churn prediction, customer's lifetime value evaluation.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον καθηγητή κ. Παναγιώτη Τσανάκα για την υποστήριξη και καθοδήγηση που μου προσέφερε, καθώς και για την ευκαιρία που μου έδωσε να εκπονήσω την διπλωματική μου στο συγκεκριμένο θέμα.

Ιδιαίτερες ευχαριστίες απευθύνονται στον μεταδιδακτορικό ερευνητή Βρεττό Μουλό, ο οποίος με την εμπειρία και τις συμβουλές του με βοήθησε να ορίσω την κατεύθυνση της έρευνάς μου και να την ολοκληρώσω με επιτυχία.

Ακόμη, θα ήθελα να ευχαριστήσω τον υποψήφιο διδάκτορα Άγγελο Κολαΐτη για την πολύτιμη βοήθειά του στο κομμάτι ανάπτυξης της Βάσης Δεδομένων.

Τέλος, θέλω να εκφράσω την αμέριστη ευγνωμοσύνη πρωτίστως στην οικογένειά μου και ύστερα στους φίλους μου, για τη συνεχή και κάθε είδους υποστήριξη σε όλα τα χρόνια των σπουδών μου.

Περιεχόμενα
Table of Contents

Κεφάλαιο 1
Εισαγωγή

1.1 Η Γενική Εικόνα

Η διατήρηση αλλά απόκτηση νέων πελατών αποτελούν έναν από τους σημαντικότερους στόχους μιας επιχείρησης ή ενός οργανισμού. Ο ανταγωνισμός, οι γρήγοροι ρυθμοί της καθημερινότητας, η αύξηση του βιοτικού επιπέδου είναι μόνο μερικοί από τους λόγους για τους οποίους η «συμπεριφορά» των πελατών παρουσιάζει έντονες διακυμάνσεις. Ο όρος «συμπεριφορά», αναφέρεται στοχευμένα στο εάν και κατά πόσο ένας πελάτης παραμένει πιστός στην εταιρία (Customer's Loyalty), δηλαδή εάν συνεχίζει να καταναλώνει τα προϊόντα ή τις υπηρεσίες που παρέχει η εταιρία και με ποια συχνότητα.

Η ανάγκη για ένα συστηματικό τρόπο αξιολόγησης και κατ' επέκταση πρόβλεψης συμπεριφοράς του πελάτη, ώθησε ήδη από τις αρχές της δεκαετίας του χίλια εννιακόσια εβδομήντα, στην ανάπτυξη του τομέα της Διαχείρισης Πελατειακών Σχέσεων (CRM). Η τελευταία αποτελεί το σύνολο των διαδικασιών κατά το οποίο μια επιχείρηση ή ένας οργανισμός, διαχειρίζεται τις αλληλεπιδράσεις μεταξύ της ίδιας και των πελατών της, με σκοπό την μεγιστοποίηση του κέρδους. Χρησιμοποιώντας κατά κόρον τεχνικές ανάλυσης δεδομένων, ένα σύστημα Διαχείρισης Πελατειακών Σχέσεων συλλέγει και καταγράφει δεδομένα των πελατών, στη συνέχεια τα επεξεργάζεται κατάλληλα, εξάγοντας μοτίβα. Η ορθή ερμηνεία των τελευταίων, δίνει πολλές φορές τη δυνατότητα στην εταιρία να προσαρμοστεί καλύτερα στις ανάγκες των πελατών, προσφέροντας έτσι θελκτικότερα προϊόντα ή υπηρεσίες. Θελκτικότερα προϊόντα ή υπηρεσίες φυσικά, θέτουν τα θεμέλια για μακροχρόνιες και σταθερότερες πελατειακές σχέσεις, συνεπώς και υψηλότερα κέρδη.

Εμβαθύνοντας στις πελατειακές σχέσεις, είναι κομβικής σημασίας για μια εταιρία να διαθέτει όσο το δυνατό μεγαλύτερο πλήθος σταθερών και «καλών» πελατών. Κατά αυτόν τον τρόπο εισέρχεται η έννοια της αξίας του πελάτη (CLV ή LTV). Η εταιρία συνεπώς χρειάζεται μετρικές οι οποίες προσδιορίζουν την αξία ενός πελάτη, ούτως ώστε να υπάρχει πληρέστερη εικόνα για τη δυναμική και τη κατανομή του πελατειακού κοινού της. Μέσω της σωστής διαδικασίας διαχωρισμού των πελατών (Customer's Segmentation) σε τμήματα, η εταιρία πλέον μπορεί να εμβαθύνει ακόμα περισσότερο σε κάθε group, δημιουργώντας για εκείνο κατάλληλα πακέτα και προσφορές. Ωστόσο, όσο καλό και προσαρμοσμένο στις ανάγκες του πελάτη είναι ένα προϊόν ή υπηρεσία, πάντα θα υπάρχει μερίδα πελατών που θα σταματήσει να το χρησιμοποιεί, είτε με το να καταφεύγει στον ανταγωνισμό, είτε παύοντας να δείχνει πλέον ενδιαφέρον για εκείνο. Άρα, η διαδικασία πρόβλεψης της μερίδας των πελατών που δεν είναι πλέον ενεργοί για την εταιρία (Churn Prediction) καθίσταται απαραίτητη, μια και προσδιορίζει άμεσα τη ζημία της εταιρίας.

1.2 Αντικείμενο της διπλωματικής

Κύρια έμπνευση για την εκπόνηση της παρούσας διπλωματικής εργασίας αποτέλεσε η πρόσβαση σε δεδομένα χρηστών ηλεκτρονικού εισιτηρίου (e-pass) σε οδικά δίκτυα και συγκεκριμένα, σε δεδομένα χρηστών της εταιρίας Αττικές Διαδρομές Α.Ε.. Η επαφή με τα δεδομένα αυτά, οδήγησε στη σύλληψη της κεντρικής ιδέας, η οποία ήταν η δημιουργία μέρους ενός συστήματος Διαχείρισης Πελατειακών Σχέσεων (CRM).

Ένα κομμάτι συστήματος δηλαδή το οποίο θα συλλέγει, θα επεξεργάζεται και θα αναλύει δεδομένα χρηστών των Αττικών Διαδρομών σε πρώτη φάση και στη συνέχεια θα ειδικεύεται σε πιο συγκεκριμένες λειτουργίες για την εξαγωγή χρήσιμων συμπερασμάτων που αφορούν το συγκεκριμένο πελατειακό κοινό. Όπως προαναφέρθηκε, ο προσδιορισμός μεγεθών όπως η αξία του πελάτη (CLV), η τμηματοποίηση του πελατειακού κοινού βάσει προβλεπόμενων κριτηρίων (Customer's Segmentation) και η πρόβλεψη μετάπτωσης πελατών σε αδράνεια (Churn Prediction) είναι εργασίες ζωτικές για τη καλή λειτουργία μιας επιχείρησης. Η υψηλή συσχέτιση μεταξύ των τριών αυτών tasks και το γεγονός ότι υπάγονται στο γενικό κλάδο της Διαχείρισης Πελατειακών Σχέσεων αποτέλεσε και τα κριτήρια επιλογής τους για τη μελέτη και το προσδιορισμό τους βάσει των δεδομένων που διατέθηκαν.

1.3 Δομή Εργασίας

Το κείμενο της διπλωματικής εργασίας αποτελείται από 8 κεφάλαια:

Στο κεφάλαιο 2 παρουσιάζεται το θεωρητικό υπόβαθρο σε ότι αφορά τη μηχανική μάθηση, καθώς υπήρξε εκτενής χρήση σχετικών αλγορίθμων και τεχνικών για την επίτευξη του στόχων. Συγκεκριμένα εξετάζεται η μηχανική μάθηση ως έννοια και χρησιμότητα, παρουσιάζονται οι διάφορες κατηγορίες των μεθόδων της, ενώ συνοπτική αναφορά γίνεται και σε κάθε αλγόριθμο μηχανικής μάθησης που χρησιμοποιήθηκε.

Στο κεφάλαιο 3 γίνεται παρουσίαση των εργαλείων που χρησιμοποιήθηκαν για και κατά την εκτέλεση του κώδικα, όπως και τη παρουσίαση αποτελεσμάτων του συστήματος. Συγκεκριμένα αναφέρεται η γλώσσα προγραμματισμού Python στην οποία γράφτηκε ο κώδικας, όπως και οι κύριες βιβλιοθήκες που χρησιμοποιήθηκαν για την ανάλυση δεδομένων (pandas, numpy), τη εφαρμογή αλγορίθμων μηχανικής μάθησης (scikit-learn), όπως και βιβλιοθήκες απεικόνισης αποτελεσμάτων (seaborn, matplotlib). Επιπρόσθετα, παρουσιάζεται το περιβάλλον εκτέλεσης του κώδικα, δηλαδή το Jupyter Notebook, ενώ τέλος γίνεται αναφορά στη βάση δεδομένων TimescaleDB, η οποία χρησιμοποιήθηκε για τη πληρότητα του συστήματος.

Στο κεφάλαιο 4 γίνεται μια εισαγωγή στις Αττικές Διαδρομές Α.Ε. και παρουσιάζονται τα δεδομένα στην αρχική τους μορφή. Στη συνέχεια, περιγράφονται πλήρως όλες οι διεργασίες συλλογής, ομαδοποίησης, επεξεργασίας, ανάλυσης και απεικόνισης αυτών, με τη σειρά που υλοποιήθηκαν.

Στο κεφάλαιο 5 αναλύεται και προσδιορίζεται ο διαχωρισμός του πελατειακού κοινού (Customer's Segmentation). Πρώτα πρώτα ορίζεται η έννοια της τμηματοποίησης των πελατών, παρουσιάζοντας τα κριτήρια διαχωρισμού, δηλαδή το μοντέλο RFM. Μετέπειτα αναλύονται τα βήματα που ακολουθήθηκαν για τη δημιουργία των τμημάτων, καταλήγοντας σε γραφικές αναπαραστάσεις των αποτελεσμάτων, ερμηνεύοντάς τες. Στη συνέχεια εξετάζεται εναλλακτικός τρόπος εύρεσης της πελατειακής κατανομής χρησιμοποιώντας μη-επιβλεπόμενο αλγόριθμο μηχανικής μάθησης (k-means clustering), συγκρίνοντας εποπτικά τα αποτελέσματα των 2 μεθόδων.

Στο κεφάλαιο 6 παρουσιάζεται και υπολογίζεται η πιθανότητα μετάπτωσης πελατών (ενεργών χρηστών) σε αδράνεια (Churn Prediction). Ξεκινώντας, εισάγεται αναλυτικότερα η παραπάνω έννοια, ενώ ακόμα παρουσιάζονται τα οφέλη του υπολογισμού της για μια εταιρία ή οργανισμό. Ακολουθούν έπειτα αναλυτικά τα βήματα υπολογισμού της πιθανότητας αυτής, στα οποία περιλαμβάνεται χρήση πολλαπλών επιβλεπόμενων αλγορίθμων μηχανικής μάθησης (classification). Αφότου επιλέγεται συγκριτικά ο πλέον κατάλληλος, τροποποιείται περαιτέρω για καλύτερη απόδοση, προσδίδοντας ακριβέστερο τελικό αποτέλεσμα.

Στο κεφάλαιο 7 εξετάζεται και υπολογίζεται η αξία των πελατών (Customer's Lifetime Value). Αρχικά ορίζεται η έννοια της αξίας του πελάτη, παρουσιάζοντας επίσης και τα οφέλη που παρέχει στην εταιρία ή οργανισμό ο υπολογισμός της. Στη συνέχεια εισάγεται η βιβλιοθήκη lifetimes, μια βιβλιοθήκη που παρέχει κατάλληλες συναρτήσεις και εργαλεία απεικόνισης για τον υπολογισμό της αξίας του πελάτη. Ακολούθως περιγράφονται αναλυτικά όλα τα βήματα που ακολουθήθηκαν για την εύρεση του CLV, συμπεριλαμβάνοντας γραφικές αναπαραστάσεις σε αρκετά στάδια για καλύτερη εποπτεία των αποτελεσμάτων. Τέλος, κατά τη διαδικασία εύρεσης του CLV προκύπτει εναλλακτικός τρόπος υπολογισμού του Customer's Churn, οπότε γίνεται σύγκριση των δυο μεθόδων.

Τέλος, στο κεφάλαιο 8 προτείνονται μελλοντικές προσθήκες που θα μπορούσαν να βελτιώσουν την εφαρμογή.

Ο κώδικας που εκτελέστηκε για την φόρτωση, επεξεργασία και ανάλυση δεδομένων, το κομμάτι του διαχωρισμού πελατών, το κομμάτι πρόβλεψης μετάπτωσης πελατών σε αδράνεια και το κομμάτι υπολογισμού αξίας πελάτη παρατίθενται στο τέλος του εγγράφου, στα Παραρτήματα Α, Β, Γ και Δ αντίστοιχα. Ακόμη, στο Παράρτημα Ε παρατίθεται ο εναλλακτικός τρόπος φόρτωσης των δεδομένων μέσω της βάσης δεδομένων TimescaleDB ενώ στο Παράρτημα Στ' αναγράφεται η δημιουργία του κατάλληλου TABLE που αξιοποίησε η βάση.

Κεφάλαιο 2
Θεωρητικό Υπόβαθρο

2.1 Μηχανική Μάθηση

2.1.1 Ορισμός

Η μηχανική μάθηση αποτελεί κομμάτι της επιστήμης των υπολογιστών. Είναι η επιστήμη του προγραμματισμού υπολογιστών με τέτοιο τρόπο ώστε να μαθαίνουν από τα δεδομένα. Σύμφωνα με τον Arthur Samuel, Αμερικανό πρωτοπόρο σε θέματα τεχνητής νοημοσύνης:

“ Η μηχανική μάθηση είναι το πεδίο μελέτης που δίνει στους υπολογιστές τη δυνατότητα να μαθαίνουν χωρίς να έχουν ρητά προγραμματιστεί ”

Ένας λιγότερο αφηρημένος ορισμός της μηχανικής μάθησης δίνεται από τον Tom Mitchell, Αμερικανό επιστήμονα υπολογιστών, όπου αναφέρει:

“ Ένα πρόγραμμα υπολογιστή λέγεται ότι μαθαίνει από μία εμπειρία E σχετικά με ένα έργο T και μία μετρική απόδοσης P, αν η απόδοση του στο T, όπως μετριέται από την μετρική P, βελτιώνεται από την εμπειρία E.”

2.1.2 Κατηγοριοποίηση Μεθόδων Μηχανικής Μάθησης

Τα συστήματα μηχανικής μάθησης μπορούν σε γενικές γραμμές να κατηγοριοποιηθούν με τους εξής τρόπους:

- Βάσει του αν η εκμάθηση τους γίνεται με επίβλεψη από τον άνθρωπο (Επιβλεπόμενη μάθηση, Μη επιβλεπόμενη μάθηση, Ημί-επιβλεπόμενη μάθηση, Ενισχυτική μάθηση).

- Βάσει του αν η εκμάθηση γίνεται σταδιακά ή επί τόπου (Batch learning ή Online learning).

- Βάσει του τρόπου λειτουργίας τους, δηλαδή είτε συγκρίνοντας νέα δεδομένα με ήδη γνωστά, είτε ανακαλύπτοντας μοτίβα σε δεδομένα εκμάθησης, χτίζοντας έτσι ένα μοντέλο πρόβλεψης. (Instance based ή model based learning).

2.1.3 Χρησιμότητα

Οι τεχνικές μηχανικής μάθησης αναδεικνύουν τη χρησιμότητα τους σε περιπτώσεις όπου τα δεδομένα ακολουθούν κάποιο μοτίβο, το οποίο διαφοροποιείται με την πάροδο του χρόνου ή είναι ιδιαίτερα πολύπλοκο, με αποτέλεσμα να χρειάζεται πολυάριθμους υπολογισμούς, πέρα από την αντίληψη μας. Σε τέτοιες περιπτώσεις είναι αδύνατο για τους προγραμματιστές να προγραμματίσουν ρητά κάποια λύση, ενώ οι τεχνικές μηχανικής μάθησης με την ικανότητα τους να «μαθαίνουν» από την εμπειρία μπορούν να προσφέρουν τις λύσεις αυτές.

2.2 Αλγόριθμοι Μηχανικής Μάθησης που χρησιμοποιήθηκαν

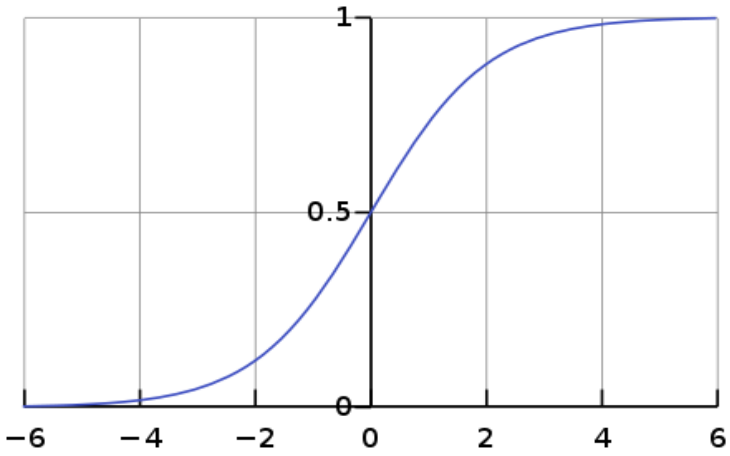
2.2.1 Logistic Regression

Logistic regression ή στα ελληνικά Λογιστική Παλινδρόμηση είναι ένα μοντέλο δυαδικής Κατάταξης (binary Classification) που υπάγεται στη κατηγορία αλγορίθμων επιβλεπόμενης μάθησης. Αναλυτικότερα, στη Λογιστική Παλινδρόμηση ένα γραμμικό μοντέλο της μορφής $\beta_0 + \beta_1 \chi$ περιλαμβάνεται σε μια λογιστική συνάρτηση (αλλιώς και σιγμοειδής) της μορφής $\frac{1}{1+e^{-z}}$, τέτοια ώστε:

$$P(y_i = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 \chi)}}$$

Όπου $P(y_i = 1 \mid X)$ είναι η πιθανότητα της i-οστής παρατήρησης της μεταβλητής απόκρισης, y_i, να ανήκει στη κλάση 1, X είναι τα δεδομένα εκπαίδευσης και β_0, β_1 παράμετροι προς υπολογισμό από το ίδιο το μοντέλο.

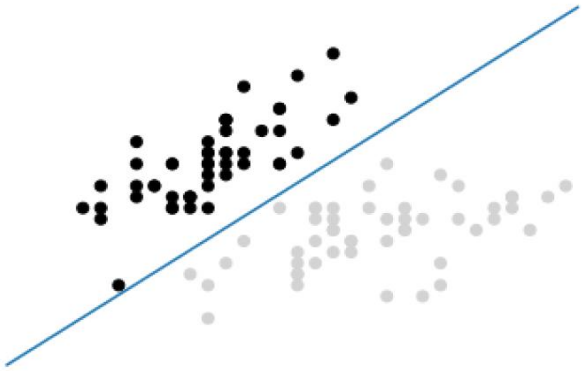
Η επίδραση της συνάρτησης αυτής είναι να περιορίσει το σύνολο τιμών της εξόδου από 0 έως 1, έτσι ώστε να μπορεί να ερμηνευτεί σαν πιθανότητα. Δηλαδή, εάν $P(y_i = 1 \mid X)$ έχει τιμή μεγαλύτερη από 0.5, αποδίδεται η κλάση 1, διαφορετικά η κλάση 0. Σχηματικά για τη σιγμοειδή συνάρτηση έχουμε:



2.2.2 Support Vector Machines (Linear SVC)

Support Vector Machines ή αλλιώς Μηχανές Διανυσμάτων Υποστήριξης είναι μια οικογένεια αλγορίθμων επιβλεπόμενης μάθησης που χρησιμοποιούνται σε προβλήματα κατάταξης άλλα και παλινδρόμησης. Για τη κατανόηση της λειτουργίας του αλγορίθμου, απαιτείται η κατανόηση της έννοιας του 'Υπερεπιπέδου'. Υπερεπίπεδο είναι ο υποχώρος ή υποπεριοχή, διάστασης μιας λιγότερης από τον περιβάλλοντα χώρο. Οι αλγόριθμοι SVC συνεπώς, αναλόγως του πλήθους των κλάσεων που υπάρχουν από δεδομένα εκπαίδευσης, προσπαθούν να υπολογίσουν το Υπερεπίπεδο, το οποίο θα περιέχει το μεγαλύτερο περιθώριο ανάμεσα στις κλάσεις.

Για παράδειγμα, για δεδομένα εκπαίδευσης που περιλαμβάνουν 2 κλάσεις, το υπερεπίπεδο θα είναι μια ευθεία και συγκεκριμένα η ευθεία που τα στοιχεία της μίας κλάσης απέχουν τη μεγαλύτερη δυνατή απόσταση από τα στοιχεία της έτερης κλάσης. Σχηματικά, θα μπορούσε να έχει την εξής μορφή:



Στη παραπάνω απεικόνιση όλες οι παρατηρήσεις για τη κάθε κλάση φέρουν το ίδιο χρώμα (μαύρο για τη μία, γκρίζο για την άλλη). Συνεπώς κάθε νέο στοιχείο που τοποθετείται κάτω δεξιά της γραμμής θα ανήκει στη κλάση με τα γκρίζα στοιχεία, ενώ αν βρίσκεται πάνω αριστερά της, θα ανήκει στη κλάση με τα μαύρα στοιχεία.

2.2.3 K-Nearest Neighbors Classifier

Οι Κ-εγγύτεροι γείτονες αποτελούν έναν από τους πιο απλούς, αλλά και συνάμα πιο διαδεδομένους αλγορίθμους κατάταξης επιβλεπόμενης μάθησης. Αναλυτικότερα, ο αλγόριθμος αυτός υπολογίζει τις αποστάσεις μεταξύ των στοιχείων προς ταξινόμηση των ήδη ταξινομημένων στοιχείων και επιλέγει εκείνα τα στοιχεία πλήθους Κ, τα οποία βρίσκονται πλησιέστερα στο στοιχείο προς ταξινόμηση.

Η απόσταση μεταξύ των στοιχείων μπορεί να προσδιοριστεί βάσει διάφορων ορισμών όπως:

- Ευκλείδεια: $d_{euclidean} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$
- Manhattan: $d_{manhattan} = \sum_{i=1}^{n} |x_i - y_i|$
- Minkowski: $d_{minkowski} = (\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p}$

Προκειμένου ο αλγόριθμος να είναι αποδοτικός, είναι πολύ σημαντικός ο προσδιορισμός κατάλληλου ακεραίου Κ. Αυτό μπορεί να γίνει με διαδοχικές δοκιμές ξεκινώντας από μικρότερο ακέραιο σε μεγαλύτερο. Για κάθε μια περίπτωση υπολογίζεται το σφάλμα που προέκυψε και αφού ολοκληρωθούν οι δοκιμές, αναπαρίστανται γραφικά ο αριθμός Κ συναρτήσει του σφάλματος. Στο σημείο που η γραφική παρουσιάζει τη πιο απότομη κύρτωση (Καμπύλη Γονάτου) θεωρείται το ιδανικό σημείο για την επιλογή του ακεραίου Κ.

2.2.4 K-Means Clustering

Ο αλγόριθμος K-means είναι ο διασημότερος αλγόριθμος συσταδοποίησης (Clustering), μιας μεθόδου μη επιβλεπόμενης μηχανικής μάθησης. Δηλαδή σε αντίθεση με τους ανωτέρω αλγορίθμους που διέθεταν ετικέτες και προκαθορισμένο αριθμό κλάσεων, εδώ οι κλάσεις και ο διαχωρισμός καθορίζονται από τον ίδιο τον αλγόριθμο, δυναμικά.

Αναλυτικότερα, ο αλγόριθμος πρωτίστως δημιουργεί Κ ‘κέντρα’ σε τυχαίες τοποθεσίες. Στη συνέχεια για κάθε νέα παρατήρηση υπολογίζονται η απόσταση μεταξύ της τελευταίας και των Κ ‘κέντρων’, αποδίδοντας στην εκάστοτε παρατήρηση την ομάδα με το κοντινότερο κέντρο. Τα σημεία που βρίσκονται τα κέντρα, με τη προσθήκη νέων παρατηρήσεων ‘μετακινούνται’ στο μέσο κάθε ομάδας (cluster) που δημιουργείται. Όπως και στο K-neighbors έτσι και εδώ παίζει σημαντικό ρόλο ο προσδιορισμός του βέλτιστου ακεραίου Κ. Ακολουθώντας την ίδια διαδικασία με πριν, υπολογίζεται το βέλτιστο Κ.

2.2.5 Decision Tree Classifier

Οι αλγόριθμοι Δένδρων Απόφασης ανήκουν στην ευρύτερη κατηγορία αλγορίθμων επιβλεπόμενης μηχανικής μάθησης, οι οποίοι χρησιμοποιούνται τόσο σε προβλήματα κατάταξης όσο και παλινδρόμησης. Η βάση των αλγορίθμων αυτών είναι μια σειρά από κανόνες απόφασης, οι οποίοι στο σύνολό τους αναπαριστούν ένα γράφο στον οποίο διακρίνονται τα εξής:

- Ένας αρχικός κόμβος, η ρίζα
- Οι εσωτερικοί κόμβοι
- Οι εξωτερικοί κόμβοι, τα φύλλα

Σε κάθε εσωτερικό κόμβο αντιστοιχεί ένα χαρακτηριστικό που χρησιμοποιείται για περαιτέρω διαχωρισμό του δέντρου, ενώ για κάθε κόμβο (είτε εσωτερικό είτε εξωτερικό) αντιστοιχεί μια συνθήκη ελέγχου με βάση το διαχωριστικό χαρακτηριστικό. Συνεπώς η διαδικασία κατασκευής ενός δέντρου απόφασης είναι επαναληπτική και περιγράφεται ως ακολούθως: Αρχικά, επιλέγεται ένα χαρακτηριστικό, το οποίο αναφέρεται στη ρίζα του δέντρου, και στη συνέχεια κατασκευάζεται μια ακμή και ένας κόμβος για καθεμία από τις διακριτές τιμές του

χαρακτηριστικού. Αυτά τα δύο βήματα επαναλαμβάνονται συνεχώς, μέχρις ότου όλα τα χαρακτηριστικά να εισαχθούν στους κόμβους του δέντρου.

2.2.6 Random Forest Classifier

Στηριζόμενο στον αλγόριθμο Δέντρου Απόφασης, ο αλγόριθμος Τυχαίου Δάσους αποτελεί ένα πιο αποτελεσματικό αλγόριθμο κατάταξης επιβλεπόμενης μηχανικής μάθησης, αφού δημιουργεί και στη συνέχεια συνδυάζει πολλαπλά Δέντρα Απόφασης, προσφέροντας έτσι σταθερότερες και ακριβέστερες προβλέψεις.

Πιο αναλυτικά, ο αλγόριθμος Τυχαίου Δάσους αντί να ψάχνει το πιο σημαντικό χαρακτηριστικό ώστε βάσει αυτό να χωρίσει τους κόμβους, διαλέγει το σημαντικότερο χαρακτηριστικό από ένα τυχαίο υποσύνολο όλων των χαρακτηριστικών. Αυτό έχει ως αποτέλεσμα την εμφάνιση περισσότερων ασυσχέτιστων αποτελεσμάτων (Δένδρων), προσφέροντας μεγαλύτερη ποικιλία και δημιουργώντας τελικά καλύτερο μοντέλο πρόβλεψης.

Κεφάλαιο 3
Παρουσίαση Εργαλείων

3.1 Η Γλώσσα Προγραμματισμού Python

Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε για την εκπόνηση της διπλωματικής εργασίας ήταν η Python. Η γλώσσα προγραμματισμού Python δημιουργήθηκε από το Guido van Rossum και κυκλοφόρησε για το κοινό για πρώτη φορά το 1991. Αποτελεί διερμηνευόμενη (interpreted) γλώσσα υψηλού επιπέδου και γενικού σκοπού. Υποστηρίζει πολλαπλά προγραμματιστικά υποδείγματα όπως δομημένο προγραμματισμό, προστακτικό προγραμματισμό, αντικειμενοστραφή αλλά και συναρτησιακό προγραμματισμό. Η έκδοση 2.0 της Python κυκλοφόρησε το 2000 ενώ η 3.0 έκδοση, το 2008. Βασικοί πυλώνες στους οποίους βασίστηκε ο σχεδιασμός της γλώσσας αποτελούν η εύκολη ανάγνωση του κώδικά της καθώς και η ευκολία της χρήσης της. Η Python αποτελεί βασική επιλογή για την ανάπτυξη εφαρμογών μηχανικής μάθησης καθώς και ανάλυσης δεδομένων.

3.2 Κύριες Βιβλιοθήκες

3.2.1 Pandas

Η βιβλιοθήκη Pandas αποτέλεσε το βασικότερο εργαλείο για την ανάλυση δεδομένων της εργασίας. Η βιβλιοθήκη αυτή αναπτύχθηκε από τον προγραμματιστή λογισμικού Wes McKinney το 2008 κατά τη παραμονή του στην εταιρία διαχείρισης επενδύσεων AQR Capital Management. Από τα τέλη του 2009 η βιβλιοθήκη έχει γίνει ανοιχτού κώδικα, επιτρέποντας έτσι σε πολλούς προγραμματιστές παγκοσμίως να προτείνουν διορθώσεις και επεκτάσεις.

Το βασικό πλεονέκτημα των Pandas είναι η συμπερίληψη των DataFrame αντικειμένων, δηλαδή δισδιάστατες δομές δεδομένων με στήλες ιδίου ή διαφορετικού τύπου δεδομένων, τα οποία χάρη στη ταχύτητα και την ευελιξία τους, διευκολύνουν σημαντικά στις διαδικασίες προετοιμασίας, καθαρισμού και διαχωρισμού των δεδομένων.

3.2.2 NumPy

Η βιβλιοθήκη NumPy αποτελεί το θεμελιώδες πακέτο της γλώσσας προγραμματισμού Python για επιστημονικούς υπολογισμούς. Πιο συγκεκριμένα, η βιβλιοθήκη αυτή προσφέρει τεράστια ποικιλία μαθηματικών συναρτήσεων και υπολογιστικών πράξεων πάνω σε πολυδιάστατους πίνακες. Τυπικά, αυτές οι πράξεις εκτελούνται αποδοτικότερα και με χρήση λιγότερου κώδικα από ενσωματωμένες συναρτήσεις και λειτουργίες της Python.

3.2.3 Scikit-learn

Η Scikit-learn είναι μια ελεύθερου λογισμικού βιβλιοθήκη μηχανικής μάθησης για τη γλώσσα προγραμματισμού Python. Περιλαμβάνει πληθώρα αλγορίθμων μηχανικής μάθησης, συμπεριλαμβανομένων των όσων προαναφέρθηκαν και χρησιμοποιήθηκαν στη παρούσα διπλωματική εργασία, ενώ μάλιστα είναι σχεδιασμένη ώστε να είναι συμβατή με αριθμητικές και επιστημονικές βιβλιοθήκες όπως η NumPy.

3.2.4 Matplotlib & Seaborn

Η Matplotlib αποτελεί μια ολοκληρωμένη βιβλιοθήκη για την απεικόνιση στατικών, κινούμενων αλλά και διαδραστικών γραφικών αναπαραστάσεων για τη γλώσσα προγραμματισμού Python. Η Seaborn είναι και εκείνη μια βιβλιοθήκη γραφικής αναπαράστασης δεδομένων για τη γλώσσα προγραμματισμού Python, η οποία στηρίζεται στη Matplotlib.

Στα πλαίσια της διπλωματικής, αμφότερες οι βιβλιοθήκες Seaborn και Matplotilib χρησιμοποιήθηκαν εκτενώς και, αναλόγως το μέγεθος και τη συγκεκριμένη απεικόνιση, επιλεγόταν η καταλληλότερη. Για πολυπλοκότερες απεικονίσεις επιλέχθηκε η Seaborn λόγω της μεγάλης ποικιλίας συγκεκριμένων γραφημάτων, ενώ για απλούστερες η Matplotlib λόγω της μεγάλης ευελιξίας, απλότητας αλλά και απόδοσης που διαθέτει.

3.3 Jupyter Notebook

Το Jupyter Notebook αποτελεί ένα web-based, διαδραστικό υπολογιστικό περιβάλλον για τη δημιουργία εγγράφων, το οποίο αναπτύχθηκε από τον μη κερδοσκοπικό οργανισμό Project Jupyter. Ένα έγγραφο Jupyter Notebook είναι στην πραγματικότητα ένα έγγραφο JSON, το οποίο περιέχει μία διατεταγμένη λίστα από

κελιά εισόδου/εξόδου τα οποία περιέχουν κώδικα, κείμενο, μαθηματικές παραστάσεις, γραφήματα και ποικίλα μέσα όπως εικόνες. Το περιβάλλον Jupyter Notebook δύναται να συνδεθεί με πλήθος πυρήνων (kernels) επιτρέποντας τον προγραμματισμό σε διάφορες γλώσσες προγραμματισμού όπως η Python, η R και η Haskell. Στα πλαίσια της παρούσας διπλωματικής εργασίας, ο πηγαίος κώδικας συμπεριλαμβανομένων των γραφικών απεικονίσεων εκτελέστηκε εξολοκλήρου στο Jupyter Notebook.

3.4 Η Βάση Δεδομένων TimescaleDB

Η TimescaleDB είναι η μόνη βάση δεδομένων χρονολογικών σειρών ανοιχτού κώδικα που υποστηρίζει εγγενώς πλήρη SQL, συνδυάζοντας τη δύναμη, την αξιοπιστία και την ευκολία χρήσης μιας σχεσιακής βάσης δεδομένων με την επεκτασιμότητα που παρατηρείται συνήθως στα συστήματα NoSQL. Βασισμένη στη PostgreSQL, η TimescaleDB χρησιμοποιείται σε πληθώρα εφαρμογών, συμπεριλαμβανομένης της ανάλυσης βιομηχανικών δεδομένων, των σύνθετων συστημάτων παρακολούθησης, της αποθήκευσης λειτουργικών δεδομένων, της διαχείρισης χρηματοοικονομικού κινδύνου, της τεχνολογίας διαφήμισης, τις τηλεπικοινωνίες και πολλά άλλα. Στα πλαίσια της διπλωματικής, βάσει της μορφής των δεδομένων επιλέχτηκε η συγκεκριμένη βάση δεδομένων με σκοπό τη πληρότητα της εφαρμογής.

Κεφάλαιο 4
Τα Δεδομένα

4.1 Εισαγωγή

Η εταιρεία ‘Αττικές Διαδρομές Α.Ε.’ ιδρύθηκε το 1999 κι έχει αναλάβει τη λειτουργία και συντήρηση της Αττικής Οδού. Κύριος στόχος της εταιρείας είναι η συνεχής, αδιάλειπτη και ομαλή λειτουργία του αυτοκινητόδρομου, μεριμνώντας παράλληλα για τη παροχή υπηρεσιών υψηλού επιπέδου στους χρήστες. Στις καθημερινές δραστηριότητες της ‘Αττικές Διαδρομές Α.Ε.’ περιλαμβάνονται:

- Η διαχείριση της κυκλοφορίας.
- Ο εντοπισμός, η επέμβαση και η αντιμετώπιση συμβάντων και ατυχημάτων.
- Η συντήρηση του έργου (προληπτική και διορθωτική).
- Η συλλογή διοδίων.
- Ο σχεδιασμός της πολιτικής διοδίων και η διαχείριση των διαφόρων συνδρομητικών - εμπορικών πακέτων πληρωμής διοδίων.
- Η παροχή υπηρεσιών προστιθέμενης αξίας σε όλους τους εμπλεκομένους μέσω της λειτουργίας της Αττικής Οδού (π.χ η ενημέρωση και εκπαίδευση σε θέματα οδικής ασφάλειας).

Στην Αττική Οδό, όλοι οι σταθμοί διοδίων έχουν τοποθετηθεί στις εισόδους του αυτοκινητόδρομου, σε ειδικά διαμορφωμένους χώρους. Έτσι, οι οδηγοί πληρώνουν διόδια μόνο μία φορά, κατά την είσοδό τους στον αυτοκινητόδρομο. Το σύστημα διοδίων που εφαρμόζεται είναι ανοιχτού τύπου, με ενιαία χρέωση για όλες τις διαδρομές. Συνολικά, υπάρχουν 39 σταθμοί διοδίων με 195 λωρίδες (πύλες). Οι διαθέσιμοι τρόποι πληρωμής διέλευσης είναι οι ακόλουθοι:

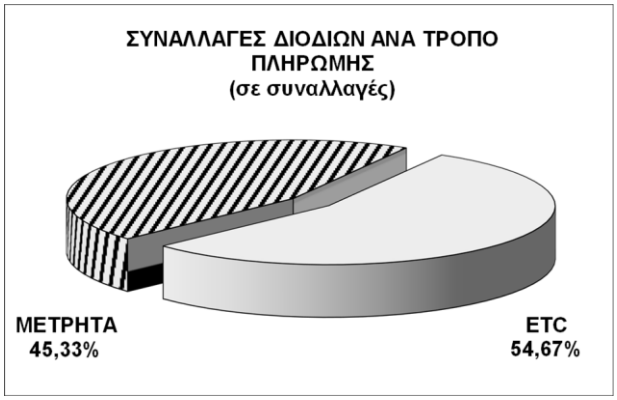
- Με μετρητά σε λωρίδα με εισπράκτορα διοδίων.
- Με τη χρήση οποιασδήποτε τραπεζικής κάρτας σε λωρίδα με εισπράκτορα διοδίων.
- Με τη χρήση του ηλεκτρονικού πομποδέκτη e-Pass.

Η κατανομή των πυλών, αλλά και η χωρητικότητα ανά τρόπο πληρωμής διέλευσης εικονίζονται παρακάτω:

ΛΩΡΙΔΕΣ	ΧΩΡΗΤΙΚΟΤΗΤΑ	ΛΩΡΙΔΕΣ (ΠΥΛΕΣ)
Λειτουργία με εισπράκτορα	300 οχήματα / ώρα	99 έως 156
Λειτουργία με e-Pass	840 οχήματα / ώρα	39 έως 96

Η Αττική Οδός εισήγαγε πρώτη στην Ελλάδα τη χρήση της ηλεκτρονικής συσκευής e-Pass, μια συσκευή που δίνει τη δυνατότητα στους οδηγούς να τη χρησιμοποιούν για τις διελεύσεις τους στις ηλεκτρονικές λωρίδες διοδίων, αποφεύγοντας τη διαδικασία πληρωμής με μετρητά. Η απόκτηση της συσκευής συνοδεύει κάθε εγγραφή στα συνδρομητικά προγράμματα τα οποία είναι τα: EXPRESS, BONUS, FRIENDLY, MOTO, BUSINESS, EXPRESS TRUCK, FRIENDLY TRUCK. Τα προγράμματα αυτά καλύπτουν τις ανάγκες όλων των οδηγών, αφού διαφοροποιούνται ανάλογα με το είδος του οχήματος, τη συχνότητα χρήσης του αυτοκινητόδρομου και τη μέθοδο πληρωμής (προπληρωμή – μεταπληρωμή).

Από την έναρξη του e-Pass τον Ιούνιο του 2002, ο αριθμός των συνδρομητών έφτασε τους 100.000 ενεργούς πομποδέκτες το Νοέμβριο του 2005, τους 200.000 το Φεβρουάριο του 2007, τους 300.000 το Δεκέμβριο του 2008, τους 400.000 τον Ιούνιο του 2010, τους 500.000 το Μάιο του 2014 και τους 600.000 το Φεβρουάριο του 2018. Η διαρκής αύξηση των συνδρομητών της Αττικής Οδού είχε ως αποτέλεσμα και την αύξηση των ηλεκτρονικών συναλλαγών. Όπως αποτυπώνεται και στο παρακάτω διάγραμμα, με την εξέλιξη των συνδρομητών της Αττικής Οδού, καθ' όλη τη διάρκεια του 2021, οι ηλεκτρονικές συναλλαγές κινήθηκαν, κατά μέσο όρο, στο 55% περίπου, ενώ η διείσδυση του e-Pass πλησίασε το 64,34% τον Απρίλιο του 2020, κατά τη διάρκεια δηλαδή του πρώτου lockdown, στο πλαίσιο αντιμετώπισης της πανδημίας COVID-19. Στις ώρες αιχμής, σε ορισμένες περιπτώσεις, οι ηλεκτρονικές συναλλαγές έφτασαν μέχρι και το 72% των συνολικών συναλλαγών.



4.2 Παρουσίαση των Δεδομένων

Όπως αναφέρθηκε στην Εισαγωγή της διπλωματικής εργασίας, τα δεδομένα που διατέθηκαν, έχουν να κάνουν με στοιχεία συνδρομητών ηλεκτρονικού εισιτηρίου e-Pass των 'Αττικών Διαδρομών Α.Ε.'. Η αρχική μορφή των δεδομένων, όπως διατέθηκαν, ήταν αρχεία τύπου CSV (Comma-separated value), δηλαδή τιμών διαχωρισμένων με κόμματα. Τα δεδομένα αυτά, χωρίστηκαν σε 2 κατηγορίες βάσει του περιεχομένου τους:

- Αρχείο αποτελούμενο από γενικά στοιχεία συνδρομητών (account_data), τα οποία βρίσκονταν σε ένα ενιαίο αρχείο CSV
- Σύνολο αρχείων αποτελούμενο από στοιχεία εσόδων διελεύσεων από συνδρομητές (all_revenues), τα οποία αποτελούνταν από πολλαπλά αρχεία CSV, χωρισμένα βάσει ημερομηνίας.

4.2.1 Γενικά στοιχεία συνδρομητών

Σε ότι αφορά το συγκεκριμένο αρχείο, όπως δηλώνει και ο τίτλος του, αποτελούταν από γενικά στοιχεία που αφορούν τους συνδρομητές ηλεκτρονικού εισιτηρίου. Κατά το άνοιγμά του μέσω του προγράμματος Microsoft Excel, παρατηρήθηκαν συνολικά 580428 γραμμές, με τα εξής πεδία ανά στήλη:

- ACCOUNT_ID: Η στήλη αυτή περιείχε ένα μοναδικό οκταψήφιο αριθμό που αντιστοιχούσε στον αριθμό 'ταυτότητας' του κάθε συνδρομητή.
- ACCOUNT_STATUS: Η στήλη αυτή έδινε πληροφορία για το εάν το εκάστοτε account διέθετε πομποδέκτη τον οποίο είχε καταχωρημένο σε περισσότερα από ένα οχήματα.
- ACCOUNT_STATUS_LDATETIME: Η στήλη αυτή αναφερόταν στη ημερομηνία τελευταίας τροποποίησης δήλωσης για το εάν υπήρχε καταχωρημένος πομποδέκτης σε περισσότερα οχήματα.
- CREATE_LDATETIME: Η στήλη αυτή περιλάμβανε την ημερομηνία σε ώρα, μέρα, μήνα και χρονιά που δημιουργήθηκε το εκάστοτε account.
- MAILING_POSTAL_CODE: Η στήλη αυτή έδινε το ταχυδρομικό κώδικα των accounts.

- PROVINCE_NAME: Η στήλη αυτή δήλωνε τον νομό στον οποίο είναι καταχωρημένος ο κάτοχος του account.

- COUNTRY_NAME: Η στήλη αυτή περιείχε το όνομα της χώρας στην οποία είναι καταχωρημένο το εκάστοτε account.

- PROFESSION: Η στήλη αναφερόταν στο επάγγελμα/ιδιότητα για κάθε account.

- ACCOUNT_PROFILE_NAME: Η στήλη αυτή ήταν η μοναδική που περιλάμβανε 2 διαφορετικά πεδία. Συγκεκριμένα ανέφερε το συνδρομητικό πρόγραμμα, αλλά παράλληλα έδινε πληροφορίες για την ιδιότητα των πελατών.

- BILLING_TYPE: Η στήλη αυτή αναφερόταν στο εάν ο λογαριασμός είναι προπληρωμής (prepaid) ή μεταπληρωμής (billed).

- PREAUTH_STATUS: Η συγκεκριμένη στήλη αναφερόταν στο εάν το εκάστοτε account έχει ενεργοποιημένη πάγια εντολή, δηλαδή εάν ανανεώνεται αυτόματα το υπόλοιπο του λογαριασμού του με χρέωση τραπεζικής κάρτας.

- E_INVOICE_STATUS: Η συγκεκριμένη στήλη αναφερόταν στο εάν το εκάστοτε account είχε επιλέξει να του στέλνονται οι ειδοποιήσεις για το υπόλοιπο του λογαριασμού του μέσω ηλεκτρονικού ταχυδρομείου.

- WEB_USER: Η στήλη αυτή έδειχνε εάν ο εκάστοτε χρήστης είχε κάνει εγγραφή στην ηλεκτρονική υπηρεσία ‘my e-Pass’ στην ιστοσελίδα της Αττικής Οδού.

- WEB_USER_ACTIVATION_DATE: Για τα accounts που είχαν κάνει εγγραφή στην ηλεκτρονική υπηρεσία ‘my e-Pass’, η στήλη αυτή δήλωνε την ημερομηνία σε ώρα, μέρα, μήνα και χρονιά της εγγραφής τους.

Σημειώνεται το γεγονός πως αν και έγινε αναφορά σε όλα τα πεδία, κατά τη διαδικασία ανάλυσης των δεδομένων επιλέχτηκαν συγκεκριμένα, εκείνα τα οποία εξυπηρετούσαν στους στόχους της διπλωματικής.

4.2.2 Έσοδα Διελεύσεων

Τα συγκεκριμένα αρχεία, έδιναν πληροφορίες για τα έσοδα από τις διελεύσεις, αλλά και τον αριθμό των διελεύσεων ανά account. Κάθε αρχείο, περιείχε τις πληροφορίες αυτές για τη διάρκεια ενός μήνα. Συνολικά, διατέθηκαν 51 αρχεία, τα οποία εκτείνονταν χρονικά από Ιανουάριο του 2017 έως και Μάρτιο του 2021. Κάθε αρχείο, παρουσίαζε διαφορετικό αριθμό γραμμών, άρα και account, μια και όπως φάνηκε μετέπειτα στην ανάλυση των δεδομένων, για κάθε ένα αρχείο περιλαμβάνονταν μόνο τα account που είχαν διέλευση σε εκείνο το μήνα. Ανοίγοντας τα αρχεία μέσω του προγράμματος Microsoft Excel εντοπίστηκαν τα εξής πεδία ανά στήλη (ισχύουν για κάθε αρχείο) :

- ACCOUNT_ID: Η στήλη αυτή περιείχε ένα μοναδικό οκταψήφιο αριθμό που αντιστοιχούσε στον αριθμό ‘ταυτότητας’ του κάθε συνδρομητή.

- STATEMENT_ISSUE_ID: Η στήλη αυτή αναφερόταν στον αύξοντα αριθμό του μήνα τιμολόγησης.

- STATEMENT_ISSUE_COUNT_ID: Όπως η παραπάνω στήλη, έτσι και αυτή αναφερόταν στον αύξοντα αριθμό του μήνα τιμολόγησης. (σε ορισμένα αρχεία είχε αυτόν το τίτλο)

- DESCRIPTION: Η στήλη αυτή ανάγραφε το μήνα στον οποίο αντιστοιχούσε το εκάστοτε αρχείο.

- AO_CNT: Η στήλη αυτή αντιστοιχούσε στον αριθμό διελεύσεων σε διόδια εντός του οδικού δικτύου της Αττικής Οδού για κάθε account στο αρχείο.

- MOREAS_CNT: Η στήλη αυτή αντιστοιχούσε στον αριθμό διελεύσεων σε διόδια εντός του αυτοκινητόδρομου Α7 ή Μορέα για κάθε account στο αρχείο.

- OLYMPIA_CNT: Η στήλη αυτή αντιστοιχούσε στον αριθμό διελεύσεων σε διόδια εντός του αυτοκινητόδρομου Α8 ή Ολυμπίας Οδού για κάθε account στο αρχείο.

- AIGAIO_CNT: Η στήλη αυτή αντιστοιχούσε στον αριθμό διελεύσεων σε διόδια εντός του αυτοκινητόδρομου Αιγαίου (κομμάτι του αυτοκινητόδρομου Α1) για κάθε account στο αρχείο.

- GEFYRA_CNT: Η στήλη αυτή αντιστοιχούσε στον αριθμό διελεύσεων στα διόδια της γέφυρας Ρίου-Αντιρρίου για κάθε account στο αρχείο.

- NEA_CNT: Η στήλη αυτή αντιστοιχούσε στον αριθμό διελεύσεων σε διόδια εντός του αυτοκινητόδρομου Α3 (Ε65) ή Κεντρικής Ελλάδας για κάθε account στο αρχείο.

- KENTRIKI_CNT: Η στήλη αυτή αντιστοιχούσε στον αριθμό διελεύσεων σε διόδια εντός της Νέας Οδού, η οποία περιλαμβάνει την Ιόνια Οδό, αλλά και τμήμα του αυτοκινητόδρομου Α.Θ.Ε για κάθε account στο αρχείο.

- EGNATIA_CNT: Η στήλη αυτή αντιστοιχούσε στον αριθμό διελεύσεων σε διόδια εντός του αυτοκινητόδρομου Α2 ή Εγνατίας Οδού για κάθε account στο αρχείο.

- AO_AMT: Η στήλη αυτή αντιστοιχούσε στο συνολικό αντίτιμο σε ευρώ των διοδίων, κατά τις διελεύσεις εντός του οδικού δικτύου της Αττικής Οδού, ανά account στο αρχείο.

- MOREAS_AMT: Η στήλη αυτή αντιστοιχούσε στο συνολικό αντίτιμο σε ευρώ των διοδίων, κατά τις διελεύσεις εντός του αυτοκινητόδρομου Α7 ή Μορέα για κάθε account στο αρχείο.

- OLYMPIA_AMT: Η στήλη αυτή αντιστοιχούσε στο συνολικό αντίτιμο σε ευρώ των διοδίων, κατά τις διελεύσεις εντός του αυτοκινητόδρομου Α8 ή Ολυμπίας Οδού για κάθε account στο αρχείο.

- AIGAIO_AMT: Η στήλη αυτή αντιστοιχούσε στο συνολικό αντίτιμο σε ευρώ των διοδίων, κατά τις διελεύσεις εντός του αυτοκινητόδρομου Αιγαίου (κομμάτι του αυτοκινητόδρομου Α1) για κάθε account στο αρχείο.

- GEFYRA_AMT: Η στήλη αυτή αντιστοιχούσε στο συνολικό αντίτιμο σε ευρώ των διοδίων, κατά τις διελεύσεις στη γέφυρα Ρίου-Αντιρρίου για κάθε account στο αρχείο.

- NEA_AMT: Η στήλη αυτή αντιστοιχούσε στο συνολικό αντίτιμο σε ευρώ των διοδίων, κατά τις διελεύσεις εντός του αυτοκινητόδρομου Α3 (Ε65) ή Κεντρικής Ελλάδας για κάθε account στο αρχείο.

- KENTRIKI_AMT: Η στήλη αυτή αντιστοιχούσε στο συνολικό αντίτιμο σε ευρώ των διοδίων, κατά τις διελεύσεις εντός της Νέας Οδού, η οποία περιλαμβάνει την Ιόνια Οδό, αλλά και τμήμα του αυτοκινητόδρομου Α.Θ.Ε, για κάθε account στο αρχείο.

- EGNATIA_AMT: Η στήλη αυτή αντιστοιχούσε στο συνολικό αντίτιμο σε ευρώ των διοδίων, κατά τις διελεύσεις εντός του αυτοκινητόδρομου Α2 ή Εγνατίας Οδού για κάθε account στο αρχείο.

4.3 Φόρτωση των Δεδομένων

Το πρώτο βήμα πριν την επεξεργασία και ανάλυση των δεδομένων, που μετέπειτα οδήγησαν στο προσδιορισμό των μεγεθών και επίτευξη των στόχων της διπλωματικής εργασίας, αποτέλεσε η φόρτωσή τους στο υπολογιστικό περιβάλλον. Όπως προαναφέρθηκε, το Jupyter Notebook αποτέλεσε την επιλογή για ένα τέτοιο περιβάλλον, συνεπώς δημιουργήθηκε ένα αντίστοιχο αρχείο για τη αποθήκευση αλλά και εκτέλεση του κώδικα. Το τελευταίο περιέλαβε όλο τον κώδικα που υλοποιήθηκε για την εφαρμογή, πέραν εκείνο της δημιουργίας βάσης δεδομένων. Ξεκινώντας λοιπόν από τη φόρτωση, εφαρμόστηκαν 2 τρόποι, ο ένας από τη βάση δεδομένων, ενώ ο άλλος τοπικά, από τη μνήμη του υπολογιστή. Καθώς η ταχύτητα φόρτωσης στη περίπτωση της βάσης δεδομένων ήταν πολύ μικρή λόγω του απομακρυσμένου διακομιστή, τα δεδομένα φορτώνονταν κατά κόρον τοπικά.

4.3.1 Φόρτωση του αρχείου Γενικών Στοιχείων Συνδρομητών

Για τη φόρτωση του συγκεκριμένου αρχείου εκτελέστηκαν τα εξής βήματα με την αντίστοιχη σειρά:

- Προσδιορισμός του σχετικού μονοπατιού (relative path) που βρίσκεται το αρχείο προς φόρτωση σε σχέση με τη τοποθεσία του αρχείου του Jupyter Notebook.

- Χρήση της συνάρτησης read_csv() της βιβλιοθήκης pandas για τη φόρτωση του αρχείου, προσδιορισμός των κατάλληλων παραμέτρων της και μετατροπή του αρχείου σε δομή δεδομένων DataFrame με όνομα account_data μεγέθους 580.427 γραμμών και 14 στηλών. Οι πρώτες 5 γραμμές και 6 από τα 14 πεδία (στήλες) εικονίζονται παρακάτω:

ACCOUNT_ID	ACCOUNT_STATUS	ACCOUNT_STATUS_LDATETIME	CREATE_LDATETIME	MAILING_POSTAL_CODE	PROVINCE_NAME	COUNTRY_NAME	PROFES
10102560	Active	23/6/2008 11:00	19/6/2002 13:34	19016	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	
10102571	Active Partial	13/9/2010 13:52	19/6/2002 14:04	19002	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	Χ ΕΜΠΟΡ SUPER
10102626	Active	13/11/2019 11:05	19/6/2002 15:55	15125	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	Ε ΕΝ
10102663	Active	10/12/2008 14:51	19/6/2002 17:06	15121	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	Ε
10100805	Active	1/2/2014 9:10	6/6/2002 10:38	19400	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	ΚΑΤΑΣΚ

Σε ότι αφορά ορισμένες παραμέτρους τις συνάρτησης read_csv(), αναφέρεται ότι καθώς είναι αρχείο CSV με υποδιαστολή κόμματος για δεκαδικά ψηφία, οι χαρακτήρες των separator και decimal ήταν «;» και «,» αντίστοιχα. Τέλος, για τη κωδικοποίηση του συγκεκριμένου αρχείου χρησιμοποιήθηκε το πρότυπο ‘iso8859_7’, το οποίο είναι σχεδιασμένο για την σύγχρονη Ελληνική Γλώσσα.

4.3.2 Φόρτωση των αρχείων Εσόδων Διελεύσεων

Για τη φόρτωση των συγκεκριμένων αρχείων, απαιτήθηκαν κάποια επιπλέον βήματα πέραν των παραπάνω. Αναλυτικότερα εκτελέστηκαν τα εξής βήματα:

- Δημιουργία λίστας με όνομα files, κάθε στοιχείο της οποίας αντιστοιχεί στο path κάθε αρχείου προς φόρτωση, με τη βοήθεια της μεθόδου listdir().

- Δημιουργία 2 κενών λιστών με ονόματα months και dfs ούτως ώστε να αποθηκεύονται οι μήνες και τα DataFrames αρχείων αντίστοιχα.

- Δημιουργία μεταβλητής τύπου Datetime και ονόματος date και ανάθεση της ημερομηνίας που αντιστοιχεί στο παλαιότερο μήνα από το σύνολο των δεδομένων (2017-01-01).

- Χρήση δομής επανάληψης η οποία διατρέχει τα στοιχεία της λίστας files. Για κάθε επανάληψη, βάσει της συνάρτησης read_csv() δημιουργείται ένα DataFrame το οποίο προστίθεται στη λίστα dfs. Παράλληλα, η τιμή της μεταβλητή date σε κάθε επανάληψη αυξάνει κατά ένα μήνα και στη συνέχεια το αποτέλεσμα προστίθεται στη λίστα months.

- Μετά την ολοκλήρωση της δομής επανάληψης, στα ήδη υπάρχοντα Dataframes προστίθεται νέα στήλη (πεδίο) ονόματι Months η οποία περιλαμβάνει την ημερομηνία που αντιστοιχεί στο κάθε DataFrame. Η διαδικασία αυτή επιτυγχάνεται με χρήση δομή επανάληψης στη συνάρτησης zip(), που ως ορίσματα έχει τα αντίστοιχα στοιχεία των λιστών dfs και months.

- Τέλος, χάρη της συνάρτησης concat() των Pandas γίνεται ένωση όλων των παραπάνω Dataframe που αντιστοιχούν σε κάθε αρχείο, δηλαδή κάθε μήνα, σε ένα κοινό, δημιουργώντας ένα DataFrame ονόματος revenue_data μεγέθους 12.396.706 γραμμών και 18 στηλών. Οι πρώτες 5 γραμμές και 11 πεδία (στήλες) από τα 18 εικονίζονται παρακάτω:

	ACCOUNT_ID	AO_CNT	MOREAS_CNT	OLYMPIA_CNT	AIGAIO_CNT	GEFYRA_CNT	NEA_CNT	KENTRIKI_CNT	EGNATIA_CNT	AO_AMT	MOREAS_AMT	OL
0	10100043	10854	4	25	4	0	0.0	0.0	0.0	26966.64	9.8	
1	10100067	9799	0	0	0	0	0.0	0.0	0.0	50603.78	0.0	
2	10100079	1080	4	10	0	0	0.0	0.0	0.0	2696.63	9.8	
3	10100100	2225	78	476	637	64	0.0	0.0	0.0	10325.66	492.2	
4	10100111	104	0	0	0	0	0.0	0.0	0.0	259.45	0.0	

Η χρήση της μεθόδου listdir() προϋπέθεσε την εισαγωγή (import) της λειτουργικής μονάδας os. Σε ότι αφορά τις παραμέτρους των συναρτήσεων read_csv(), επιβλήθηκαν οι ίδιες τιμές, όπως ακριβώς με το αρχείο Γενικών Στοιχείων Συνδρομητών εκτός ενός αρχείου, το οποίο λόγω διαφορετικού separator κατά τη συγγραφή του, χρειάστηκε όπως διαπιστώθηκε χρήση του συμβόλου «:» αντί του «;».

4.4 Επεξεργασία των Δεδομένων

Μετά τη φόρτωση των δεδομένων και τη δημιουργία δομών αποθήκευσης και απεικόνισής τους, δηλαδή τη δημιουργία στοιχείων DataFrames, ξεκίνησε η επεξεργασία τους. Κατά την επεξεργασία έλαβαν μέρος διεργασίες όπως διαγραφές πεδίων που δεν εξυπηρετούσαν στους στόχους της διπλωματικής, διαγραφή προβληματικών στοιχείων, έλεγχοι για την ορθότητα των αποτελεσμάτων, ομαδοποιήσεις πεδίων, έτσι ώστε να παρθεί ένα ολοκληρωμένο και ταξινομημένο DataFrame το οποίο θα εξυπηρετούσε στη μετέπειτα βαθύτερη ανάλυση των δεδομένων.

4.4.1 Επεξεργασία των αρχείων Εσόδων Διελεύσεων

Όπως προαναφέρθηκε, το Dataframe ονόματος revenue_data περιείχε όλα τα δεδομένα από τα αρχεία Εσόδων Διελεύσεων. Η επεξεργασία των δεδομένων αυτών ξεκίνησε ήδη από δομή επανάληψης που χρησιμοποιήθηκε κατά τη φόρτωση των αρχείων. Αναλυτικότερα, τα πεδία STATEMENT_ISSUE_ID,STATEMENT_ISSUE_COUNT_ID και DESCRIPTION αφαιρέθηκαν με χρήση της μεθόδου drop() από το DataFrame καθώς το μεν δε προσέφερε κάποια διαχειρίσιμη πληροφορία, ενώ το δεύτερο παρείχε πλεονάζουσα πληροφορία μια και οι μήνες είχαν αποθηκευτεί σε διαφορετική στήλη.

Καθώς τα δεδομένα στο σύνολό τους επικεντρώνονταν στους χρήστες και σε ιδιότητες τους που αφορούσαν ειδικά τις διελεύσεις εντός της Αττικής Οδού, κρίθηκε σκόπιμη η συγχώνευση των δεδομένων των υπόλοιπων αυτοκινητόδρομων. Συγκεκριμένα οι τιμές των πεδίων: MOREAS_CNT, OLYMPIA_CNT, AIGAIO_CNT, GEFYRA_CNT, NEA_CNT, KENTRIKI_CNT και EGNATIA_CNT που αφορούσαν τον αριθμό διελεύσεων ανά αυτοκινητόδρομο ανά account ανά μήνα (αναλόγως αν ήταν ενεργός, δηλαδή αν έκανε έστω και μια διέλευση εκείνο το μήνα), προστέθηκαν μεταξύ τους και εκχωρήθηκαν σε νέα στήλη (πεδίο) με όνομα ELSE_CNT και στη συνέχεια αφαιρέθηκαν από το DataFrame με χρήση της μεθόδου drop().

Αντίστοιχα οι τιμές των πεδίων: MOREAS_AMT, OLYMPIA_AMT, AIGAIO_AMT, GEFYRA_AMT, NEA_AMT, KENTRIKI_AMT και EGNATIA_AMT που αφορούσαν το συνολικό αντίτιμο των διοδίων εντός των αυτοκινητοδρόμων ανά account ανά μήνα (αναλόγως αν ήταν ενεργός, δηλαδή αν έκανε έστω και μια διέλευση εκείνο το μήνα), προστέθηκαν μεταξύ τους και εκχωρήθηκαν σε νέα στήλη (πεδίο) με όνομα ELSE_AMT και στη συνέχεια αφαιρέθηκαν από το DataFrame με χρήση της μεθόδου drop(). Μετά τις παραπάνω τροποποιήσεις, το DataFrame είχε την εξής μορφή (εικονίζονται οι 5 πρώτες σειρές):

	ACCOUNT_ID	AO_CNT	AO_AMT	Month	ELSE_CNT	ELSE_AMT
0	10100043	10854	26966.64	2017-01-01	33.0	70.3
1	10100067	9799	50603.78	2017-01-01	0.0	0.0
2	10100079	1080	2696.63	2017-01-01	14.0	29.3
3	10100100	2225	10325.66	2017-01-01	1255.0	8118.0
4	10100111	104	259.45	2017-01-01	0.0	0.0

Ακολούθως, διαγράφηκαν τα στοιχεία τα οποία ενώ είχαν διελεύσεις, δηλαδή μη αρνητικό CNT, είχαν μηδενικό AMT. Με άλλα λόγια εάν για μηδενικό στοιχείο του πεδίου AO_AMT (αντίστοιχα ELSE_AMT) υπήρξε μη μηδενικό στοιχείο του πεδίου AO_CNT (αντίστοιχα ELSE_CNT). Κάτι τέτοιο προφανώς θα ήταν ανεπιθύμητο αφού είτε πρόκειται για παράβαση του χρήστη, είτε για τυπογραφικό λάθος του dataset. Για να γίνει ευκολότερα αυτό σε μια διαδικασία, διατηρήθηκαν στην ουσία τα account τα οποία είτε είχαν μηδενικές διελεύσεις, είτε δεν είχαν μηδενικό λογαριασμό, είτε και τα 2 ταυτόχρονα.

Τέλος, ακολούθησε έλεγχος για εύρεση πιθανών λαθών στο DataFrame. Συγκεκριμένα ελέγχθηκε εάν υπάρχουν στοιχεία χωρίς τιμή (Nan) σε ολόκληρο το DataFrame με τη βοήθεια της συνάρτησης isnull(), όπως επίσης και εάν οι τύποι δεδομένων κάθε στήλης είναι όντως αυτοί που έπρεπε να είναι με τη βοήθεια της ιδιότητας dtypes, ούτως ώστε να βρίσκονταν πιθανά τυπογραφικά λάθη στο αρχείο. Όπως διαπιστώθηκε, δεν υπήρξαν στοιχεία χωρίς τιμή, αλλά και οι τύποι των στοιχείων ανά πεδίο πράγματι ήταν σωστοί.

4.4.2 Επεξεργασία του αρχείου Γενικών Στοιχείων Συνδρομητών

Όπως ειπώθηκε παραπάνω, το Dataframe με όνομα account_data περιείχε όλα τα δεδομένα από το αρχείο Γενικών Στοιχείων Συνδρομητών. Αρχικά, πραγματοποιήθηκε έλεγχος για τη μοναδικότητα των accounts. Δηλαδή, ελέγχθηκε εάν ο συνολικός αριθμός γραμμών του DataFrame (μέσω της επιλογής του πρώτου στοιχείου από τη συνάρτηση shape) ταυτιζόταν με των αριθμό των ξεχωριστών τιμών του πεδίου ACCOUNT_ID (μέσω της συνάρτησης nunique()), πράγμα που διαπιστώθηκε αληθές.

Στη συνέχεια, εφόσον τα accounts που εμφανίζονταν στο αρχείο ήταν μοναδικά, τοποθετήθηκαν ως δείκτης του DataFrame για μείωση των πεδίων:

ACCOUNT_ID	ACCOUNT_STATUS	ACCOUNT_STATUS_LDATETIME	CREATE_LDATETIME	MAILING_POSTAL_CODE	PROVINCE_NAME	COUNTRY_NAME	PROFES
10102560	Active	23/6/2008 11:00	19/6/2002 13:34	19016	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	
10102571	Active Partial	13/9/2010 13:52	19/6/2002 14:04	19002	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	ΕΜΠΟΡ SUPEF
10102626	Active	13/11/2019 11:05	19/6/2002 15:55	15125	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	E EN
10102663	Active	10/12/2008 14:51	19/6/2002 17:06	15121	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	E
10100805	Active	1/2/2014 9:10	6/6/2002 10:38	19400	ΑΤΤΙΚΗΣ	ΕΛΛΑΔΑ	ΚΑΤΑΣΚ

Στη συνέχεια, το πεδίο CREATE_LDATETIME, το οποίο περιείχε τις ημερομηνίες δημιουργίας των accounts μετατράπηκε σε αντικείμενο datetime έτσι ώστε να μπορούσε να υπάρξει ευελιξία στη μετέπειτα διαχείρισή του.

Έπειτα, ξεκίνησε η διαδικασία διαγραφής περιττών-μη διαχειρίσιμων πεδίων. Πρωτού συμβεί αυτό, ελέγχθηκε ποια πεδία είχαν μηδενικές τιμές με χρήση του αθροίσματος sum() των αποτελεσμάτων της συνάρτησης isnull() παρακάτω συνάρτησης:

```
ACCOUNT_STATUS                0
ACCOUNT_STATUS_LDATETIME      0
CREATE_LDATETIME              0
MAILING_POSTAL_CODE           0
PROVINCE_NAME                 0
COUNTRY_NAME                  0
PROFESSION                 2004
ACCOUNT_PROFILE_NAME          3
BILLING_TYPE                  0
PREAUTH_STATUS                0
E_INVOICE_STATUS              0
WEB_USER                      0
WEB_USER_ACTIVATION_DATE  384631
Unnamed: 14               580426
dtype: int64
```

Συγκεκριμένα για το πεδίο ACCOUNT_PROFILE_NAME βρέθηκαν τα 3 accounts τα οποία δε περιείχαν τιμή και απομακρύνθηκαν από το DataFrame με τη μέθοδο drop(). Στη συνέχεια, για κάθε πεδίο χρησιμοποιήθηκαν οι συναρτήσεις nunique() ώστε να μπορούσε να βρεθεί το πλήθος των διαφορετικών τιμών που

λάμβανε κάθε στήλη, όπως και η value_counts() για την εκτίμηση της κατανομής του συνόλου των τιμών στις διαφορετικές τιμές κάθε στήλης. Συγκεκριμένα λοιπόν, διαγράφτηκαν τα εξής πεδία:

- ACCOUNT_STATUS, μια και δε περιείχε αξιοποιήσιμη πληροφορία για την ανάλυση.
- ACCOUNT_STATUS_LDATETIME μια και δε χρησιμοποιήθηκε ούτε το παραπάνω πεδίο, προφανώς δεν υπάρχει λόγος συμπερίληψης του τελευταίου.
- MAILING_POSTAL_CODE δεδομένου ότι η αξιοποίηση του Τ.Κ., δηλαδή η αντιστοίχισή του σε Δήμο, πέρα από εξαιρετικά επίπονη διαδικασία θα προσέθετε υπερβολικό θόρυβο στο dataset.
- COUNTRY_NAME δεδομένου ότι τα περισσότερα accounts ανήκαν σε κατοίκους της Ελλάδας.
- PROFESSION διότι η πληθώρα μη αξιοποιήσιμων διαφορετικών τιμών δε συνείσφερε στην ανάλυση.
- WEB_USER_ACTIVATION_DATE μια και αποτελούσε στήλη που αφορά εξαιρετικά ειδική πληροφορία, συνεπώς παραλήφθηκε.
- Unnamed: 14, πρόκειται για τυπογραφικό λάθος κατά τη συγγραφή του αρχείου.
- BILLING_TYPE καθώς η πλειονότητα των accounts όπως παρατηρήθηκε χρησιμοποιούσε μέθοδο προπληρωμής για την εξόφληση του λογαριασμού.

Συνεπώς το αρχείο account_data είχε μετά την αρχική επεξεργασία την ακόλουθη μορφή:

ACCOUNT_ID	CREATE_LDATETIME	PROVINCE_NAME	ACCOUNT_PROFILE_NAME	PREAUTH_STATUS	E_INVOICE_STATUS	WEB_USER
10102560	2002-06-19 13:34:00	ΑΤΤΙΚΗΣ	EXPRESS 1.2	N	N	Y
10102571	2002-06-19 14:04:00	ΑΤΤΙΚΗΣ	EXPRESS 3.1	N	Y	Y
10102626	2002-06-19 15:55:00	ΑΤΤΙΚΗΣ	EXPRESS 3.1	N	N	N
10102663	2002-06-19 17:06:00	ΑΤΤΙΚΗΣ	EXPRESS 2.1	N	N	N
10100805	2002-06-06 10:38:00	ΑΤΤΙΚΗΣ	EXPRESS 3.1	N	N	N
...	...	...	...	...	...	...
16106728	2020-08-27 13:55:00	ΑΤΤΙΚΗΣ	EXPRESS 3.1	N	N	Y
16107215	2020-08-28 08:05:00	ΑΤΤΙΚΗΣ	EXPRESS 1.2	N	N	N
16107854	2020-08-28 13:45:00	ΑΤΤΙΚΗΣ	BONUS 3.1	N	Y	Y
21159570	2020-01-09 09:31:00	ΑΤΤΙΚΗΣ	EXPRESS 1.2	N	N	N
21159581	2020-01-09 09:33:00	ΑΤΤΙΚΗΣ	EXPRESS 1.2	N	N	N

580424 rows × 6 columns

4.5 Ανάλυση των Δεδομένων

Εφόσον ολοκληρώθηκε και η διαδικασία της επεξεργασίας, ακολούθησε η ανάλυση των δεδομένων των πλέον διαμορφωμένων Dataframes. Κατά τη διαδικασία αυτή, δημιουργήθηκαν κατάλληλες συναρτήσεις και πεδία τα οποία είχαν διττό σκοπό. Αφενός να μπορούσαν να αξιοποιηθούν για το προσδιορισμό των βασικών μεγεθών και στόχων της διπλωματικής, αλλά και αφετέρου να συνδράμουν στην καλύτερη εποπτεία των δεδομένων μέσω γραφικών αναπαραστάσεων.

4.5.1 Ανάλυση των αρχείων Εσόδων Διελεύσεων

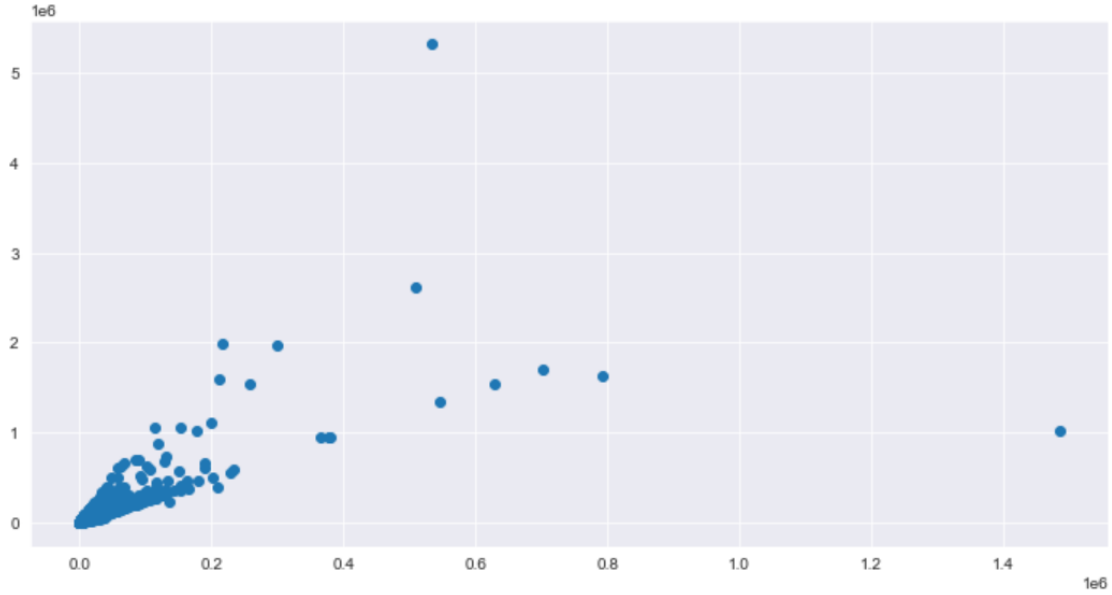
Για την ανάλυση των δεδομένων Εσόδων Διελεύσεων που βρίσκονταν μετά τη φόρτωση και επεξεργασία τους στο revenue_data Dataframe αρχικά δημιουργήθηκαν 2 νέα πεδία. Το μεν (TOTAL_AMT) αφορούσε το συνολικό αντίτιμο των διοδίων εντός όλων των αυτοκινητοδρόμων ανά account ανά μήνα (αναλόγως αν ήταν ενεργός, δηλαδή αν έκανε έστω και μια διέλευση εκείνο το μήνα) συμπεριλαμβανομένης της Αττικής Οδού, ενώ το δε (TOTAL_CNT), αφορούσε τον συνολικό αριθμό διελεύσεων από όλους τους αυτοκινητοδρόμους ανά account ανά μήνα (αναλόγως αν ήταν ενεργός, δηλαδή αν έκανε έστω και μια διέλευση εκείνο το μήνα), πάλι λαμβάνοντας υπόψη την Αττική Οδό. Τα παραπάνω αυτά πεδία δημιουργήθηκαν συνεπώς μέσω της πρόσθεσης των αντίστοιχων πεδίων (TOTAL_AMT = AO_AMT + ELSE_AMT) (TOTAL_CNT = AO_CNT + ELSE_CNT). Συγκρίνοντας τα αθροίσματα των στηλών AO_CNT με ELSE_CNT και αντίστοιχα AO_AMT με ELSE_AMT:

```
AO_AMT      4.483119e+08
ELSE_AMT    1.849083e+08
dtype: float64

AO_CNT      166284766.0
ELSE_CNT     62199996.0
dtype: float64
```

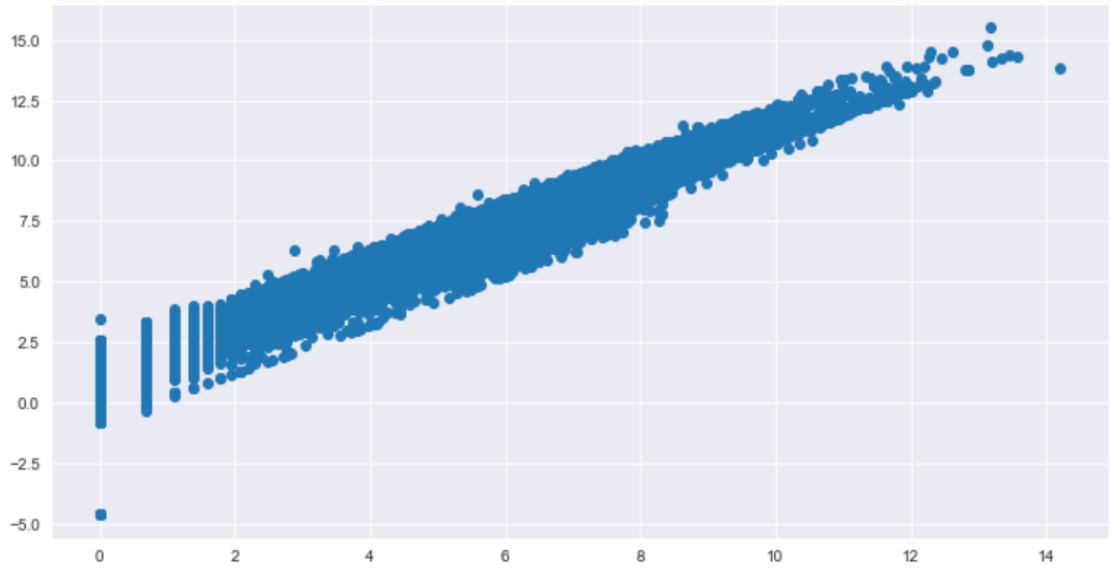
Παρατηρήθηκε ότι τα δεδομένα που αφορούσαν αποκλειστικά την Αττική Οδό παραμέναν πολυπληθέστερα του συνόλου των υπόλοιπων αυτοκινητόδρομων, πράγμα που έκανε ορθή την επιλογή συγχώνευσης, αφού δε χανόταν σημαντικό μέρος πληροφορίας.

Ακολούθησε η ανάλυση μέσω κατηγοριοποίησης του Dataframe ανά account, μέσω δηλαδή του πεδίου ACCOUNT_ID. Συγκεκριμένα δημιουργήθηκε νέο Dataframe με όνομα revenue_data_byaccounts, το οποίο περιλάμβανε 3 πεδία: τα συνολικά Έσοδα σε ευρώ των αυτοκινητόδρομων από το εκάστοτε account (TOTAL_AMT), τον αριθμό των συνολικών διελεύσεων (TOTAL_CNT) όπως και τη πιο πρόσφατη ημερομηνία κατά την οποία ο χρήστης ήταν ενεργός (δηλαδή έκανε έστω και μια διέλευση). Η δημιουργία του DataFrame επιτεύχθηκε με τη βοήθεια της συνάρτησης groupby() με επέκταση τη μέθοδο agg() η οποία ως ορίσματα δεχόταν απλές συναρτήσεις λ, οι οποίες μέσω των μεθόδων sum() για τα έσοδα και τις διελεύσεις ανά account και max() για την εύρεση πιο πρόσφατης ημερομηνίας δημιούργησαν τα 3 ζητούμενα πεδία.

Σε ότι αφορά τη συσχέτιση μεταξύ συνολικών διελεύσεων με το συνολικό αντίτιμο διοδίων ανά account, προέκυψε το εξής αποτέλεσμα:

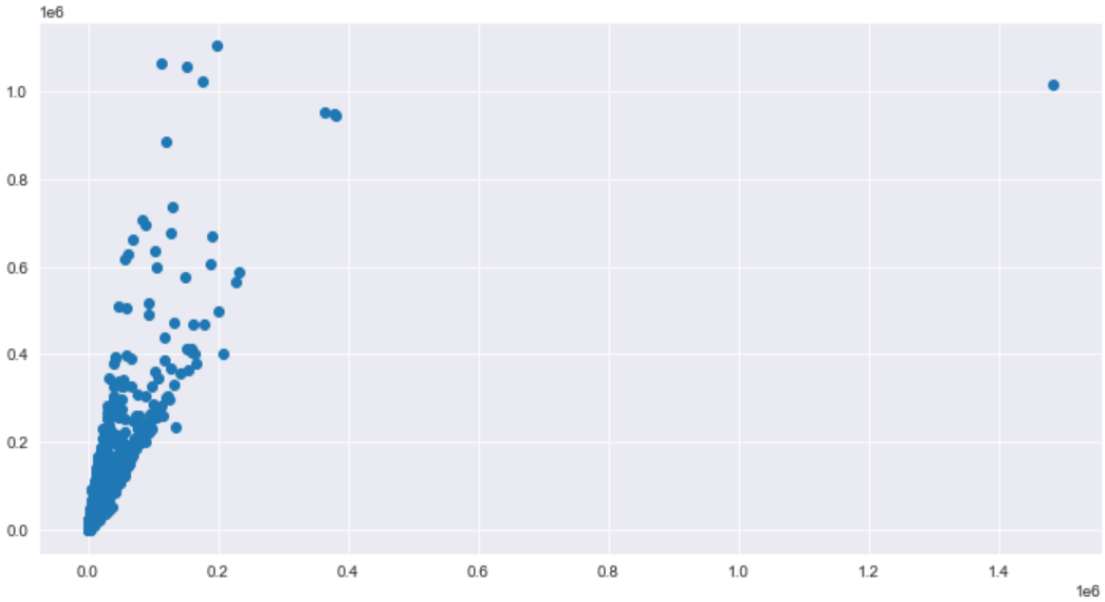


Λόγω της μορφής της παραπάνω γραφικής αναπαράστασης, δεν ήταν σαφής η ερμηνεία της, αφού πολλά στοιχεία επικαλύπτονταν. Συνεπώς η ίδια διαδικασία επαναλήφθηκε, αυτή τη φορά ωστόσο σε λογαριθμικούς άξονες:

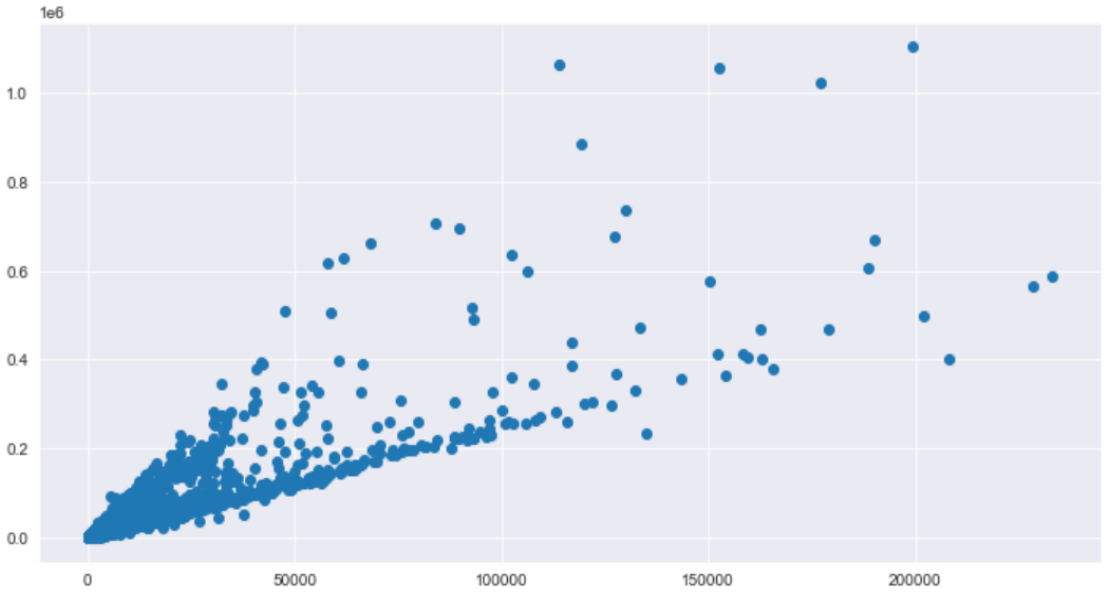


Από αυτή τη μορφή, ήταν εμφανής η γραμμική συσχέτιση μεταξύ συνολικών διελεύσεων με το συνολικό αντίτιμο διοδίων ανά account. Αυτό που παρουσιάζει ωστόσο ενδιαφέρον ήταν το τεράστιο εύρος της κατανομής των τιμών αυτών. Δηλαδή, υπήρχαν πολλά account κατά το χρονικό διάστημα των 51 μηνών που εξετάστηκε τα οποία είχαν ελάχιστες διελεύσεις, ενώ άλλα μέτρησαν χιλιάδες. Σε πλήρη αντιστοιχία λοιπόν, τα account με τις λιγοστές διαδρομές κατέβαλλαν αντίτιμο της τάξης μερικών δεκάδων ευρώ, ενώ εκείνα τα οποία είχαν χιλιάδες διελεύσεις πλήρωσαν ποσά της τάξης των χιλιάδων, ακόμα και δεκάδες χιλιάδων σε ορισμένες περιπτώσεις.

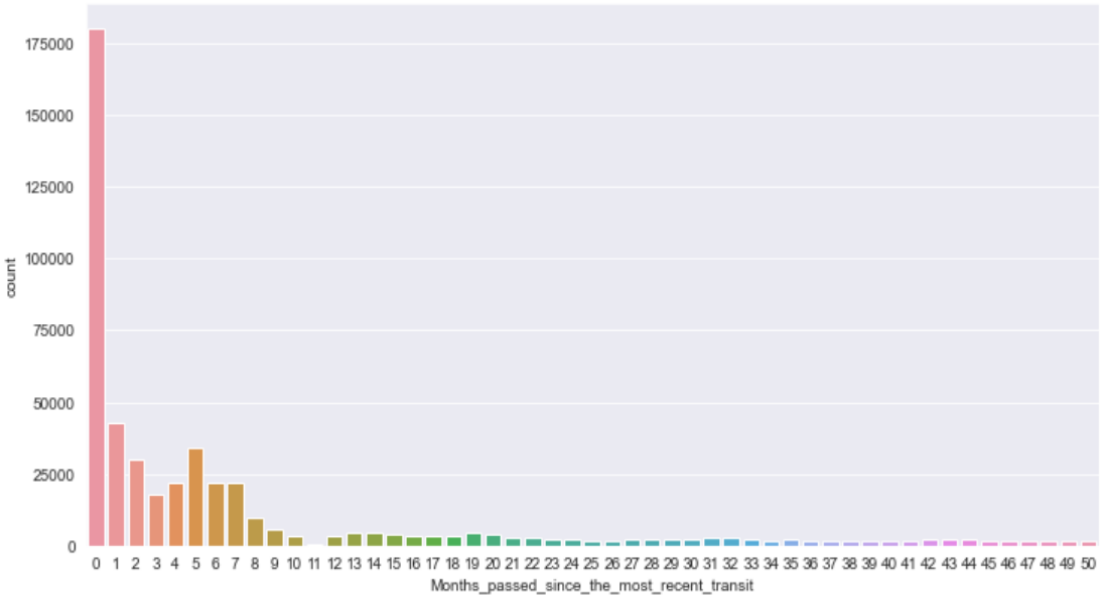
Προκειμένου οι αλγόριθμοι μηχανικής μάθησης που εφαρμόστηκαν στα επόμενα κεφάλαια να κατόρθωναν να δουλέψουν όσο το δυνατόν αποδοτικότερα, αποφασίστηκε να εντοπιστούν και να απομακρυνθούν από το DataFrame τα accounts τα οποία είχαν ακραίες τιμές (outliers). Συγκεκριμένα, προσδιορίστηκαν και απομακρύνθηκαν όλα τα accounts τα οποία είχαν συνολικό λογαριασμό μικρότερο από 1.4 ευρώ, το οποίο είναι το ελάχιστο διόδιο για μια διέλευση. Αυτό επιτεύχθηκε εύκολα με το να κρατηθούν μόνο τα accounts με συνολικό λογαριασμό μεγαλύτερο ή ίσο του 1.4. Ακόμη, με τη βοήθεια της μεθόδου sort() με επεκτάσεις head() και index εντοπίστηκαν τα 10 account με τους μεγαλύτερους λογαριασμούς, τα οποία με τη μέθοδο drop() απομακρύνθηκαν από το DataFrame. Για τη συσχέτιση μεταξύ συνολικών διελεύσεων με το συνολικό αντίτιμο διοδίων ανά account χωρίς τις ακραίες αυτές τιμές, προέκυψε το εξής γράφημα:



Βάσει του παραπάνω γραφήματος, είναι εμφανές ότι υπήρχαν ακόμη 4 ακραίες τιμές σε ότι αφορά τον αριθμό διελεύσεων, οι οποίες έπρεπε και αυτές να απομακρυνθούν. Επαναλαμβάνοντας τα ίδια βήματα στο DataFrame για τη στήλη TOTAL_CNT, η γραφική αναπαράσταση της συσχέτισης μεταξύ συνολικών διελεύσεων με το συνολικό αντίτιμο διοδίων ανά account χωρίς το σύνολο των ακραίων τιμών ήταν:

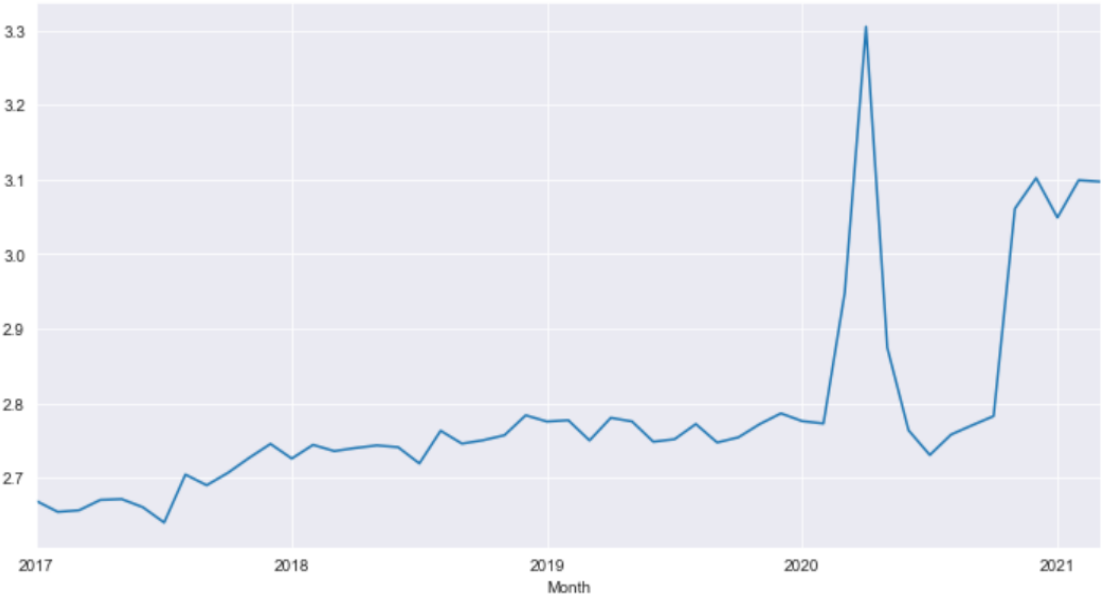


Τέλος, υπολογίστηκε και απεικονίστηκε ο αριθμός των μηνών από τη πιο πρόσφατη ημερομηνία, για τον οποίο το εκάστοτε account ήταν ανενεργό, δηλαδή δεν είχε διέλευση για όλη τη διάρκεια εκείνου του μήνα. Αυτό ήταν πολύ σημαντικό για τη μελέτη του CLV και του Churn Prediction όπως θα δειχτεί στη συνέχεια, μια και προσέδωσε μια πρώτη εκτίμηση για το πλήθος των ενεργών account ανά χρονική περίοδο. Αρχικά δημιουργήθηκε κατάλληλη συνάρτηση ονόματος months_passed η οποία δεχόταν ως όρισμα στοιχεία της στήλης Month, εξέταζε σε ποια χρονιά ανήκε ο μήνας από τη τελευταία διέλευση και αναλόγως προσέθετε τους μήνες που μεσολαβούσαν μέχρι το πιο πρόσφατο μήνα για τον οποίο υπήρχαν διαθέσιμα δεδομένα, δηλαδή το Μάρτιο του 2021. Εφόσον υλοποιήθηκε η συνάρτηση, δημιουργήθηκε νέο πεδίο στο DataFrame με τίτλο Months_passed_since_the_most_recent_transit, με τη βοήθεια της συνάρτησης apply() η οποία ως όρισμα δεχόταν τη παραπάνω συνάρτηση. Συνεπώς για το πλήθος των accounts βάσει των μηνών που ήταν ανενεργοί από τη πιο πρόσφατη ημερομηνία:

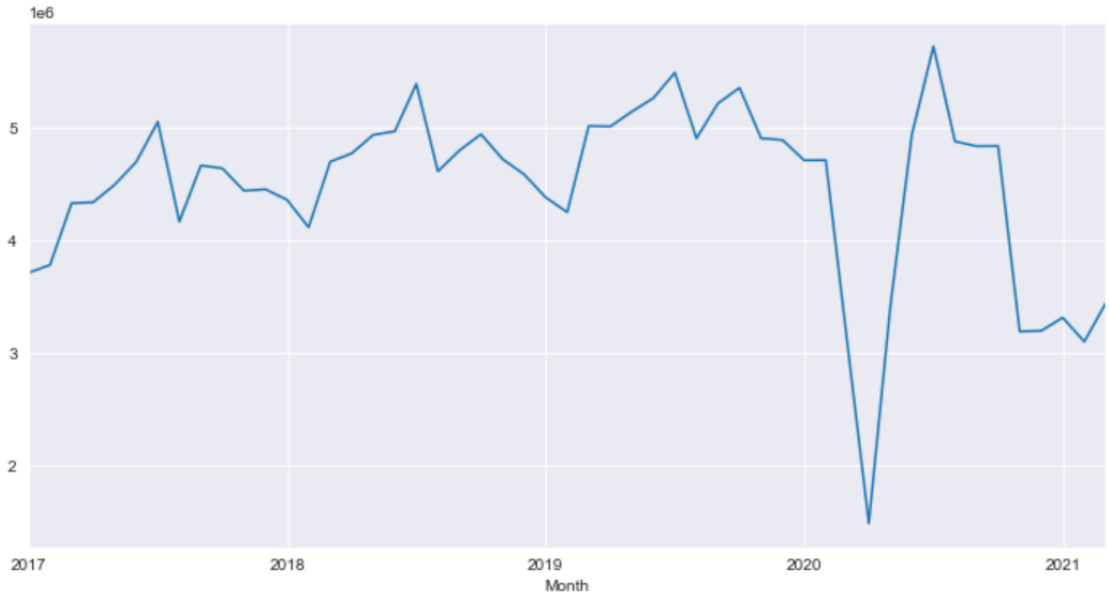


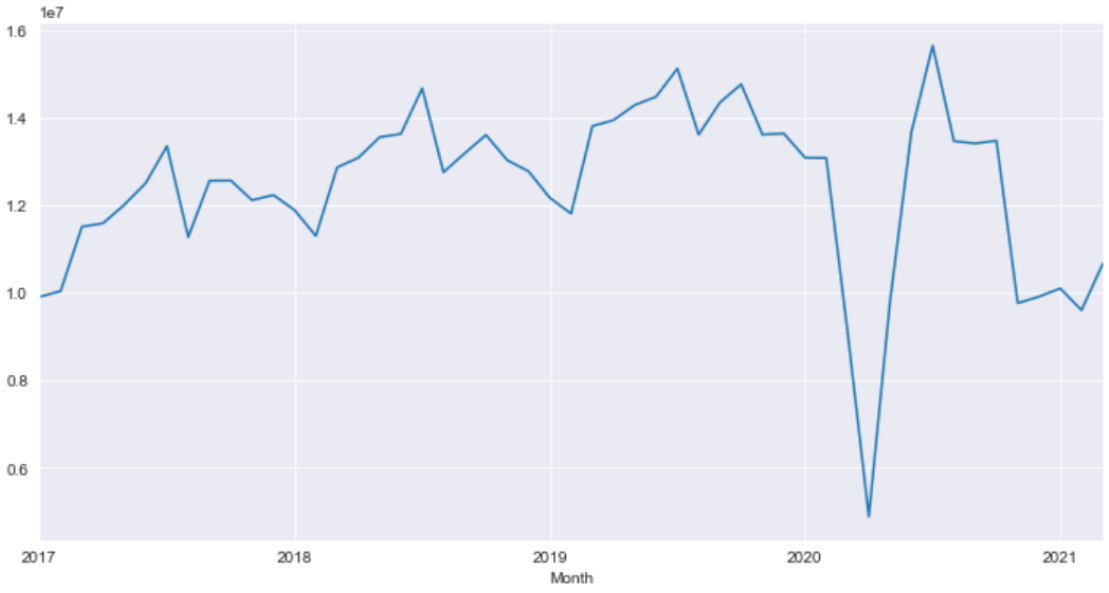
Από το παραπάνω διάγραμμα φάνηκε πως η συντριπτική πλειονότητα των accounts ήταν ενεργά μέχρι και τη πιο πρόσφατη ημερομηνία ενώ, από το προηγούμενο εξάμηνο και μετά, παρατηρήθηκε σχετικά ίδιος αριθμός account που από εκείνο το χρονικό σημείο και έπειτα, δεν έκανε άλλη διέλευση, τουλάχιστον στο διάστημα που εξετάζεται.

Εν συνεχεία, επιστρέφοντας στο αρχικό DataFrame revenue_data, πραγματοποιήθηκε κατηγοριοποίηση των στοιχείων του βάσει του μήνα, δηλαδή χρησιμοποιήθηκε η συνάρτηση groupby() με όρισμα τη στήλη Month. Η απεικόνιση του λόγου του μέσου συνολικού ποσού εσόδων δια τις συνολικές διελεύσεις ανά μήνα, εικονίζεται παρακάτω:

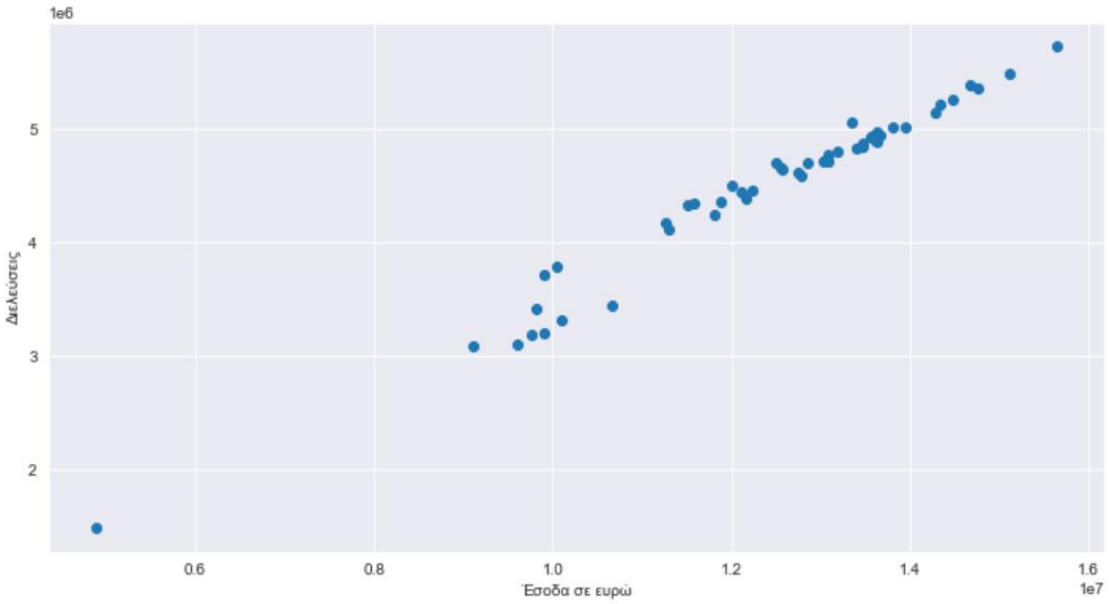


Η παραπάνω απεικόνιση ουσιαστικά έδειξε το μέσο κέρδος των αυτοκινητόδρομων σε ευρώ ανά διέλευση, άρα κατά μία έννοια τη μέση τιμή του διοδίου ανά μήνα. Όπως φάνηκε, μέχρι τους πρώτους μήνες του έτους 2020, παρουσιάστηκε σχετική σταθερότητα αφού η διακύμανση ήταν της τάξεως των 10 λεπτών του ευρώ, δηλαδή από 2.7 σε 2.8 ευρώ. Ωστόσο, τους πρώτους μήνες του 2020 και μέχρι τα μισά του έτους, παρατηρήθηκε τεράστια άνοδος της καμπύλης, η οποία σε ένα σημείο μάλιστα ξεπέρασε τα 3.3 ευρώ μέσο κέρδος ανά διέλευση. Προκειμένου να διερευνηθεί περαιτέρω η αιτία για την αύξηση αυτή, παρουσιάστηκαν γραφικά οι συνολικές διελεύσεις αλλά και τα συνολικά κέρδη χρονικά:





Εξετάζοντας την ομοιότητα των 2 παραπάνω γραφικών παραστάσεων δηλαδή απεικονίζοντας γραφικά τη συσχέτιση των συνολικών διελεύσεων ανά μήνα με τα συνολικά έσοδα από τα διόδια, επαληθεύτηκε η γραμμική σχέση τους. Περισσότερες διελεύσεις συνεπώς, αποφέρουν μεγαλύτερο κέρδος:



Ο λόγος για τον οποίο υπήρξαν πολύ χαμηλά κέρδη το πρώτο τρίμηνο του 2020 ήταν το γεγονός ότι κατά τις περιόδους αυτές, εφαρμόστηκαν μέτρα απαγόρευσης κυκλοφορίας εξαιτίας της πανδημίας του ιού SARS-CoV-2, τα οποία αναμενόμενα μείωσαν ραγδαία τον αριθμό των διελεύσεων. Ωστόσο παρά τη μείωση των διελεύσεων άρα και των κερδών, η μέση τιμή ανά διέλευση (διόδιο) όπως παρατηρήθηκε την ίδια περίοδο παρουσίασε μεγάλη αύξηση. Η πιο πιθανή εξήγηση είναι η εξής: Η στατιστική μείωση των διελεύσεων από ιδιώτες και κατόχους ΙΧ, σε συνδυασμό με τη διατήρηση του πλήθους των μετακινήσεων μεγάλων οχημάτων εταιριών (όπως φορτηγά και νταλίκες) που, λόγω της φύσης του επαγγέλματος (παραδείγματος χάρη μεταφορές πρώτων υλών) έπρεπε να συνεχιστούν ακόμα και εκείνη τη περίοδο. Σαν αποτέλεσμα, λιγότερες διελεύσεις απέφεραν μεγαλύτερα κέρδη αφού τα οχήματα αυτά κατέβαλλαν μεγαλύτερο αντίτιμο. Ωστόσο, αν και σαφώς υπήρχε επιρροή, δεν ήταν τόσο ισχυρή ώστε να οδηγήσει στη μη συμπερίληψη εκείνων των χρονικών διαστημάτων.

4.5.2 Ανάλυση του αρχείου Γενικών Στοιχείων Συνδρομητών

Ξεκινώντας την ανάλυση των δεδομένων του αρχείου Γενικών Στοιχείων Συνδρομητών, ερευνήθηκε η κατανομή των τιμών του πεδίου PROVINCE_NAME, για την αξιοποίηση πεδίου το οποίο έδινε πληροφορία για τη γεωγραφική κατανομή των χρηστών. Σημειώνεται ότι τα πεδία MAILING_POSTAL_CODE και COUNTRY_NAME έδιναν και αυτά πληροφορίες για τη τοποθεσία καταχώρησης των accounts αλλά για λόγους που προαναφέρθηκαν απορρίφθηκαν. Το πεδίο PROVINCE_NAME όπως δηλώνει και η ονομασία του περιλάμβανε το νομό στον οποίο ήταν καταχωρημένος ο χρήστης του εκάστοτε account. Μέσω της συνάρτησης value_counts() προέκυψε το εξής:

```
ΑΤΤΙΚΗΣ            509260
ΑΤΤΙΚΗΣ_            17276
ΜΕΣΣΗΝΙΑΣ           11299
ΑΡΚΑΔΙΑΣ             4763
ΑΤΤΙΚΗΣ__            4637
                    ...
SCOTLAND               1
CZECH REPUBLIC         1
NORWAY                 1
MUNCHEN                1
LITHUANIA              1
Name: PROVINCE_NAME, Length: 90, dtype: int64
```

Βάσει του αποτελέσματος, εξήχθησαν τα εξής συμπεράσματα: Πρώτα πρώτα συντριπτική πλειοψηφία των accounts ανήκε στο νομό Αττικής, ενώ για accounts καταχωρημένα εκτός Ελλάδας, ως νομός αναγραφόταν η ίδια η χώρα. Τέλος, παρατηρήθηκε ότι υπήρχαν διαφορετικές καταχωρήσεις του νομού Αττικής, με ελαφρώς διαφορετική γραφή λόγω πιθανών τυπογραφικών λαθών ή ασυνεπειών κατά τη συγγραφή του αρχείου. Για το λόγο αυτό, μέσω της συνάρτησης unique() επιθεωρήθηκαν όλες οι διαφορετικές τιμές του πεδίου, έτσι ώστε για να μπορούσε να προσδιοριστεί το ακριβές πλήθος των τιμών που ενδέχεται να αναφερόταν στον ίδιο νομό:

```
array(['ΑΤΤΙΚΗΣ', 'ΑΤΤΙΚΗΣ_', 'ΒΟΙΩΤΙΑΣ', 'ΕΥΒΟΙΑΣ', 'ΚΟΡΙΝΘΙΑΣ',
       'ΜΑΓΝΗΣΙΑΣ', 'ΘΕΣΠΡΩΤΙΑΣ', 'ΦΘΙΩΤΙΔΑΣ', 'ΔΩΔΕΚΑΝΗΣΩΝ', 'ΑΧΑΪΑΣ',
       'ΗΛΕΙΑΣ', 'ΚΥΚΛΑΔΩΝ', 'ΠΡΕΒΕΖΗΣ', 'ΡΕΘΥΜΝΟΥ', 'ΚΟΖΑΝΗΣ',
       'ΛΑΣΙΘΙΟΥ', 'ΚΑΡΔΙΤΣΑΣ', 'ΜΕΣΣΗΝΙΑΣ', 'ΘΕΣΣΑΛΟΝΙΚΗΣ',
       'ΚΕΦΑΛΛΗΝΙΑΣ', 'ΕΒΡΟΥ', 'ΚΙΛΚΙΣ', 'ΑΡΓΟΛΙΔΟΣ', 'ΧΑΝΙΩΝ',
       'ΑΙΤΩΛ/ΝΙΑΣ', 'ΑΡΚΑΔΙΑΣ', 'ΣΑΜΟΥ', 'ΛΕΥΚΑΔΟΣ', 'ΗΡΑΚΛΕΙΟΥ',
       'ΛΑΚΩΝΙΑΣ', 'ΠΙΕΡΙΑΣ', 'ΤΡΙΚΑΛΩΝ', 'ΦΛΩΡΙΝΗΣ', 'ΛΑΡΙΣΗΣ', 'ΧΙΟΥ',
       '    ΑΤΤΙΚΗΣ', 'ΦΩΚΙΔΑΣ', 'ΣΕΡΡΩΝ ', 'ΕΥΡΥΤΑΝΙΑΣ', 'ΙΩΑΝΝΙΝΩΝ',
       'ENGLAND', 'ΚΕΡΚΥΡΑΣ', 'ΑΡΤΑΣ', 'ΑΤΤΙΚΗΣ', 'ΚΑΒΑΛΑΣ', 'ΚΑΣΤΟΡΙΑΣ',
       'ΖΑΚΥΝΘΟΥ', 'ΗΜΑΘΙΑΣ', 'ΞΑΝΘΗΣ', 'ΚΥΠΡΟΣ', 'ΔΡΑΜΑΣ', 'ΛΕΣΒΟΥ',
       'ΠΕΛΛΗΣ', '     ΑΤΤΙΚΗΣ', 'ΡΟΔΟΠΗΣ', 'GERMANY ', '    ΑΤΤΙΚΗΣ',
       'NEDERLAND', 'ΑΤΤΙΚΗΣ__', 'ΓΡΕΒΕΝΩΝ', 'ΑΤΤΙΚΗΣ ', '__ΑΤΤΙΚΗΣ__',
       'SLOVAKIA', 'FRANCE', 'ΧΑΛΚΙΔΙΚΗΣ', 'BULGARIA', 'DENMARK',
       'LITHUANIA', 'ITALY', 'ROMANIA', 'SWITZERLAND', 'U.S.A.',
       'LUXEMBOURG', 'PARIS', 'BRUSSELS', 'ALBANIA', 'SPAIN', 'SOFIA',
       'AUSTRIA', 'CZECH REPUBLIC', 'UKRAINE', 'SLOVENIA', 'TURKEY',
       'POLAND', 'SERBIA', 'HUNGARY', 'SCOTLAND', 'NORWAY', 'SWEDEN',
       'MUNCHEN'], dtype=object)
```

Παρατηρώντας προσεκτικά τα παραπάνω αποτελέσματα, διακρίθηκαν 9 διαφορετικές καταχωρήσεις που αναφέρονταν στο νομό Αττικής. Εφόσον λοιπόν ο νομός Αττικής περιλάμβανε τη πλειονότητα των στοιχείων αλλά και η προσθήκη χωρών αντί νομών για accounts του εξωτερικού περιέπλεκε τα πράγματα, κρίθηκε καταλληλότερο ο διαχωρισμός να γίνει βάσει του αν το account ήταν καταχωρημένο εντός, ή εκτός νομού Αττικής. Για να γίνει αυτό, δημιουργήθηκε συνάρτηση ονόματος attiki η οποία έλεγχε εάν το account της στήλης PROVINCE_NAME άνηκε σε μία εκ των 9 διαφορετικών καταχωρήσεων που αναφέρονται στο νομό Αττικής αποδίδοντας τιμή ‘ΑΤΤΙΚΗ’, διαφορετικά απέδιδε τιμή ‘ΕΚΤΟΣ_ΑΤΤΙΚΗΣ’. Στη συνέχεια, με τη συνάρτηση apply() καταχωρήθηκαν τα αποτελέσματα της συνάρτησης attiki στην ίδια τη στήλη PROVINCE_NAME.

Συνεχίζοντας την ανάλυση, σε ότι αφορά το πεδίο ACCOUNT_PROFILE_NAME, όπως προαναφέρθηκε αποτέλεσε τη μοναδική στήλη που περιλάμβανε 2 διαφορετικά πεδία. Συγκεκριμένα κάθε στοιχείο του πεδίου περιείχε μία λέξη, ακολουθούμενη από ένα αριθμό. Η λέξη αναφερόταν στο συνδρομητικό πρόγραμμα που ήταν εγγεγραμμένο το account, ενώ ο αριθμός που ακολουθούσε αναφερόταν σε ιδιότητα του ίδιου του χρήστη. Ζητώντας διευκρίνηση για τη σημασιολογία του αριθμού προέκυψε η ακόλουθη αντιστοιχία:

- Αριθμός της μορφής 1.1 ή 1.2 αντιστοιχούσε σε Ιδιώτη.
- Αριθμός της μορφής 2.1 αντιστοιχούσε σε Ελεύθερο Επαγγελματία.
- Αριθμός της μορφής 3.1 αντιστοιχούσε σε Νομικό Πρόσωπο.

- Ειδική περίπτωση αποτέλεσε το πακέτο BUISNESS το οποίο από μόνο του αντιστοιχούσε και αυτό σε Ιδιώτη.

Βάσει της αντιστοιχίας αυτής, δημιουργήθηκε συνάρτηση με όνομα paketa, η οποία ανάπτυσσε την ιδιότητα των account με σκοπό τον μετέπειτα διαχωρισμό των πεδίων, ενώ παράλληλα ομαδοποίησε τα λιγοστά στοιχεία που δεν ανήκαν σε κάποιο ξεκάθαρο πακέτο, έτσι ώστε να διαγραφούν στη συνέχεια. Αναλυτικότερα, μέσω της συνάρτησης unique() βρέθηκαν όλες οι διαφορετικές τιμές που λάμβανε η στήλη ACCOUNT_PROFILE_NAME, έτσι ώστε να μπορέσουν να χρησιμοποιηθούν από τη συνάρτηση. Στη συνέχεια, για κάθε τιμή, σημειώθηκε το πακέτο, ακολουθούμενο από κενό (έτσι ώστε να υπάρχει τρόπος διαχωρισμού) και έπειτα την ιδιότητα, βάσει της παραπάνω αντιστοιχίας.

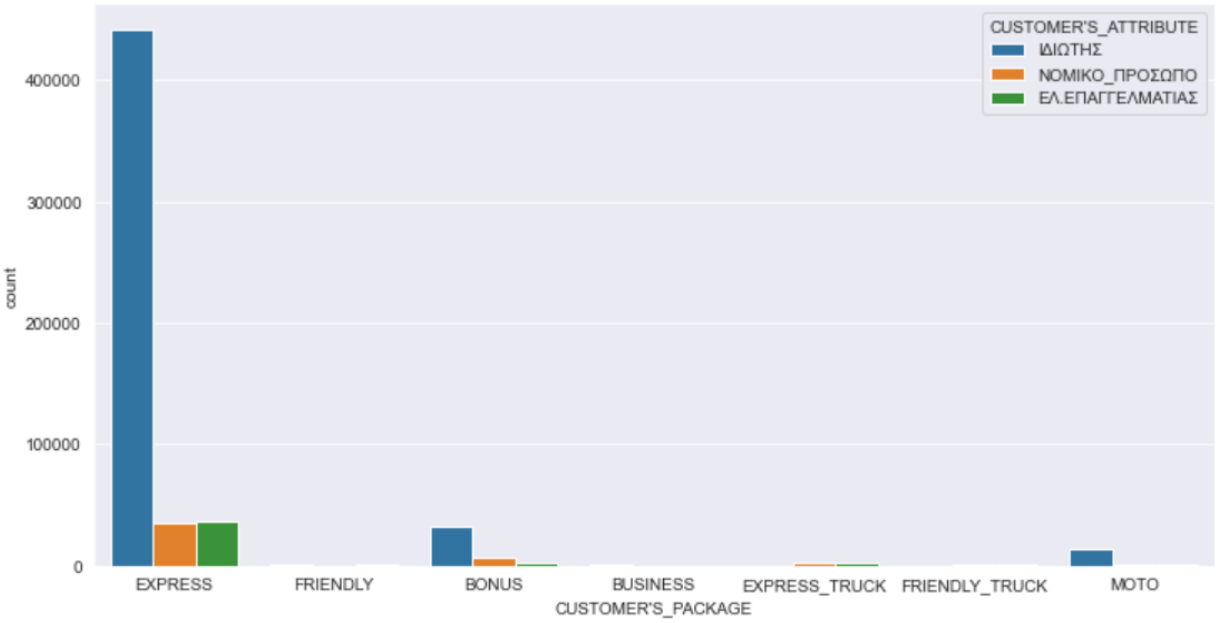
Έτσι, χρησιμοποιώντας τη μέθοδο split() με παράμετρο το ‘ ‘ (κενό) διαχωρίστηκε το πακέτο από την ιδιότητα, αποθηκεύοντας τις τιμές αυτές σε 2 νέες στήλες με ονόματα CUSTOMER'S_PACKAGE και CUSTOMER'S_ATTRIBUTE αντίστοιχα. Εφόσον η στήλη ACCOUNT_PROFILE_NAME, πλέον δε προσέφερε νέα πληροφορία, απομακρύνθηκε από το DataFrame.

Στη συνέχεια, κρίθηκε θεμιτό να γίνει ομαδοποίηση των account βάσει της παλαιότητάς τους, ώστε να εκτιμηθεί η κατανομή τους. Για να γίνει αυτό, αρχικά βρέθηκαν το παλαιότερο και το νεότερο account αντίστοιχα:

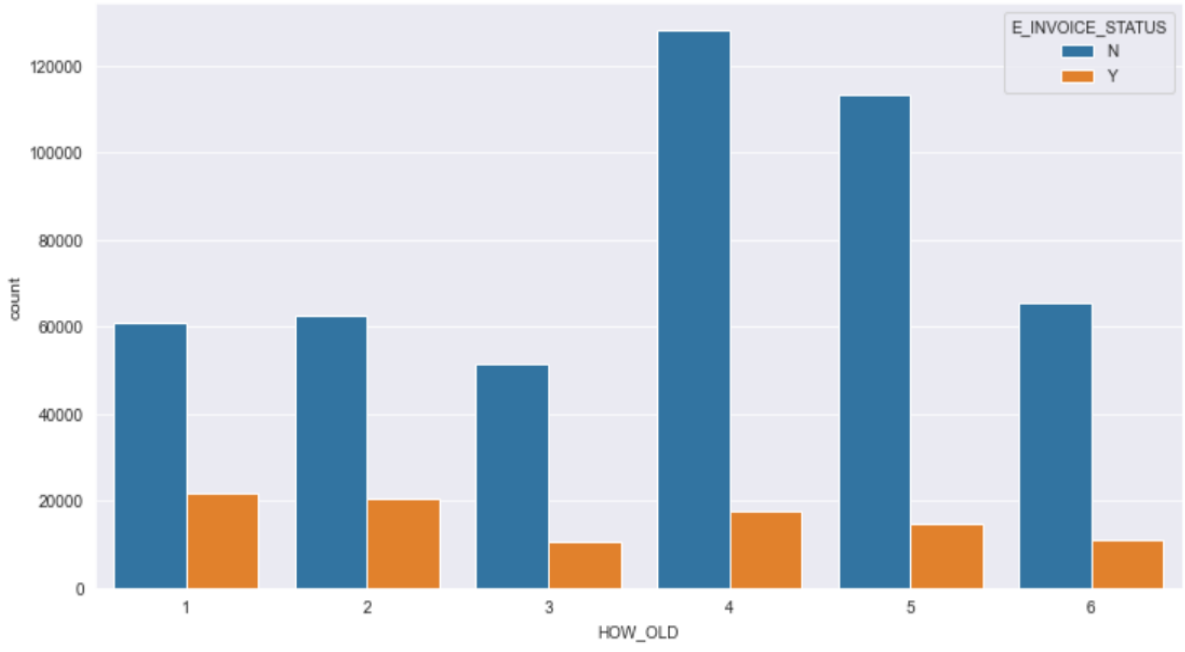
```
Timestamp('2002-01-07 13:47:00')

Timestamp('2020-12-10 17:44:00')
```

Δεδομένου ότι το πιο παλιό account δημιουργήθηκε το έτος 2002 ενώ τα πιο πρόσφατο το έτος 2020, ένα καλό χρονικό διάστημα για το διαχωρισμό των accounts εκτιμήθηκε ότι θα ήταν ανά 3 χρόνια. Έτσι δημιουργήθηκε η συνάρτηση years η οποία ανά 3 έτη (με εξαίρεση τα πρώτα 4) επέστρεφε έναν ακέραιο από το 1 έως το 6, όπου 6 αντιστοιχούσε στα παλαιότερα account φτάνοντας στα πιο πρόσφατα που είχαν τον αριθμό 1. Έτσι, με τη συνάρτηση apply() δημιουργήθηκε νέο πεδίο στο DataFrame ονόματος HOW_OLD, το οποίο για κάθε account περιλάμβανε τον αντίστοιχο ακέραιο που αποδόθηκε αναλόγως το έτος δημιουργίας του account.

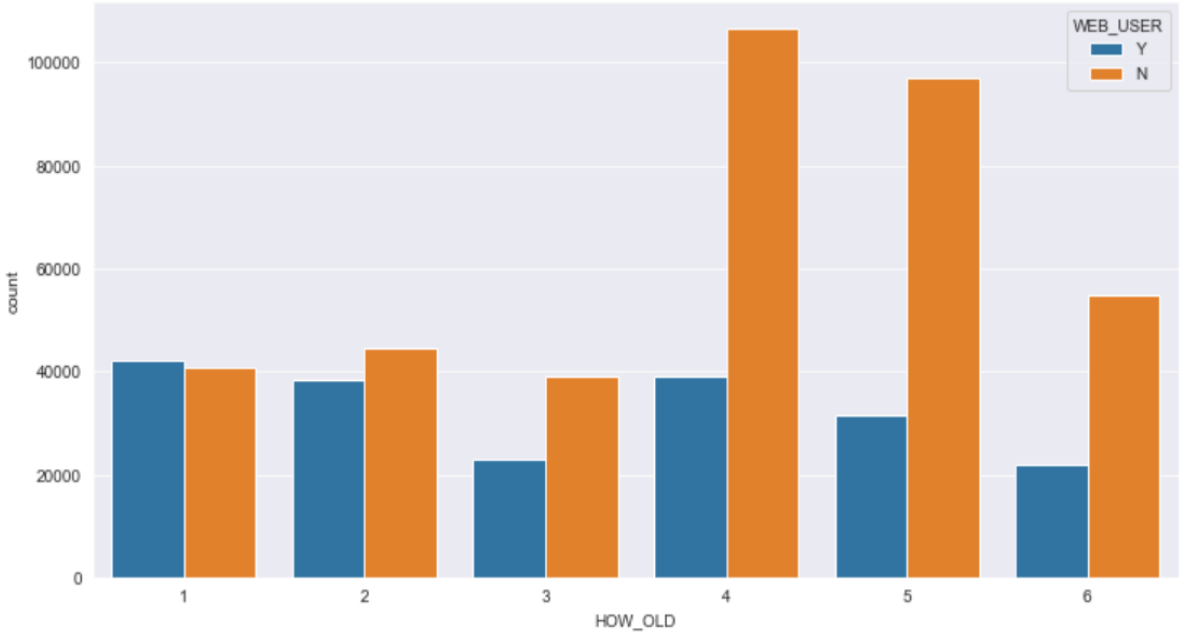
Βάσει όλων των παραπάνω τροποποιήσεων, οπτικοποιήθηκαν τα εξής πεδία:



Από τη παραπάνω γραφική η οποία αναπαριστούσε το πλήθος των account ανά πακέτο, βάσει της ιδιότητας τους, ανέδειξε το EXPRESS ως το δημοφιλέστερο πακέτο, ενώ μάλιστα οι περισσότεροι χρήστες ηλεκτρονικού εισιτηρίου καταλογίζονταν ως Ιδιώτες.



Η παραπάνω γραφική αναφερόταν στο πλήθος των account σύμφωνα με τη παλαιότητάς τους, με διάκριση ως προς το εάν είχε επιλεγεί να τους στέλνονται οι ειδοποιήσεις για το υπόλοιπο του λογαριασμού του μέσω ηλεκτρονικού ταχυδρομείου. Όπως φάνηκε, η πλειονότητα δεν είχε ενεργοποιημένες τις ειδοποιήσεις ενώ όσο πιο παλιός είναι ο χρήστης, τόσο μικρότερος είναι αναλογικά ο αριθμός των account που τις είχαν ενεργοποιημένες.



Τέλος, με βάση τα παραπάνω αποτελέσματα τα οποία αφορούσαν το πλήθος των account σύμφωνα με τη παλαιότητάς τους, με διάκριση ως προς το εάν είχαν κάνει εγγραφή στην ηλεκτρονική υπηρεσία ‘my e-Pass’ στην ιστοσελίδα της Αττικής Οδού, φάνηκε ότι τα τελευταία χρόνια το πλήθος των εγγεγραμμένων χρηστών ήταν σχεδόν ίδιο με εκείνο των μη. Αντίθετα, accounts που δημιουργήθηκαν παλαιότερα έτειναν να μη χρησιμοποιούν την υπηρεσία αυτήν.

Κεφάλαιο 5
Customer’s Segmentation

5.1 Εισαγωγή

Ο όρος Customer’s Segmentation ή αλλιώς, Τμηματοποίηση Πελατών, αναφέρεται στη πρακτική διαίρεσης των πελατών μιας εταιρίας σε ομάδες, των οποίων τα στοιχεία μοιράζονται κάποια κοινά χαρακτηριστικά. Η διαδικασία της κατάτμησης πελατών στοχεύει στη καλύτερη διαχείριση και προσαρμογή της εταιρίας στις ανάγκες των πελατών ανά τμήμα (Segment), προκειμένου η ίδια να μεγιστοποιήσει την αξία καθενός πελάτη και μακροπρόθεσμα το κέρδος της.

Βάσει της ποσότητας και ποιότητας των δεδομένων που διατίθενται για τους πελάτες (συμπεριλαμβάνοντας και τους δυνητικούς πελάτες), μια ανάλυση κατάτμησης πελατών έχει τη δυνατότητα να χρησιμοποιήσει πολλαπλούς δείκτες για το προσδιορισμό των διακριτών ομάδων, όπως δημογραφικών, συμπεριφορικών, γεωγραφικών, ψυχογραφικών και πολλών άλλων. Για τη μεγιστοποίηση της ακρίβειας, κατά τη τμηματοποίηση πελατών περιλαμβάνονται η παρακολούθηση δυναμικών αλλαγών, όπως και η συχνή ενημέρωση των νέων δεδομένων.

Υπάρχουν πολλοί τύποι μοντέλων τμηματοποίησης πελατών. Μερικοί από τους πιο συνηθισμένους τύπους είναι η τμηματοποίηση μέσω συσταδοποίησης (clustering), μέσω του RFM μοντέλου, αλλά και μέσω της παλαιότητας των πελατών (longevity). Ορισμένες εταιρίες μάλιστα μπορεί να συνδυάσουν ένα ή περισσότερα μοντέλα κατάτμησης για να επιτύχουν τους στόχους τους. Ανεξαρτήτως όμως από το μοντέλο που θα επιλεγεί, όλα απαιτούν τη δημιουργία ομάδων πελατών οι οποίες χρησιμεύουν ως το πρώτο βήμα για τη τμηματοποίησης της πελατειακής βάσης. Συνήθως αυτό έχει ως αποτέλεσμα η εταιρία να έχει μια σειρά βαθμίδων για κάθε τύπο μοντέλου τμηματοποίησης, έτσι ώστε μέσω της ανάμειξης διαφορετικών επιπέδων μεταξύ των μοντέλων να δημιουργούνται πιο εξειδικευμένα τμήματα.

5.2 Κατάτμηση Πελατών μέσω του μοντέλου RFM

5.2.1 Το μοντέλο RFM

Η ανάλυση κατά RFM αποτελεί μια τεχνική προερχόμενη από το κλάδο του μάρκετινγκ που χρησιμοποιείται για την ποσοτική κατάταξη και ομαδοποίηση πελατών με βάση του πόσο πρόσφατα συναλλάσσεται με την εταιρία, με ποια συχνότητα, αλλά και το συνολικό χρηματικό πόσο των ίδιων των συναλλαγών για τον εντοπισμό των καλύτερων πελατών και την εκτέλεση στοχευμένων καμπανιών μάρκετινγκ. Το σύστημα εκχωρεί σε κάθε πελάτη αριθμητικές βαθμολογίες βάσει των παραπάνω χαρακτηριστικών έτσι ώστε να παράξει μια αντικειμενική ανάλυση. Αναλυτικότερα, οι παράγοντες που χρησιμοποιούνται κατά την RFM ανάλυση αποτελούνται από:

- Recency: Ο όρος Recency αναφέρεται στο πόσο πρόσφατη ήταν χρονικά η τελευταία συναλλαγή μεταξύ εταιρίας και πελάτη. Για παράδειγμα οι πελάτες που πραγματοποίησαν πρόσφατα μια αγορά θα έχουν το προϊόν στο μυαλό τους και είναι πιο πιθανό να αγοράσουν ή να χρησιμοποιήσουν ξανά το προϊόν. Οι επιχειρήσεις συχνά μετρούν το Recency σε ημέρες. Αλλά, ανάλογα με το προϊόν, μπορεί να το μετρήσουν σε χρόνια, εβδομάδες ή ακόμα και ώρες.

- Frequency: Ο όρος Frequency αναφέρεται στο πόσο συχνά ο πελάτης πραγματοποίησε συναλλαγές με την εταιρία σε μια δεδομένη χρονική περίοδο. Οι πελάτες που χρησιμοποιούν προϊόντα ή υπηρεσίες σε τακτική βάση είναι πιο πιθανό να συνεχίσουν να τα χρησιμοποιούν. Επιπλέον, πελάτες οι οποίοι αγόρασαν πρώτη φορά αποτελούν στόχους για την εταιρία η οποία μέσω σωστής διαφήμισης μπορεί να τους μετατρέψει σε τακτικούς πελάτες.

- Monetary: Ο όρος Monetary αναφέρεται στο συνολικό χρηματικό ποσό που ξόδεψε ο πελάτης σε μια δεδομένη χρονική περίοδο. Αναμενόμενα λοιπόν, πελάτες που ξοδεύουν πολλά χρήματα είναι πιο πιθανό να ξοδέψουν χρήματα στο μέλλον συνεπώς έχουν μεγάλη αξία για μια επιχείρηση.

5.2.2 Κατασκευή του μοντέλου

Για τη κατασκευή του μοντέλου, αρχικά προσδιορίστηκαν οι στήλες Recency, Frequency και Monetary και εκχωρήθηκαν στο νέο DataFrame με όνομα RFMScores. Αυτό έγινε εφικτό μέσω της συνάρτησης groupby() στη στήλη ACCOUNT_ID του DataFrame account_data, με προέκταση τη συνάρτηση agg(), η οποία επέτρεψε ξεχωριστή διαχείριση σε κάθε στήλη. Συγκεκριμένα:

- Για το Recency, υπολογίστηκαν οι ημέρες που μεσολάβησαν από την ημερομηνία πιο πρόσφατης διέλευσης ανά account, δηλαδή το μέγιστο (max()) των στοιχείων της στήλης 'Month', μέχρι την ημερομηνία παρατήρησης των δεδομένων, δηλαδή το Μάρτιο του 2021.
- Για το Frequency, υπολογίστηκε το σύνολο (sum()) των στοιχείων της στήλης TOTAL_CNT, δηλαδή ο συνολικός αριθμός διελεύσεων ανά account.

- Για το Monetary, υπολογίστηκε και εκεί το σύνολο (sum()) των στοιχείων της στήλης TOTAL_AMT, δηλαδή το συνολικό ποσό που πλήρωσε σε διόδια το κάθε account.

Πρωτού γίνει ο διαχωρισμός σε τμήματα, τα στοιχεία της στήλης 'Month' μετατράπηκαν σε integers με τη βοήθεια της μεθόδου astype() προκειμένου να υπήρχε ευελιξία στη μετέπειτα τους διαχείριση, ενώ τα πεδία 'Month', 'TOTAL_CNT' και 'TOTAL_AMT' μετονομάστηκαν σε 'Recency', 'Frequency' και 'Monetary' αντίστοιχα. Ακολούθως, απομακρύνθηκαν μέσω της μεθόδου drop() από το DataFrame τα γνωστά από την ανάλυση δεδομένων accounts που περιείχαν ακραίες τιμές, έτσι ώστε να μπορούσε να επιτευχθεί ομαλότερη και πιο αντιπροσωπευτική κατάτμηση των δεδομένων.

Στη συνέχεια, τα στοιχεία κάθε στήλης του Dataframe RFMScores χωρίστηκαν σε 5 ίσα τμήματα, με τη βοήθεια της συνάρτησης quantile(). Στην ουσία η τιμή καθενός quantile αντιπροσώπευε το όριο, κάτω του οποίου οι τιμές των στοιχείων των στηλών του DataFrame ανήκαν στο συγκεκριμένο quantile. Στη συνέχεια, προκειμένου να αξιοποιηθεί κατάλληλα, το αποτέλεσμα μετατράπηκε σε τύπο δεδομένων dictionary, με keys τα Recency, Frequency και Monetary και values τα προηγούμενα αποτελέσματα, έτσι ώστε σε κάθε στήλη να αντιστοιχιζόταν ο διαχωρισμός της.

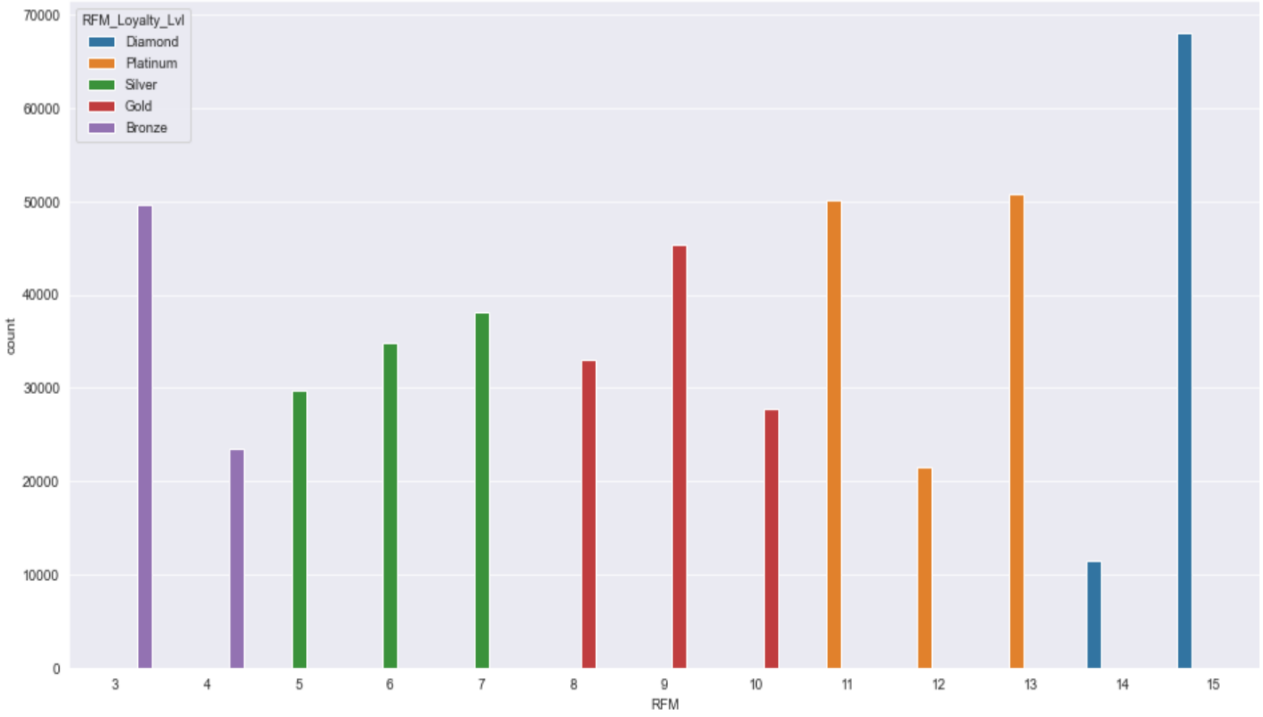
Έτσι, δημιουργήθηκαν κατάλληλες συναρτήσεις για την απόδοση «πόντων επιβράβευσης» , δηλαδή ακεραίων αριθμών εύρους από 1 έως 5, αναλόγως το τμήμα που κατατάχθηκε το στοιχείο κάθε στήλης. Για τις στήλες Frequency και Monetary, μια και όσο μεγαλύτερες οι τιμές των στοιχείων, τόσο καλύτερος πελάτης, η συνάρτηση ήταν πρακτικά η ίδια με όνομα FMScoring. Έτσι, για τα account που ανήκαν στο quantile άνω του 80%, δηλαδή για τα account με περισσότερες διελεύσεις και λογαριασμό αποδόθηκε η τιμή 5, και αντίστοιχα οι υπόλοιπες τιμές στα υπόλοιπα quantiles. Για τη στήλη Recency ωστόσο, συνέβαινε το ανάποδο αφού ήταν προτιμότερο η πιο πρόσφατη διέλευση να μην απέχει πολύ χρονικά από την ημερομηνία παρατήρησης. Έτσι δημιουργήθηκε ξεχωριστή συνάρτηση με όνομα RScoring όπου για τα account κάτω του 20%, δηλαδή εκείνων με τη μικρότερη χρονική διαφορά ημερών μεταξύ πιο πρόσφατης διέλευσης και ημερομηνίας παρατήρησης, αποδόθηκε ο αριθμός 5 και αντιστοίχως οι υπόλοιπες τιμές στα υπόλοιπα quantiles. Ως ορίσματα, οι συναρτήσεις δέχονταν το στοιχείο κάθε στήλης (παράμετρος x), το όνομα της στήλης (παράμετρος p) και το dictionary (παράμετρος d), έτσι ώστε να γίνει η αντιστοίχιση βάσει των quantiles. Συνεπώς, δημιουργήθηκαν 3 νέες στήλες με ονόματα R,F και M τα οποία για κάθε account είχαν τον αντίστοιχο ακέραιο από 1 έως 5 αναλόγως τις συναρτήσεις FMScoring και RScoring. Ακολούθως, για κάθε account υπολογίστηκε και αντιστοιχήθηκε το άθροισμα των πόντων από κάθε στήλη εκ των R,F και M και εκχωρήθηκε σε μια νέα με τίτλο RFM η οποία στην ουσία διέθετε το συνολικό σκορ που συγκέντρωσε το εκάστοτε account.

Βάσει των όσων προαναφέρθηκαν οι καλύτεροι πελάτες ήταν εκείνοι που συγκέντρωσαν αθροιστικά τους περισσότερους πόντους. Εφόσον το μεγαλύτερο άθροισμα που μπορούσε να προκύψει ήταν το 15 και το μικρότερο το 3, ορίστηκε συνάρτηση που απέδιδε 5 τίτλους «Αφοσοίωσης» (Loyalty Level) αναλόγως τη θέση στην αριθμητική κλίμακα ανά account. Οι τίτλοι αυτοί κατά αντιστοιχία από το καλύτερο προς το χειρότερο πελάτη ήταν οι «Diamond, Platinum, Gold, Silver και Bronze». Έτσι, δημιουργήθηκε συνάρτηση ονόματος Loyalty η οποία αντιστοίχιζε τους παραπάνω τίτλους ανάλογα με το σκορ που συγκέντρωσε το κάθε account. Μια και οι αριθμοί δε μπορούσαν να χωριστούν συμμετρικά σε 5 επίπεδα, αποφασίστηκε τα άκρα, δηλαδή τα accounts με αθροίσματα 3 ή 4 και 14 ή 15, να ομαδοποιηθούν ανά δυάδες και όχι τριάδες για λόγους συμμετρίας. Μια συλλογιστική για την απόφαση αυτή ήταν ότι θα ήταν σχετικά ευκολότερο για ένα πελάτη του τελευταίου τμήματος να ανέβει επίπεδο, αλλά αντίστοιχα ενός καλού σχετικά πιο δύσκολο να καταφέρει να φτάσει στο υψηλότερο επίπεδο. Εφαρμόζοντας τη συνάρτηση στη στήλη RFM και δημιουργώντας νέα στήλη με όνομα RFM_Loyalty_Lvl, επιτεύχθηκε η κατάταξη των accounts κατά το μοντέλο RFM σε 5 τμήματα. Συνολικά το DataFrame RFMScores μετά από τις τροποποιήσεις και τις εισαγωγές και εξαγωγές στηλών πήρε την εξής μορφή:

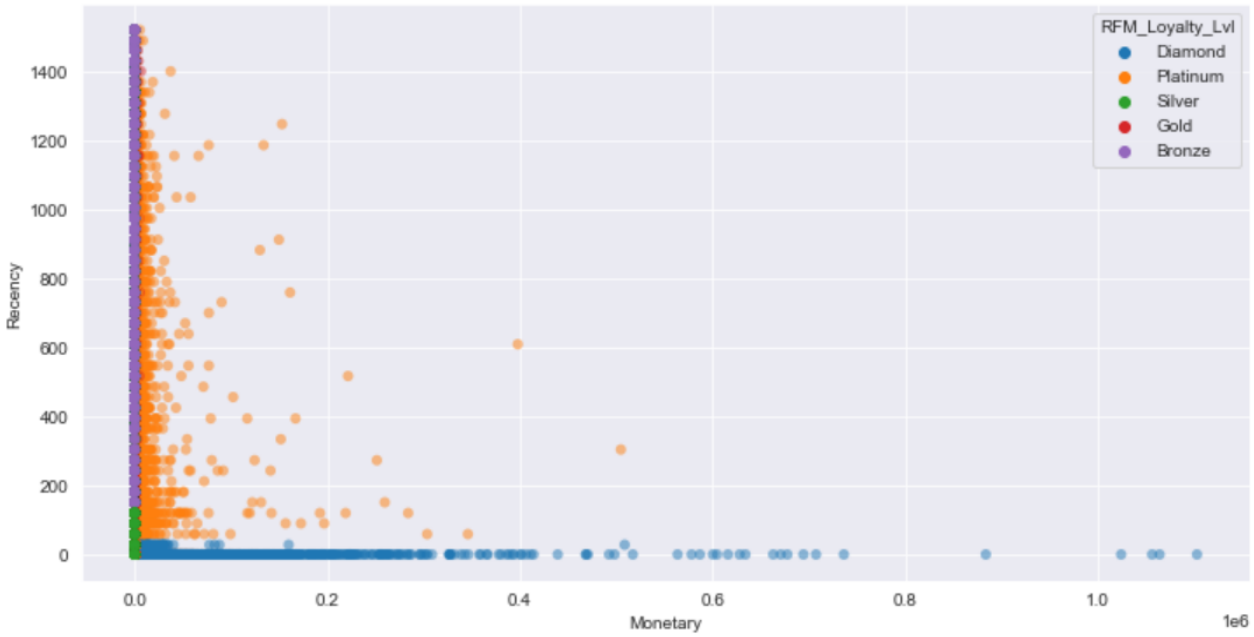
ACCOUNT_ID	Recency	Frequency	Monetary	R	F	M	RFM	RFM_Loyalty_Lvl
10100079	0	35365.0	87971.62	5	5	5	15	Diamond
10100100	1247	29394.0	153072.61	1	5	5	11	Platinum
10100111	0	9482.0	22657.79	5	5	5	15	Diamond
10100147	0	8202.0	48379.68	5	5	5	15	Diamond
10100160	0	7720.0	19646.32	5	5	5	15	Diamond
...	...	...	...	...	...	...	...	...
21177906	0	1.0	2.55	5	1	1	7	Silver
21178097	0	6.0	14.05	5	1	1	7	Silver
21178104	0	1.0	2.55	5	1	1	7	Silver
21178270	0	5.0	11.25	5	1	1	7	Silver
21178350	0	12.0	23.60	5	1	1	7	Silver

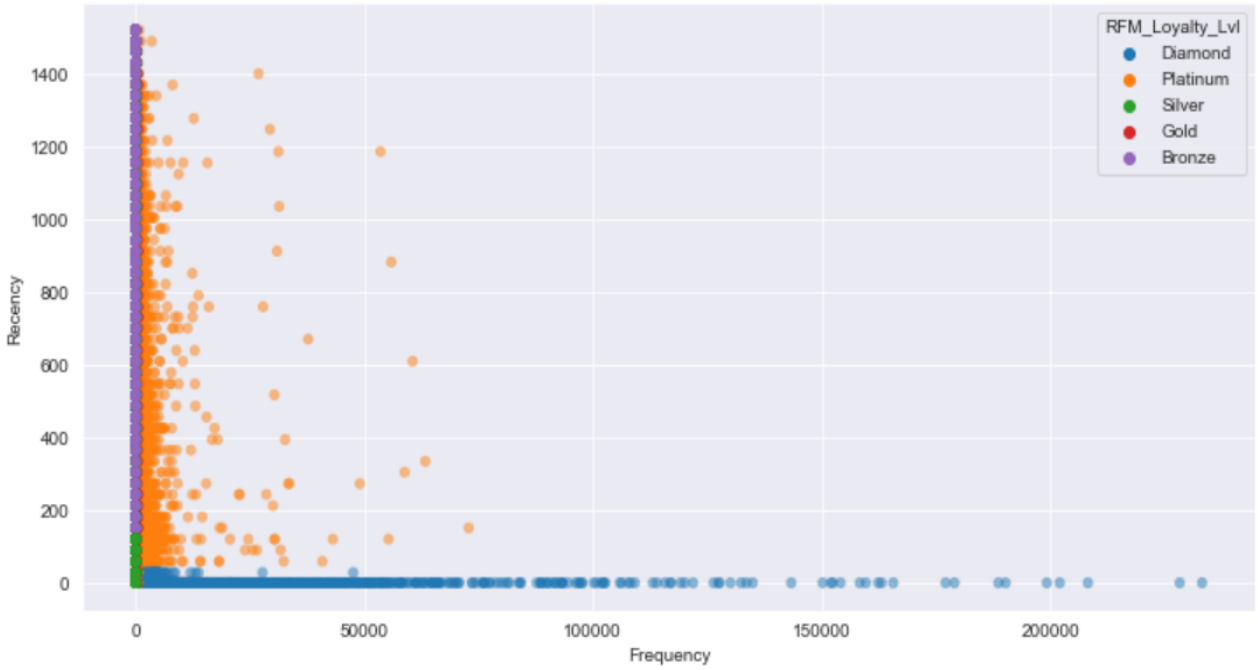
484433 rows × 8 columns

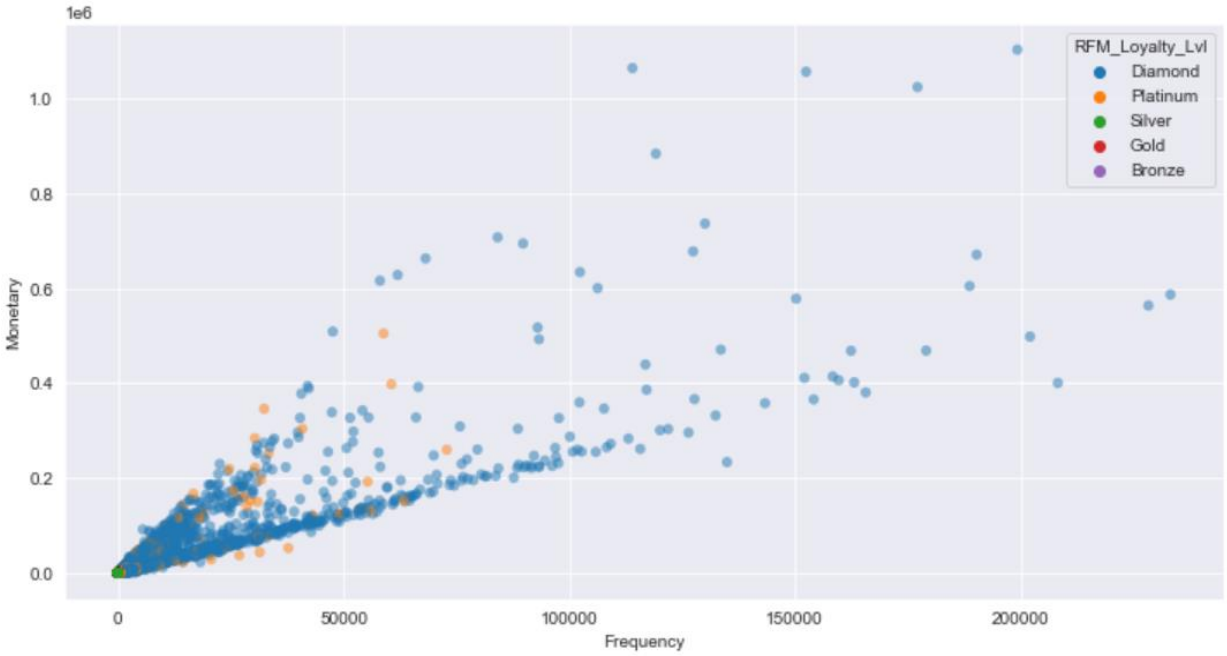
Παρακάτω εικονίζεται το πλήθος των account που αντιστοιχήθηκε σε κάθε τμήμα:



Βάσει του επιπέδου κατάταξης των account, προέκυψαν οι ακόλουθες γραφικές αναπαραστάσεις για τα στοιχεία των στηλών Recency, Frequency και Monetary, βάσει το επίπεδο που κατατάχθηκαν:







Βάσει των παραπάνω αποτελεσμάτων εξήχθησαν τα ακόλουθα συμπεράσματα:

- Για υψηλές τιμές συνολικού λογαριασμού (Monetary) ή και συχνότητας διέλευσης (Frequency), το χρονικό διάστημα που μεσολάβησε από τη πιο πρόσφατη διέλευση μέχρι την ημερομηνία παρατήρησης (Recency) φάνηκε να παίζει σημαντικό ρόλο αφού για πολύ μικρές τιμές του, τα account αυτά κατατάχθηκαν στο Diamond επίπεδο, δηλαδή τους καλύτερους πελάτες, ενώ για το υπόλοιπο φάσμα τιμών του κατατάχθηκαν στο αμέσως επόμενο επίπεδο, δηλαδή το Platinum.

- Για μεσαίες τιμές λογαριασμού ή και συχνότητας διέλευσης από τις προαναφερθέντες, το χρονικό διάστημα που μεσολάβησε από τη πιο πρόσφατη διέλευση μέχρι την ημερομηνία παρατήρησης δε φάνηκε να παίζει σημαντικό ρόλο , μια και σε όλο το φάσμα υπήρξαν accounts καταχωρημένα στο τρίτο επίπεδο, δηλαδή το Gold.

- Για μικρές τιμές λογαριασμού ή και συχνότητας διέλευσης, το χρονικό διάστημα που μεσολάβησε από τη πιο πρόσφατη διέλευση μέχρι την ημερομηνία παρατήρησης φάνηκε ωστόσο να παίζει ξανά σημαντικό

ρόλο, αφού για τις μικρές τιμές του, τα account αυτά κατατάχθηκαν στο Silver επίπεδο, δηλαδή τους πελάτες ένα επίπεδο πάνω από τους χειρότερους, ενώ για το υπόλοιπο φάσμα τιμών του κατατάχθηκαν στο αμέσως τελευταίο επίπεδο, δηλαδή το Bronze που περιλάμβανε τους χειρότερους πελάτες.

5.3 Κατάτμηση Πελατών μέσω του Αλγόριθμου K-Means

5.3.1 Κατασκευή του Μοντέλου

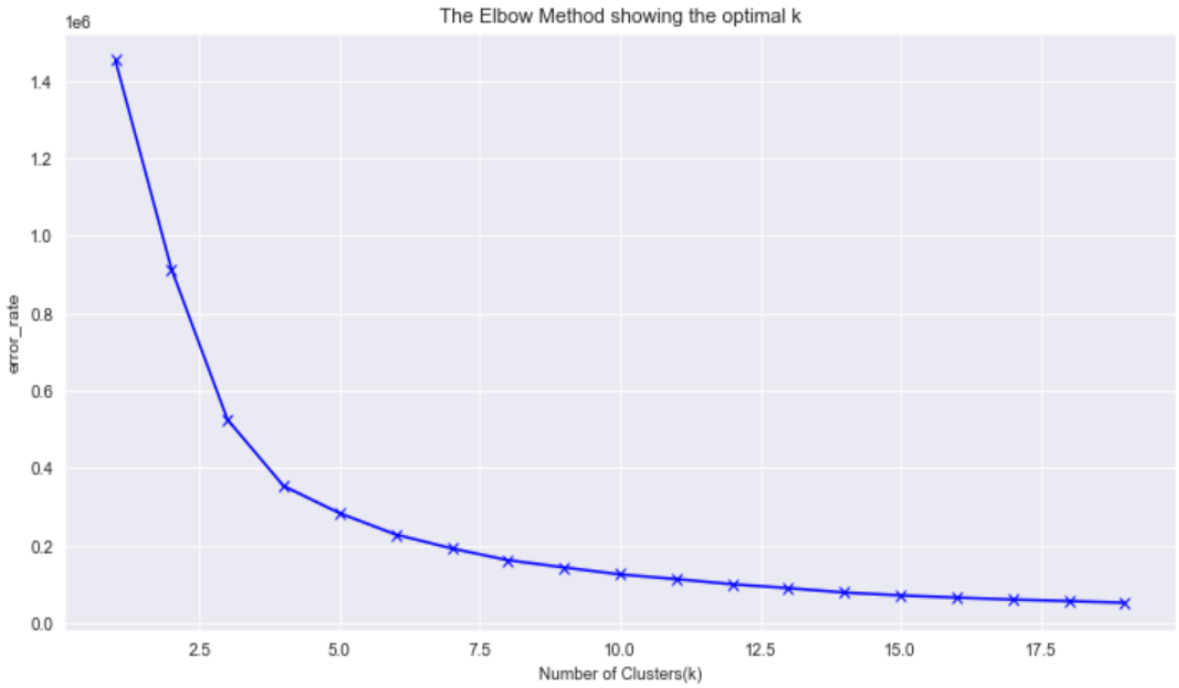
Μια εναλλακτική μέθοδος για το προσδιορισμό των επιπέδων κατά τη διαδικασία κατάτμησης των accounts ήταν μέσω της επιβολής του αλγορίθμου μηχανικής μάθησης K-Means. Όπως διατυπώθηκε αναλυτικά στο 2ο Κεφάλαιο, ο αλγόριθμος αυτός ήταν ένας αλγόριθμος συσταδοποίησης, μη επιβλεπόμενης μηχανικής μάθησης.

Η βασική ιδέα της κατάτμησης πελατών μέσω του αλγορίθμου K-Means ήταν και εδώ ο προσδιορισμός των μεγεθών Recency, Frequency και Monetary τα οποία λειτουργούσαν ως είσοδοι στον αλγόριθμο. Δεδομένου ότι τα μεγέθη αυτά προσδιορίστηκαν στη παραπάνω διαδικασία κατάτμησης κατά RFM, χρησιμοποιήθηκαν αυτούσια από το DataFrame RFMScores που ήταν αποθηκευμένα.

Για τη σωστή λειτουργία των τιμών των στοιχείων των στηλών Recency, Frequency και Monetary χρειάστηκε πρώτα να γίνει κανονικοποίηση. Αυτό επιτεύχθηκε με τη συνάρτηση StandardScaler() της βιβλιοθήκης scikit-learn, η οποία αφαιρούσε από κάθε στοιχείο τη μέση τιμή του συνόλου στην εκάστοτε στήλη και στο υπόλοιπο αυτό διαιρούσε τη τυπική απόκλιση. Έτσι δημιουργήθηκε νέο DataFrame ονόματος scaled_data το οποίο ως δείκτες είχε και αυτό τα accounts των πελατών, ενώ για στήλες περιλάμβανε τις πλέον κανονικοποιημένες Recency, Frequency και Monetary.

Στη συνέχεια, προκειμένου ο αλγόριθμος K-Means να αποδώσει ικανοποιητικά, προσδιορίστηκε ο κατάλληλος ακέραιος K που θα χρησιμοποιούταν. Για να γίνει αυτό δημιουργήθηκε μια άδεια λίστα με όνομα error_rate και με δομή επανάληψης για εύρος K από 1 έως 20, ο αλγόριθμος K-Means εφαρμόστηκε στο DataFrame scaled_data. Για κάθε κύκλο επανάληψης, στη λίστα error_rate συμπληρωνόταν το αποτέλεσμα της συνάρτησης inertia_ που δήλωνε πόσο καλά

απέδωσε ο αλγόριθμος. Για την οπτικοποίηση του μέτρου απόδοσης του αλγόριθμου για τις 20 δοκιμές που πραγματοποιήθηκαν χρησιμοποιήθηκε η συνάρτηση plot η οποία δέχτηκε ως όρισμα τη παραπάνω λίστα. Η γραφική αναπαράσταση που προέκυψε ήταν η ακόλουθη:



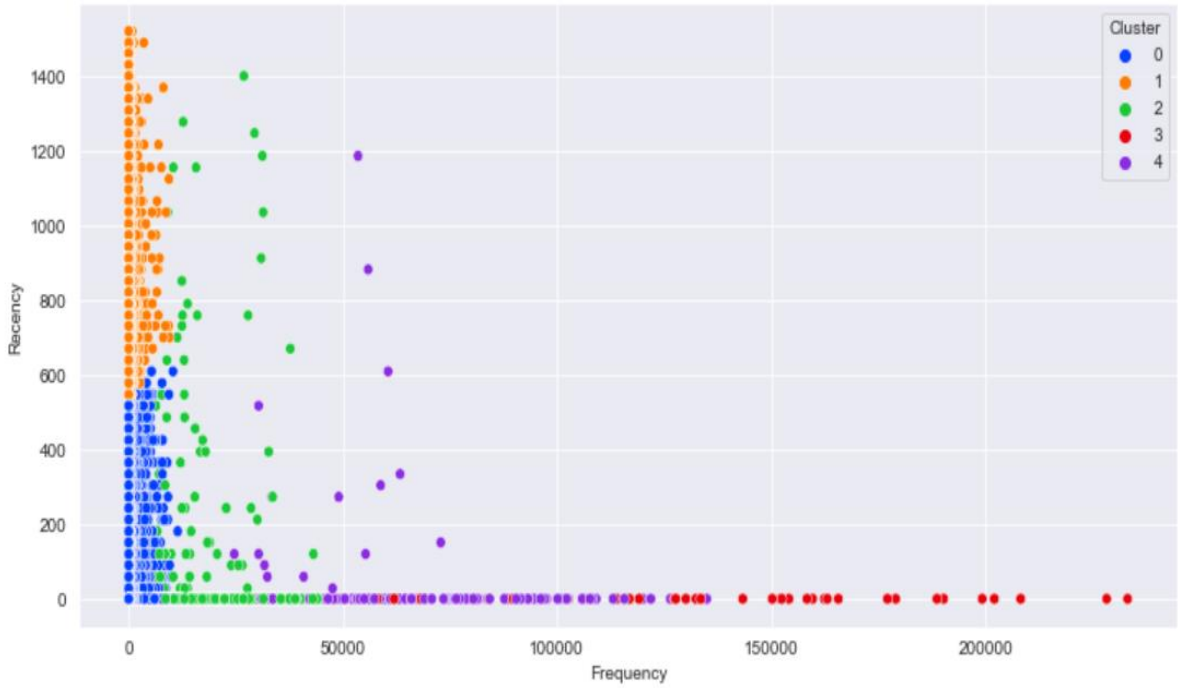
Σύμφωνα με τη γραφική, παρατηρήθηκε ότι για τιμή του K ίση με 5, η καμπύλη γονάτου βρισκόταν σε ικανοποιητική κλίση πράγμα που σήμαινε ότι για μεγαλύτερες τιμές του K , το error_rate δε θα μειωνόταν σημαντικά. Ωστόσο, η τιμή αυτή επιλέχθηκε και με γνώμονα τη μετέπειτα σύγκριση των 2 μεθόδων μια και για K=5 προφανώς προέκυπταν 5 διαφορετικά επίπεδα κατάτμησης. Συνεπώς για το συγκεκριμένο K τα στοιχεία scaled_data εφαρμόστηκαν για ακόμη μια φορά στον αλγόριθμο K-Means και, εφαρμόζοντας τη συνάρτηση labels_ στα στοιχεία, δημιουργήθηκε νέα στήλη με όνομα Cluster στο αρχικό DataFrame η οποία έδινε τον αριθμό της ομάδας (0 με 4) που αντιστοιχούσε σε κάθε account.

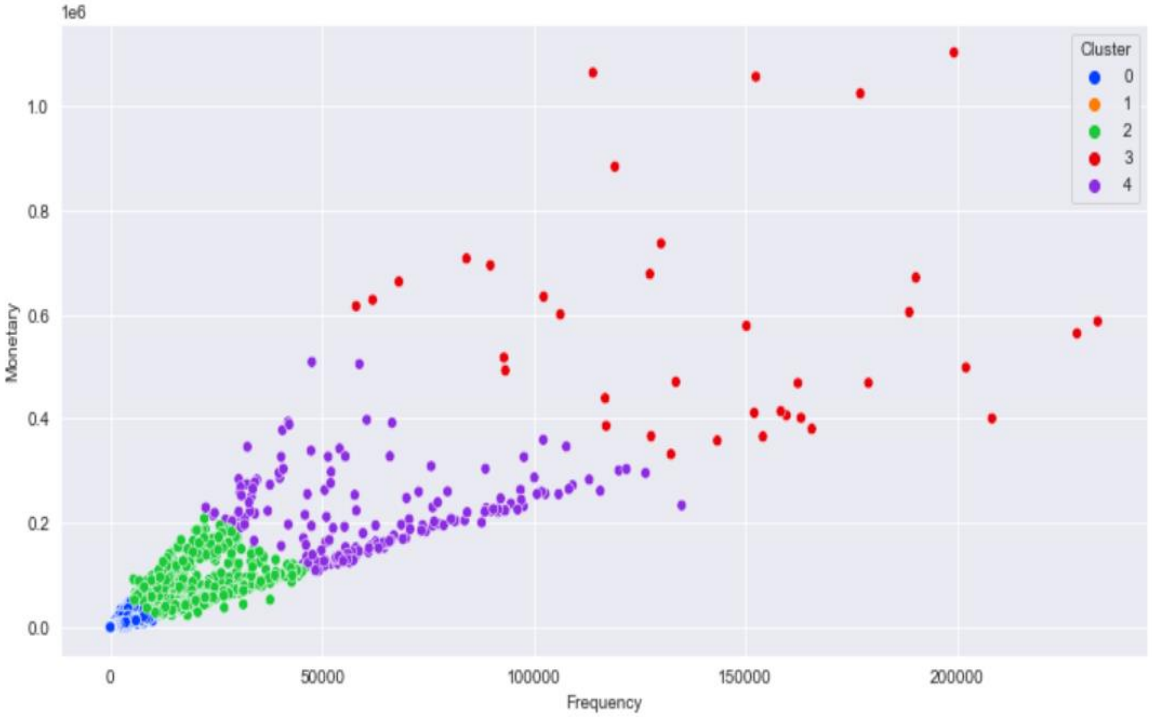
5.3.2 Ερμηνεία των Αποτελεσμάτων

Έχοντας αντιστοιχήσει κάθε account σε ένα εκ των 5 clusters, αρχικά υπολογίστηκε το πλήθος των στοιχείων ανά ομάδα, μέσω της συνάρτησης value-counts():

```
1    411627
2     71287
0      1221
3       254
4        44
Name: Cluster, dtype: int64
```

Όπως φάνηκε, η κατανομή των στοιχείων δεν ήταν σε καμία περίπτωση ομοιόμορφη μια και η συντριπτική πλειοψηφία των στοιχείων ανήκε σε 2 από τις 5 ομάδες. Οπτικά, η κατηγοριοποίηση ανά ομάδα για τις εισόδους Recency, Frequency και Monetary ήταν οι εξής:





Βάσει των παραπάνω αποτελεσμάτων εξήχθησαν τα ακόλουθα συμπεράσματα:

- Για τις πιο υψηλές τιμές συνολικού λογαριασμού (Monetary) ή και συχνότητας διέλευσης (Frequency), το χρονικό διάστημα που μεσολάβησε από τη πιο πρόσφατη διέλευση μέχρι την ημερομηνία παρατήρησης (Recency) φάνηκε να παίζει σημαντικό ρόλο αφού μόνο για τις πολύ μικρές τιμές του, τα account αυτά κατατάχθηκαν στη «Κόκκινη» ομάδα, η οποία φαινομενικά αντιπροσώπευε τους καλύτερους πελάτες.

- Για τις αμέσως επόμενες σε κλίμακα τιμές συνολικού λογαριασμού ή και συχνότητας διέλευσης, το χρονικό διάστημα που μεσολάβησε από τη πιο πρόσφατη διέλευση μέχρι την ημερομηνία παρατήρησης φάνηκε να παίζει λιγότερο σημαντικό ρόλο αφού υπήρχε μεγάλο φάσμα τιμών του Recency για το οποίο στοιχεία κατατάχθηκαν στη «Μωβ» ομάδα η οποία φαινομενικά αντιπροσώπευε τους αμέσως επόμενους καλύτερους πελάτες.

- Για τις αμέσως επόμενες σε κλίμακα τιμές συνολικού λογαριασμού ή και συχνότητας διέλευσης, το χρονικό διάστημα που μεσολάβησε από τη πιο πρόσφατη διέλευση μέχρι την ημερομηνία παρατήρησης φάνηκε να παίζει ακόμα λιγότερο σημαντικό ρόλο αφού υπήρχαν τιμές στοιχείων σε ολόκληρο το φάσμα τιμών του Recency. Αυτά τα στοιχεία αποτέλεσαν τη «Πράσινη» ομάδα η οποία φαινομενικά αντιπροσωπεύει τους άνω του μετρίου πελάτες.

- Για τις υπόλοιπες τιμές λογαριασμού ή και συχνότητας διέλευσης, δηλαδή τις μικρότερες, το χρονικό διάστημα που μεσολάβησε από τη πιο πρόσφατη διέλευση μέχρι την ημερομηνία παρατήρησης φάνηκε να παίζει καθοριστικό ρόλο, αφού στην ουσία ήταν αυτός ο κύριος διαχωρισμός των στοιχείων. Συγκεκριμένα το φάσμα τιμών του Recency χωρίστηκε σε 2 επίπεδα με τις μικρότερες προς μεσσαίες τιμές στοιχείων να αντιστοιχίζονται στη «Μπλε» ομάδα, δηλαδή τους μεσαίους πελάτες, ενώ οι υπόλοιπες στη «Πορτοκαλί», δηλαδή τους χειρότερους.

5.4 Σύγκριση των 2 Μεθόδων

Η σύγκριση των 2 μεθόδων πραγματοποιήθηκε σε 2 πλαίσια: Το γενικό-θεωρητικό το οποίο αφορούσε τα πλεονεκτήματα και τα μειονεκτήματα κάθε μεθόδου και το ειδικό, που επικεντρώθηκε στα συμπεράσματα της συγκεκριμένης ανάλυσης.

Ξεκινώντας με την ανάλυση κατά RFM, ως θετικά χαρακτηριστικά θα μπορούσαν να καταλογιστούν τα εξής:

- Χρησιμοποιεί αντικειμενικές, αριθμητικές κλίμακες που αποδίδουν μια συνοπτική και κατατοπιστική απεικόνιση υψηλού επιπέδου των πελατών.

- Είναι απλή στη χρήση και υλοποίησή της, με την έννοια ότι παρέχει πολύ ικανοποιητικά αποτελέσματα χωρίς την ανάγκη επιστημόνων δεδομένων ή εξελιγμένου λογισμικού για να την εφαρμογή της.

- Είναι μια μέθοδος διαισθητική και εύκολη στη κατανόηση. Τα αποτελέσματα δηλαδή που δίνει είναι απτά και ερμηνεύονται χωρίς ιδιαίτερη δυσκολία.

Από την άλλη, η μέθοδος RFM παρουσιάζει κάποιους περιορισμούς:

- Το γεγονός ότι χρησιμοποιεί αποκλειστικά 3 μεταβλητές μπορεί καμιά φορά να υπεραπλουστεύει το αποτέλεσμα, αφού δε λαμβάνει υπόψη παραμέτρους όπως φύλο, ηλικία, μέρος κατοικίας, εισόδημα, τα οποία προσδίδουν περισσότερη πληροφορία, άρα πιθανώς πιο κατατοπιστικό διαχωρισμό.

- Δεδομένου ότι η διακύμανση των στοιχείων κάθε επιπέδου είναι συνήθως αρκετά μεγάλη, αναπόφευκτα οι ομάδες απαρτίζονται από στοιχεία τα οποία ορισμένα ενδέχεται να αποκλίνουν σημαντικά μεταξύ τους.

- Κατά τη μέθοδο αυτή είναι πολύ δύσκολο να εκτελεστεί η τμηματοποίηση σε περισσότερες από δύο διαστάσεις.

Σε ότι αφορά τα πλεονεκτήματα της μεθόδου συσταδοποίησης (Clustering) για τη Κατάτμηση Πελατών αναφέρονται:

- Η πρακτικότητα. Αυτό διότι είναι αδύνατο να χρησιμοποιηθούν προκαθορισμένοι κανόνες για τον ακριβή διαχωρισμό των πελατών σε πολλές διαστάσεις.

- Η στατιστικά καλύτερη ομοιογένεια των στοιχείων μεταξύ των ομάδων. Δηλαδή οι παραλλαγές σε κάθε ομάδα που προκύπτουν είναι πολύ μικρότερες στην ανάλυση ομαδοποίησης, αντίθετα από τη κατάτμηση βάσει κανόνων όπως η RFM.

- Η Δυναμική ομαδοποίηση των στοιχείων μια και οι ορισμοί των συμπλεγμάτων αλλάζουν κάθε φορά που εκτελείται ο αλγόριθμος ομαδοποίησης, διασφαλίζοντας ότι οι ομάδες αντανακλούν πάντα με ακρίβεια την τρέχουσα κατάσταση των δεδομένων.

Ωστόσο στα μειονεκτήματα της παραπάνω μεθόδου καταλογίζονται τα εξής:

- Ο a priori καθορισμός των τιμών Κ. Για να είναι αποτελεσματική η συσταδοποίηση των μέσων Κ, πρέπει να καθοριστεί ο αριθμός των συμπλεγμάτων (Κ) στην αρχή εκτέλεσης ενός αλγορίθμου, πράγμα που αυξάνει τη πολυπλοκότητα της μεθόδου. Μια τυχαία επιλογή προτύπων συμπλέγματος αποφέρει διαφορετικά αποτελέσματα ομαδοποίησης με αποτέλεσμα ασυνέπεια.

- Η εκθετικά χαμηλότερη ταχύτητα εκτέλεσης όσο αυξάνονται τα δεδομένα. Στην ουσία για μεγάλα σύνολα δεδομένων, ο χρόνος εκτέλεσης του αλγορίθμου καθίσταται απαγορευτικός για την εφαρμογή.

- Ίσως το σημαντικότερο μειονέκτημα της συσταδοποίησης για κατάτμηση πελατών αποτελεί η δυσκολία ερμηνείας των αποτελεσμάτων. Αναλόγως τα δεδομένα, και τον αριθμό Κ που χρησιμοποιήθηκε, η ερμηνεία των αποτελεσμάτων μπορεί από προφανής να γίνει αδύνατη.

Ειδικότερα, για τις συγκεκριμένες αναλύσεις, τα συμπεράσματα ήταν τα εξής:

- Κατά την RFM ανάλυση, τα επίπεδα διαχωρισμού των ομάδων περιείχαν σχεδόν ίσο πλήθος account, κάτι που δε συνέβη στην ανάλυση με K-Means αφού σε εκείνη σχεδόν όλα τα accounts ήταν συγκεντρωμένα στις 2 εκ των 5 ομάδων. Από τη μια μεριά αυτό ήταν καλό γιατί κάθε ομάδα φαινομενικά είχε λιγότερη διακύμανση, ωστόσο η υπερβολικά μεγάλη συγκέντρωση των πελατών σε 2 ομάδες ενδεχομένως να μείωνε τη πρακτική αξία που έχει η διαδικασία κατάτμησης για μια εταιρία, αφού υπήρχαν ομάδες με μονάχα μερικές δεκάδες στοιχεία.

- Κατά την μέθοδο συσταδοποίησης, όπως έδειξαν τα αποτελέσματα δόθηκε ιδιαίτερη βαρύτητα στο διαχωρισμό μέσω της στήλης Monetary αφού κατά βάση παρατηρούνταν αισθητός διαχωρισμός των ομάδων όσο μειονόταν ο λογαριασμός των account. Στην ουσία, οι περισσότεροι πελάτες συγκεντρώθηκαν σε μια με δυο ομάδες που είχαν σχετικά κοινούς λογαριασμούς και διελεύσεις, ενώ για τις πιο ακραίες περιπτώσεις γινόταν μεγαλύτερη διάκριση. Ωστόσο αυτό, είχε σαν επίπτωση να μη λαμβάνονται με την ίδια βαρύτητα υπόψη οι άλλες 2 μεταβλητές δημιουργώντας μια προκατάληψη των αποτελεσμάτων.

Για τους παραπάνω λόγους, για τα συγκεκριμένα δεδομένα κρίθηκε καλύτερη επιλογή η ανάλυση κατά RFM.

Κεφάλαιο 6
Churn Prediction

6.1 Εισαγωγή

Ο όρος Churn ή αλλιώς Μετάπτωση σε Αδράνεια αναφέρεται στο φαινόμενο διακοπής της αλληλεπίδρασης ενός πελάτη/χρήστη/συνδρομητή με τα προϊόντα ή τις υπηρεσίες μιας εταιρίας. Για παράδειγμα η επιλογή ακύρωσης ή μη ανανέωσης συνδρομής, η παύση των συνδιαλλαγών όπως αγορές και επενδύσεις ή ακόμα και η παύση των επισκέψεων στην ηλεκτρονική ιστοσελίδα από πλευράς πελατών για μεγάλο χρονικό διάστημα, εμπίπτουν στη Μετάπτωση των πελατών αυτών σε Αδράνεια. Η πλήρης ζημία που επιφέρουν σε μια εταιρία οι πελάτες οι οποίοι έγιναν Αδρανείς περιλαμβάνει τόσο τα χαμένα έσοδα από την απουσία τους, όσο και το κόστος μάρκετινγκ που απαιτείται για την αντικατάστασή τους με νέους. Άλλωστε, η απόκτηση νέων πελατών είναι μια διαδικασία πολύ πιο δύσκολη και δαπανηρή από τη διατήρηση των ήδη υπαρχόντων.

Βάσει των παραπάνω, γίνεται άμεσα αντιληπτό ότι η δυνατότητα έγκαιρης πρόβλεψης πιθανής μετάπτωσης ενός πελάτη σε αδράνεια (Churn Prediction), ενώ πιθανώς υπάρχει ακόμα χρόνος να γίνουν ενέργειες για να αλλάξει αυτό, αντιπροσωπεύει μια τεράστια πρόσθετη πιθανή πηγή εσόδων για κάθε επιχείρηση. Επιπλέον, αποτελεί κατά μια έννοια δικλίδα ασφαλείας για μια εταιρία αφού η πρότερη γνώση των εσόδων βάσει της μετρικής αυτής, σταθεροποιεί τους κινδύνους ζημίας από «απρόβλεπτες» συμπεριφορές πελατών. Συμπερασματικά, η πρόβλεψη μετάπτωσης πελατών σε αδράνεια είναι μια διαδικασία απαραίτητη, αφού συμβάλλει στη κατανόηση των προληπτικών βημάτων που πρέπει να παρθούν από μια επιχείρηση για τη διασφάλιση της ελαχιστοποίησης χαμένων εσόδων.

Η τυπική προσέγγιση στο ζήτημα της πρόβλεψης μετάπτωσης πελατών σε αδράνεια είναι η χρήση αλγορίθμων μηχανικής μάθησης που εμπίπτουν στη κατηγορία των επιβλεπόμενων και συγκεκριμένα της Κατηγοριοποίησης (Classification). Δηλαδή χρησιμοποιώντας ένα αρκετά μεγάλο σύνολο δεδομένων, το οποίο περιέχει μεταξύ άλλων πληροφορία για το ποιοι πελάτες πέρασαν σε Αδράνεια και ποιοι όχι και τροφοδοτώντας τους αλγορίθμους κατηγοριοποίησης με αυτό, εκτιμάται η πιθανότητα μετάπτωσης σε αδράνεια για κάθε πελάτη ξεχωριστά.

6.2 Προσδιορισμός της Πιθανότητας Μετάπτωσης σε Αδράνεια

6.2.1 Προετοιμασία των Δεδομένων

Όπως αναφέρθηκε και παραπάνω, για την αποδοτικότερη εκτίμηση της πιθανότητας Μετάπτωσης πελατών σε Αδράνεια, απαιτείται μεγάλος όγκος δεδομένων. Σε αντίθεση με την ανάλυση για τη Κατάτμηση Πελατών, όπου χρησιμοποιήθηκαν αποκλειστικά στοιχεία που αφορούσαν συνολικά έσοδα, διελεύσεις και το χρονικό διάστημα που μεσολάβησε από τη τελευταία διέλευση έως και την ημερομηνία παρατήρησης τα οποία όλα αντλήθηκαν από το DataFrame revenue_data_byaccounts, εδώ χρησιμοποιήθηκε ο συνδυασμός των κύριων DataFrames βάσει των accounts, ώστε να μπορούσε να αξιοποιηθεί όλη η διαθέσιμη πληροφορία. Συγκεκριμένα δηλαδή, τα DataFrames revenue_data_byaccounts, τα οποία περιείχαν τα στοιχεία Εσόδων Διελεύσεων ανά account και account_data, τα οποία στην ουσία αποτελούσαν τα Γενικά Στοιχεία Συνδρομητών ενώθηκαν σε ένα εννιαίο DataFrame με όνομα accounts, με τη βοήθεια της συνάρτησης merge(). Σημειώνεται ότι οι παράμετροι left_index και right_index δήλωναν το τρόπο με τον οποίο έγινε η ένωση, δηλαδή μέσω της τομής των γραμμών, στη προκειμένη περίπτωση των accounts.

Έχοντας τα δεδομένα συγκεντρωμένα σε ένα ενιαίο DataFrame, ή αλλιώς τις μεταβλητές-πεδία εισόδου, επόμενο βήμα αποτέλεσε ο προσδιορισμός του πεδίου-μεταβλητής στόχου, απαραίτητου στοιχείου για την εκτέλεση των αλγόριθμων Κατηγοριοποίησης, οι οποίοι απαιτούν την ύπαρξη ετικετών προκειμένου να εξάγουν αποτέλεσμα. Δεδομένου ότι το ζητούμενο ήταν η πιθανότητα ο πελάτης να περάσει σε αδράνεια, δημιουργήθηκε συνάρτηση ονόματος active, η οποία για το χρονικό διάστημα των τελευταίων 9 μηνών από την ημερομηνία παρατήρησης, επέστρεφε 1 εάν το εκάστοτε account ήταν ενεργό, διαφορετικά μηδέν. Έτσι, για νέα στήλη που δημιουργήθηκε ονόματος Active_last_9_months, καταχωρήθηκαν με τον αριθμό 0 τα account τα οποία έπαψαν να είναι χρήστες των αυτοκινητόδρομων τουλάχιστον για το τελευταίο ενιάμηνο, ενώ με 1 εκείνα που συνέχιζαν.

Χρησιμοποιώντας τη συνάρτηση value_counts() παρατηρήθηκε το πλήθος των account ανά κατηγορία έτσι ώστε να εκτιμηθεί η διαφορά στα μεγέθη:

```
1    371875
0     92856
Name: Active_last_9_months, dtype: int64
```

Στη συνέχεια, οι στήλες ‘Months_passed_since_the_most_recent_transit’ και ‘Month’, απομακρύνθηκαν από το DataFrame μια και πλέον, έδιναν πλεονάζουσα πληροφορία, πράγμα που σημαίνει ότι εάν συμπεριλαμβάνονταν στους αλγορίθμους θα αλλοιωνόταν το αποτέλεσμα.

Επόμενο βήμα αποτέλεσε η ειδική προεπεξεργασία των δεδομένων για τη χρήση τους σε αλγορίθμους μηχανικής μάθησης. Συγκεκριμένα, ο τύπος δεδομένων για τα στοιχεία κάθε στήλης-πεδίου απαιτούνταν να είναι σε αριθμητική μορφή. Για να επιτευχθεί αυτό χρησιμοποιήθηκε η συνάρτηση get_dummies() για όλες τις στήλες με κατηγορικούς τύπους δεδομένων, η οποία επέστρεφε αποτέλεσμα βάσει της μεθόδου one_hot_encoding. Η μέθοδος αυτή αποτέλεσε μέθοδο μετατροπής κάθε κατηγορικής τιμή σε μια νέα κατηγορική στήλη η οποία εκχώρησε μια δυαδική τιμή 1 ή 0 σε αυτές τις στήλες και κάθε ακέραιη τιμή παριστάνεται ως δυαδικό διάνυσμα.

Ωστόσο οι εναπομείναντες στήλες που περιείχαν αριθμητικούς τύπους δεδομένων, δηλαδή οι HOW_OLD, TOTAL_CNT και TOTAL_AMT, χρειάστηκαν και εκείνες προεπεξεργασία. Ειδικότερα, για προληπτικούς λόγους, έγινε κανονικοποίηση των μεγεθών κάθε στήλης, έτσι ώστε να παρουσιάσουν μια ομοιόμορφη κατανομή. Διαφορετικά, η αυτούσια συμπερίληψή τους στους αλγορίθμους ήταν πιθανό να αλλοιώσει τα αποτελέσματα. Συγκεκριμένα για το σκοπό αυτό, χρησιμοποιήθηκε η συνάρτηση StandardScaler() από τη βιβλιοθήκη scikit-learn, η οποία όπως προαναφέραμε στο προηγούμενο κεφάλαιο αφαιρούσε από κάθε στοιχείο τη μέση τιμή του συνόλου στην εκάστοτε στήλη και στο υπόλοιπο αυτό διαιρούσε τη τυπική απόκλιση.

Έχοντας υλοποιήσει όλες τις απαραίτητες διαδικασίες για τη προετοιμασία των δεδομένων, το επόμενο βήμα πριν την εφαρμογή των αλγορίθμων ήταν ο διαχωρισμός των στηλών-πεδίων στις μεταβλητές εισόδου (feature variables) και στη μεταβλητή στόχο (target variable). Συνεπώς ως μεταβλητή στόχος θεωρήθηκε η στήλη ‘Active_last_9_months’ και εκχωρήθηκε στη μεταβλητή-διάνυσμα y ενώ όλες οι υπόλοιπες καταχωρίστηκαν ως μεταβλητές εισόδου στη μεταβλητή-διάνυσμα X.

Τέλος, τα παραπάνω διανύσματα διαχωρίστηκαν σε 2 υποκατηγορίες. Βάσει της συνάρτησης train_test_split() από τη βιβλιοθήκη scikit-learn οι μεταβλητες X και y διασπάστηκαν σε Xtrain, Xtest και ytrain, ytest αντίστοιχα. Συγκεκριμένα το 30% των στοιχείων κάθε διανύσματος εκχωρήθηκε στις μεταβλητές test ενώ το υπόλοιπο το στις μεταβλητές train. Ως random_state ορίστηκε αυθαίρετα μια τιμή η οποία καθόριζε συγκεκριμένη δειγματοληψία, έτσι ώστε τα αποτελέσματα για επαναλαμβανόμενες εκτελέσεις των μετέπειτα αλγορίθμων να παρουσιάζουν συνέπεια.

6.2.2 Εφαρμογή των Αλγορίθμων

Πρωτίστως, τα μοντέλα αλγορίθμων Κατηγοριοποίησης που χρησιμοποιήθηκαν προέρχονταν από τη βιβλιοθήκη scikit-learn και κλήθηκαν ομαδικώς, όπως επίσης και η συνάρτηση metrics, η οποία αποτέλεσε το κύριο μέσο αξιολόγησης της απόδοσης των αλγορίθμων.

Η διαδικασία που ακολουθήθηκε με εξαίρεση τον αλγόριθμο k-neighbors ο οποίος απαιτούσε κάποια βήματα παραπάνω, ήταν παρεμφερής σε όλους τους αλγόριθμους. Συγκεκριμένα, δημιουργούνταν αντικείμενο μέσω της αντίστοιχης συνάρτησης αλγορίθμου η οποία μέσω της συνάρτησης fit() δεχόταν από τις μεταβλητές εισόδου και στόχου τα στοιχεία των Xtrain και ytrain αντίστοιχα, έτσι ώστε να εκπαιδευτεί το μοντέλο. Στη συνέχεια το ίδιο αντικείμενο βάσει της συνάρτησης predict() που ως όρισμα δεχόταν τα στοιχεία του Xtest, δημιουργούσε πρόβλεψη η οποία καταχωρούταν σε μια μεταβλητή. Τέλος, χρησιμοποιώντας την επέκταση classification report() από τη metrics() της scikit-learn με ορίσματα τη μεταβλητή πρόβλεψης και το ytest, έδινε την ακρίβεια του μοντέλου σε ποσοστό αλλά και άλλες πληροφορίες όπως η κατανομή των 2 κλάσεων (κατηγοριών που προέκυψαν).

Ξεκινώντας από τον αλγόριθμο Λογιστικής Παλινδρόμησης (Logistic Regression) τα αποτελέσματα που προέκυψαν ήταν τα εξής:

	precision	recall	f1-score	support
0	0.38	0.81	0.51	27818
1	0.93	0.66	0.78	111602
accuracy			0.69	139420
macro avg	0.65	0.74	0.64	139420
weighted avg	0.82	0.69	0.72	139420

Όπως φάνηκε, η ακρίβεια του μοντέλου ήταν 69 τοις 100. Επίσης στα ορίσματα της συνάρτησης LogisticRegression() αυξήθηκε ο αριθμός των επαναλήψεων από 100 σε 200 έτσι ώστε ο αλγόριθμος να μπορούσε να επιτύχει σύγκλιση. Επιπλέον, ορίστηκε οι κλάσεις να είναι όσο το δυνατό ισορροπημένες καθώς όπως φάνηκε και από το αποτέλεσμα οι τιμές των στοιχείων που καταχωρήθηκαν στη κλάση με αριθμό 1 ήταν πολυπληθέστερες από εκείνες που καταχωρήθηκαν στη κλάση 0, συνεπώς υπήρχε θεωρητικά μια ανισορροπία που κρίθηκε θεμιτό να περιοριστεί.

Συνεχίζοντας με τον αλγόριθμο Μηχανών Διανυσμάτων Υποστήριξης (Support Vector Machines), επιλέχτηκε συγκεκριμένα το γραμμικό μοντέλο (LinearSVC) καθώς ο μεγάλος όγκος δεδομένων δεν επέτρεψε την εξαγωγή αποτελέσματος από πιο σύνθετη μέθοδο. Τα αποτελέσματα που προέκυψαν ήταν τα εξής:

	precision	recall	f1-score	support
0	0.36	0.81	0.50	27818
1	0.93	0.64	0.76	111602
accuracy			0.67	139420
macro avg	0.64	0.72	0.63	139420
weighted avg	0.82	0.67	0.70	139420

Ο αλγόριθμος αυτός απέδωσε ελαφρώς χειρότερα από το προηγούμενο καθώς η ακρίβειά του ανερχόταν στο 67 τοις 100. Επίσης για τον ίδιο λόγο που αναφέρθηκε παραπάνω, ορίστηκε η παράμετρος class_weight ως «balanced» έτσι ώστε να ελαχιστοποιηθεί η κλίση (bias) προς τη πολυπληθέστερη κλάση.

Συνεχίζοντας, ο επόμενος αλγόριθμος που εφαρμόστηκε ήταν αλγόριθμος Δένδρου Απόφασης (Decision Tree Classifier). Τα αποτελέσματα που προέκυψαν ήταν τα εξής:

	precision	recall	f1-score	support
0	0.47	0.56	0.51	27818
1	0.88	0.84	0.86	111602
accuracy			0.78	139420
macro avg	0.68	0.70	0.69	139420
weighted avg	0.80	0.78	0.79	139420

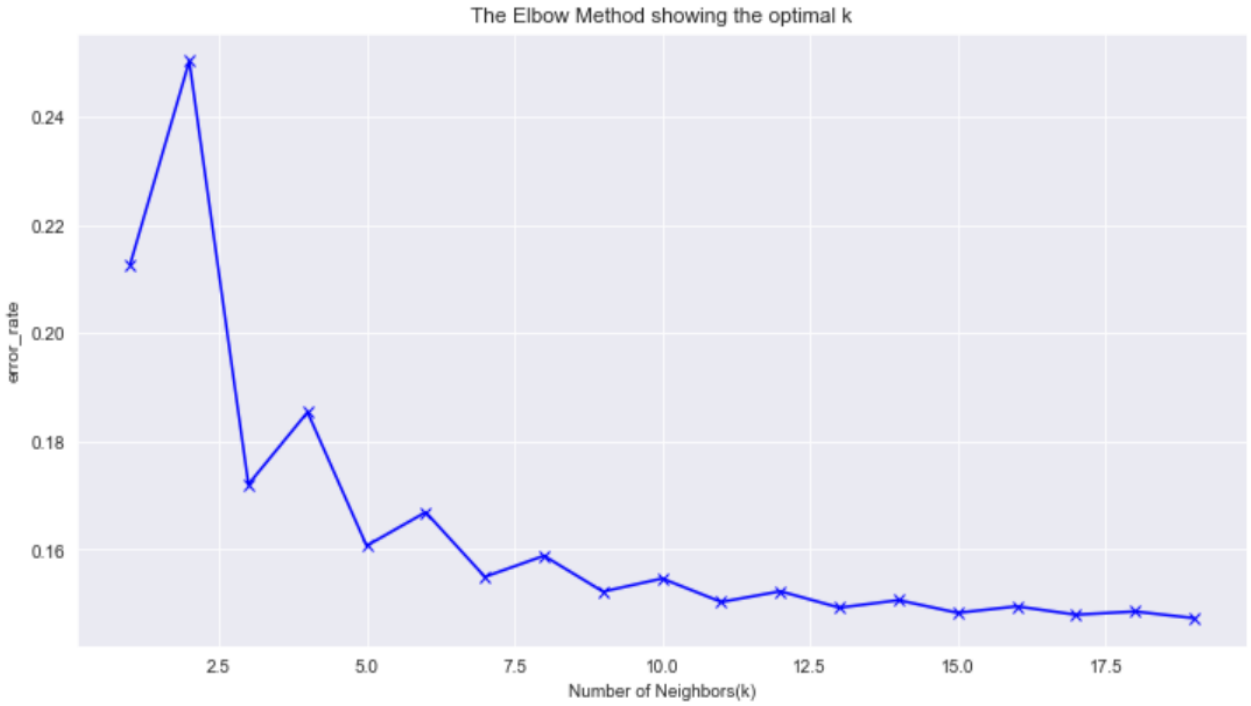
Ο παραπάνω αλγόριθμος απέδωσε αισθητά καλύτερα και από τους 2 προηγούμενους καθώς η ακρίβειά του ήταν στο 78 τοις 100. Επίσης για τον ίδιο λόγο που αναφέρθηκε παραπάνω, ορίστηκε η παράμετρος class_weight ως «balanced» έτσι ώστε να ελαχιστοποιηθεί η κλίση (bias) προς τη πολυπληθέστερη κλάση.

Συνεχίζοντας με τον αλγόριθμο Τυχαίου Δάσους (Random Forest Classifier). Τα αποτελέσματα που προέκυψαν ήταν τα εξής:

	precision	recall	f1-score	support
0	0.56	0.53	0.54	27818
1	0.88	0.89	0.89	111602
accuracy			0.82	139420
macro avg	0.72	0.71	0.72	139420
weighted avg	0.82	0.82	0.82	139420

Ο παραπάνω αλγόριθμος απέδωσε καλύτερα από τους παραπάνω μια και η ακρίβειά του άγγιξε το 82 τοις 100. Επίσης για τον ίδιο λόγο που αναφέρθηκε παραπάνω, ορίστηκε η παράμετρος class_weight ως «balanced» έτσι ώστε να ελαχιστοποιηθεί η κλίση (bias) προς τη πολυπληθέστερη κλάση.

Τέλος, ο αλγόριθμος Κ-Εγγύτεροι Γείτονες (K-nearest neighbors classifier) ήταν ο τελευταίος αλγόριθμος που εφαρμόστηκε. Ωστόσο σε αντίθεση με τους υπόλοιπους, χρειάστηκε να προσδιοριστεί το βέλτιστο Κ, για το οποίο ο αλγόριθμος απέδιδε καλύτερα. Ομοίως με τον αλγόριθμο συσταδοποίησης K-Means, χρειάστηκε η εύρεση της καμπύλης γονάτου ώστε να προσδιοριστεί το βέλτιστο Κ. Ακολουθώντας την ίδια διαδικασία με τη διαφορά ότι στη καταχώρηση της λίστας error_rate τοποθετήθηκε η μέση τιμή της διαφοράς των τιμών των μεταβλητών πρόβλεψης (knnmodel_pred) και του y_test προέκυψε το εξής γράφημα για τη καμπύλη γονάτου:



Από το παραπάνω γράφημα ήταν εμφανές ότι για κ μεγαλύτερο του 9 δεν υπήρχαν αισθητές διαφορές, συνεπώς επιλέχτηκε αυτό ως ο ακέραιος για τη τροφοδότηση του αλγορίθμου. Τα αποτελέσματα του αλγορίθμου ήταν τα εξής:

	precision	recall	f1-score	support
0	0.67	0.47	0.55	27818
1	0.88	0.94	0.91	111602
accuracy			0.85	139420
macro avg	0.77	0.70	0.73	139420
weighted avg	0.84	0.85	0.84	139420

Ο αλγόριθμος αυτός απέδωσε καλύτερα από κάθε άλλο μια και η ακρίβειά του άγγιξε το 85 τοις 100. Ωστόσο, σε σχέση με τους υπόλοιπους αλγορίθμους, ο χρόνος εκτέλεσής του, εάν συμπεριληφθεί και ο χρόνος για το προσδιορισμό του βέλτιστου Κ ήταν εκθετικά μεγαλύτερος. Συνεπώς ο αλγόριθμος Ταξινόμησης Τυχαίου Δάσους παρότι είχε λίγο μικρότερη ακρίβεια, εξισορροπούσε κατά πολύ το γεγονός ότι ήταν ταχύτερος και έτσι αποτελούσε πρακτικά καλύτερο εργαλείο για διαδοχικές παρεμφερείς μελέτες. Έχοντας πει αυτά, στην συγκεκριμένη περίπτωση που αφορούσε ειδικά μια και μόνο μέτρηση, επιλέχθηκε ο αλγόριθμος Κ-Εγγύτερων Γειτόνων προκειμένου να ληφθεί μέγιστη δυνατή ακρίβεια.

Συνεπώς για κάθε account, από τη συνάρτηση predict_proba(), η οποία επέστρεφε τη πιθανότητα να ανήκει στην εκάστοτε κλάση, επιλέχθηκε το δεύτερο στοιχείο που αφορούσε τη πιθανότητα τα στοιχεία να ανήκουν στη κλάση μηδέν. Αυτό διότι η κλάση μηδέν αφορούσε τους συνδρομητές που μετέπεισαν σε Αδράνεια, δηλαδή το ζητούμενο. Εκχωρώντας τα αποτελέσματα σε μια νέα στήλη με όνομα **Probability_of_Churn, η πιθανότητα για κάθε account να μεταβεί σε Αδράνεια ήταν:**

```
ACCOUNT_ID
10102560    88.89
10102571     0.00
10102626    33.33
10102663    55.56
10100805     0.00
            ...
16106728    11.11
16107215    22.22
16107854     0.00
21159570    44.44
21159581    44.44
Name: Propability_of_Churn, Length: 464731, dtype: float64
```

Συμπερασματικά, αν και ο προσδιορισμός της πιθανότητας μετάπτωσης σε αδράνεια είναι πολύ σημαντικός, δίχως ένα προκαθορισμένο χρονικό πλαίσιο, η πρακτική αξία του για μια επιχείρηση είναι μικρή. Ο προσδιορισμός της αξίας διάρκειας ζωής του πελάτη από την άλλη, αποτελεί μια πιο ολοκληρωμένη μετρική αφού όπως θα δειχτεί στη συνέχεια, περιλαμβάνει τη πιθανότητα μετάπτωσης σε αδράνεια για προκαθορισμένο μελλοντικό διάστημα, ως παράμετρο για το προσδιορισμό της.

Κεφάλαιο 7
Customer's Lifetime Value

7.1 Εισαγωγή

Customer's Lifetime Value ή αλλιώς Αξία της Διάρκειας Ζωής του Πελάτη (CLV ή LTV πελάτη) ορίζεται το προβλεπόμενο άθροισμα των συνολικών εσόδων ή κερδών (δηλαδή το άθροισμα τωρινών και εκτιμώμενων) που εκτιμάται ότι θα έχει δημιουργήσει ένας συγκεκριμένος πελάτης για μια επιχείρηση σε μια προκαθορισμένη χρονική περίοδο στο μέλλον. Οι ακριβείς εκτιμήσεις για το προσδιορισμό του CLV αποτελούν τη βάση για την εύρεση των καταλληλότερων τεχνικών μάρκετινγκ που θα κληθεί να χρησιμοποιήσει μια επιχείρηση ή ένας οργανισμός προκειμένου να μεγιστοποιήσει τα κέρδη της.

Αν και είναι ευκολότερο για μια εταιρία να εστιάζει στο παρόν, εντούτοις σπάνια αποτελεί το βέλτιστο τρόπο πλήρους αξιοποίησης της δυνητικής αξίας κάθε πελάτη. Για παράδειγμα, η εξέταση των ποσοστών μετατροπής και των πρώτων αγορών αγνοώντας τη μακροπρόθεσμη αξία των πελατών, ενδέχεται να οδηγήσουν στην σπατάλη πόρων για απόκτηση «φθηνών» πελατών με χαμηλή συνολική αξία εσόδων, αντί να επενδυθούν περισσότερα για την απόκτηση πελατών που θα συνεχίσουν να παρέχουν σταθερό ρεύμα εισοδήματος για τα επόμενα χρόνια. Με την ίδια λογική, οι εταιρίες και συγκεκριμένα οι εμπειρογνώμονες Διατήρησης του Πελατειακού Κοινού επικεντρώνονται στην καλλιέργεια καλύτερων πελατειακών σχέσεων με εκείνους τους πελάτες που θα συνεχίσουν να αποτελούν πηγή σημαντικών εσόδων μακροπρόθεσμα, ενώ παράλληλα εξοικονομούν πόρους που θα μπορούσαν να χαραμιστούν σε πελάτες χαμηλής αξίας.

Μολονότι απαραίτητος, ο ακριβής υπολογισμός της αξίας της διάρκειας ζωής του πελάτη αποτελεί εκ φύσεως δύσκολη διαδικασία αφού αφορά στο πυρήνα μια πρόβλεψη εξαρτώμενη από πολλές και συνήθως μη ποσοτικοποιημένες παραμέτρους. Για παράδειγμα το χρονικό διάστημα παραμονής ενός πελάτη σε μια εταιρία, ή το ποσό που θα ξοδέψει σε κάθε χρονική περίοδο, συντελούν στις παραμέτρους για το προσδιορισμό του CLV, ωστόσο η εύρεσή τους αποτελεί από μόνη της μια πρόκληση, ειδικά όταν ο πελάτης είναι νέος. Επιπρόσθετα, περαιτέρω επιπλοκές παρουσιάζονται όταν τα δεδομένα που απαιτούνται για την εκτέλεση των υπολογισμών μπορεί να κρύβονται βαθιά μέσα σε πολλές βάσεις δεδομένων. Για τους παραπάνω λόγους, δημιουργήθηκαν κατάλληλες βιβλιοθήκες οι οποίες περιλαμβάνουν συναρτήσεις οι οποίες εξυπηρετούν στο προσδιορισμό των μεγεθών που απαιτούνται για το προσδιορισμό πολύπλοκων μετρικών όπως το CLV.

7.2 Η Βιβλιοθήκη lifetimes

Η lifetimes αποτελεί μια δωρεάν βιβλιοθήκη σχεδιασμένη για τη γλώσσα προγραμματισμού Python, η οποία περιλαμβάνει μαθηματικά μοντέλα, χρήσιμες συναρτήσεις και γραφήματα που χρησιμοποιούνται κατά κύριο λόγο για διευκόλυνση σε εφαρμογές Ανάλυσης Επιβίωσης (Survival Analysis). Η τελευταία αποτελεί μια συλλογή στατιστικών διαδικασιών για την ανάλυση δεδομένων, με τη μεταβλητή αποτελέσματος στην οποία επικεντρώνεται να είναι ο χρόνος μέχρι να συμβεί ένα συμβάν. Κατά μια έννοια, ο προσδιορισμός του CLV στο πυρήνα του χρησιμοποιεί στοιχεία από την Ανάλυση Επιβίωσης, καθώς εκτιμάται ο χρόνος για τον οποίο ο πελάτης αποφέρει κέρδη στην εταιρία. Ωστόσο οι μαθηματικές φόρμουλες πίσω από αυτά τα μοντέλα είναι αρκετά περίπλοκες, συνεπώς η ενσωμάτωσή τους στη συγκεκριμένη βιβλιοθήκη και η χρήση τους, καθιστά το προσδιορισμό του CLV αρκετά πιο ακριβή.

Τα βασικά μεγέθη τα οποία χρησιμοποιούν όλοι οι αλγόριθμοι και τα μοντέλα της βιβλιοθήκης lifetimes, μοιάζουν με τα μεγέθη που χρησιμοποιήθηκαν κατά τη τμηματοποίηση πελατών μέσω του μοντέλου RFM, ωστόσο δεν έχουν την ίδια σημασία. Συγκεκριμένα, τα 4 βασικά αυτά μεγέθη είναι:

- frequency: Αντιπροσωπεύει τον αριθμό των επαναλαμβανόμενων αγορών (άρα μια λιγότερη από το σύνολο) που έχει κάνει ο πελάτης ανά συγκεκριμένη χρονική περίοδο. Είναι δηλαδή ο αριθμός των χρονικών περιόδων που πραγματοποίησε ο πελάτης μια αγορά. Έτσι, εάν για παράδειγμα χρησιμοποιούνται οι μήνες ως μονάδες, τότε είναι ο αριθμός των μηνών στις οποίες ο πελάτης είχε σημειώσει έστω και μία αγορά.

- T: Αντιπροσωπεύει την «ηλικία» του πελάτη σε οποιαδήποτε χρονική μονάδα έχει επιλεγεί (για παράδειγμα ημέρες). Αυτό ισούται με τη διάρκεια μεταξύ της πρώτης αγοράς ενός πελάτη και του τέλους της υπό μελέτη περιόδου.

- recency: Αντιπροσωπεύει την «ηλικία» του πελάτη από τις πιο πρόσφατες αγορές του. Αυτό ισούται με τη διάρκεια μεταξύ της πρώτης αγοράς ενός πελάτη και της τελευταίας αγοράς. Επομένως, εάν έχει κάνει μόνο 1 αγορά, καταλογίζεται η τιμή 0.

- monetary_value: Αντιπροσωπεύει τη μέση αξία των αγορών ενός δεδομένου πελάτη. Αυτό ισούται με το άθροισμα όλων των τιμών από αγορές ενός πελάτη, διαιρούμενο με το συνολικό αριθμό αγορών.

7.3 Προσδιορισμός της Αξίας του Πελάτη

7.3.1 Προετοιμασία και πρώτη Ανάλυση των Δεδομένων

Για τη κατάλληλη ομαδοποίηση των δεδομένων και αυτόματο προσδιορισμό των 4 παραπάνω μεγεθών, χρησιμοποιήθηκε η συνάρτηση summary_data_from_transaction_data() από τη ύπο-βιβλιοθήκη utils της lifetimes. Η συνάρτηση αυτή δεχόταν ως ορίσματα το DataFrame revenue_data με τις στήλες του όπως ακριβώς ήταν μετά την αρχική επεξεργασία, χωρίς δηλαδή τα δεδομένα να έχουν ομαδοποιηθεί ανά account. Συγκεκριμένα επιλέχθηκαν το πεδίο ACCOUNT_ID ώστε να μπορούσε να ομαδοποιηθεί ανά account το νέο DataFrame, το πεδίο Month, η ημερομηνία παρατήρησης και το πεδίο TOTAL_AMT προκειμένου να προσδιορίζονταν οι τιμές των frequency, recency, T αλλά και του monetary_value. Το νέο DataFrame το οποίο δημιουργήθηκε από τη παραπάνω συνάρτηση ονομάστηκε summary και φαίνεται παρακάτω η μορφή του:

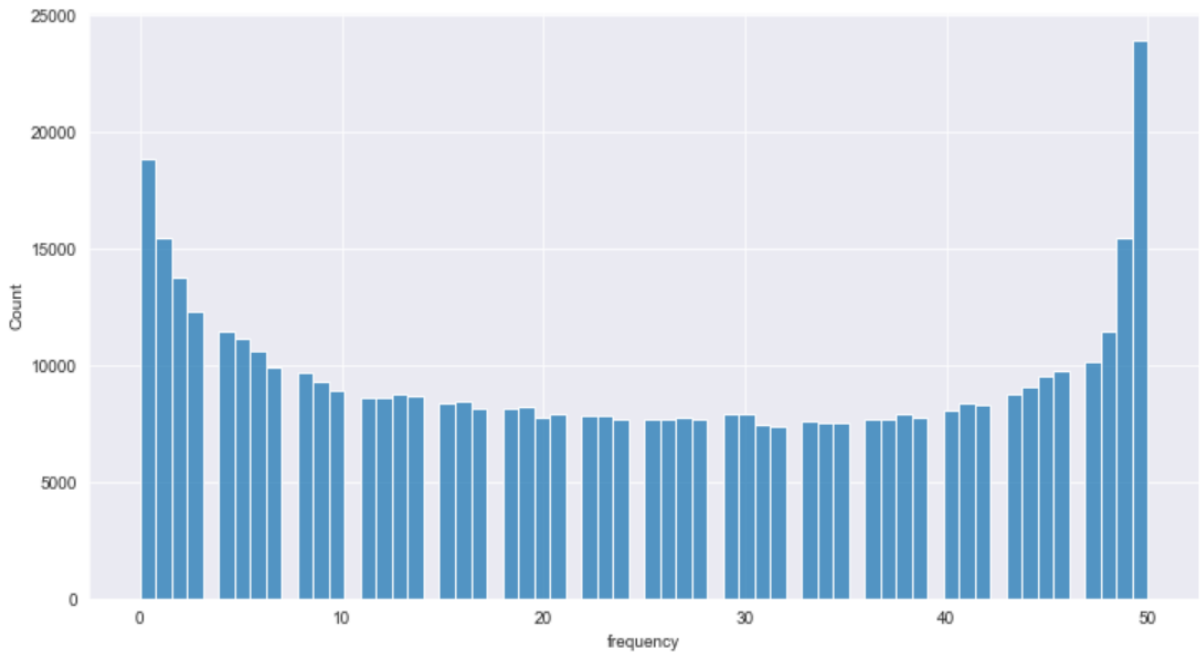
ACCOUNT_ID	frequency	recency	T	monetary_value
10100043	50.0	1520.0	1520.0	26515.780000
10100067	50.0	1520.0	1520.0	51202.053800
10100079	50.0	1520.0	1520.0	1704.913800
10100100	9.0	273.0	1520.0	14958.772222
10100111	50.0	1520.0	1520.0	447.966800
...	...	...	...	...
21177906	0.0	0.0	0.0	0.000000
21178097	0.0	0.0	0.0	0.000000
21178104	0.0	0.0	0.0	0.000000
21178270	0.0	0.0	0.0	0.000000
21178350	0.0	0.0	0.0	0.000000

485125 rows × 4 columns

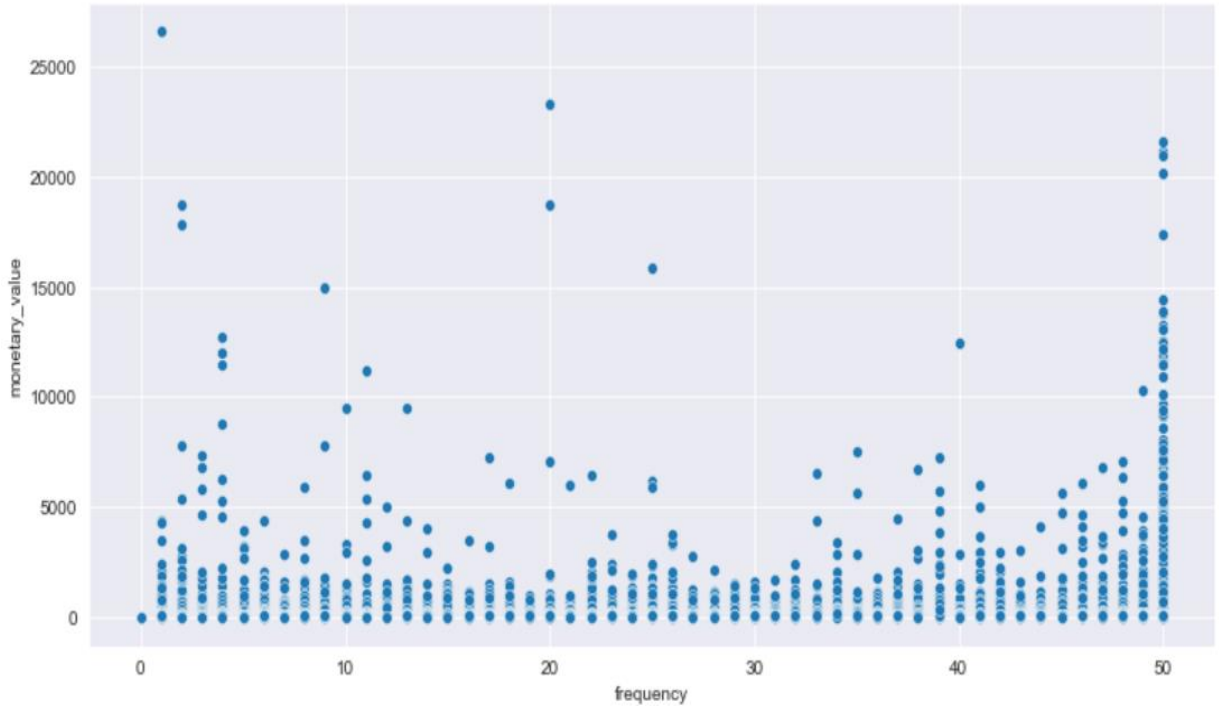
Έχοντας δημιουργήσει το νέο DataFrame το οποίο περιείχε τα 4 μεγέθη που χρειάστηκαν για την εφαρμογή αλγορίθμων και γραφικών αναπαραστάσεων, στη συνέχεια απομακρύνθηκαν τα accounts που περιείχαν ακραίες τιμές και συγκεκριμένα, τα ίδια accounts που βρέθηκαν κατά την ανάλυση των δεδομένων, προκειμένου να μην αλλοιωθούν τα αποτελέσματα εξαιτίας τους.

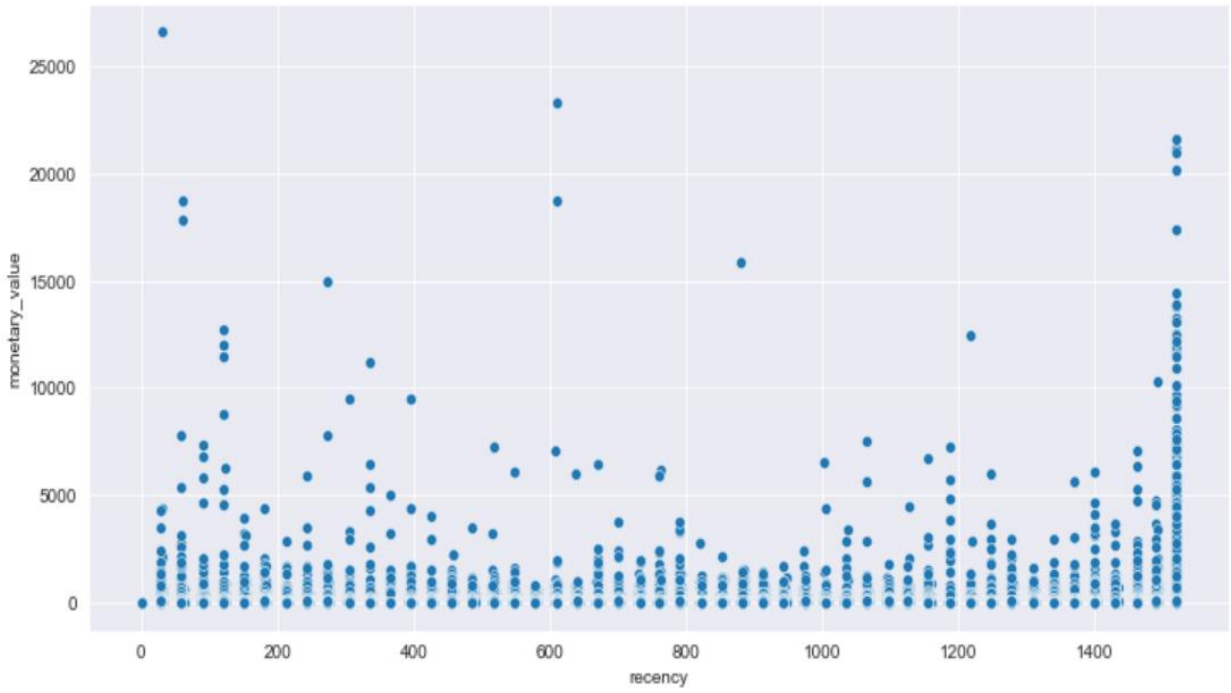
Συνεχίζοντας, δημιουργήθηκαν γραφικές αναπαραστάσεις προκειμένου να μπορούσε εκτιμηθεί η πιθανή συσχέτιση των 4 μεγεθών μεταξύ τους. Μελετώντας αρχικά μεμονωμένα τη στήλη frequency η οποία περιείχε τους επαναλαμβανόμενους μήνες για τους οποίους το εκάστοτε account ήταν ενεργό, εκτιμήθηκε ο μέσος αριθμός των μηνών για τους οποίους τα account ήταν ενεργά, αλλά και ο αριθμός των account που ήταν ενεργά για μόνο ένα μήνα:

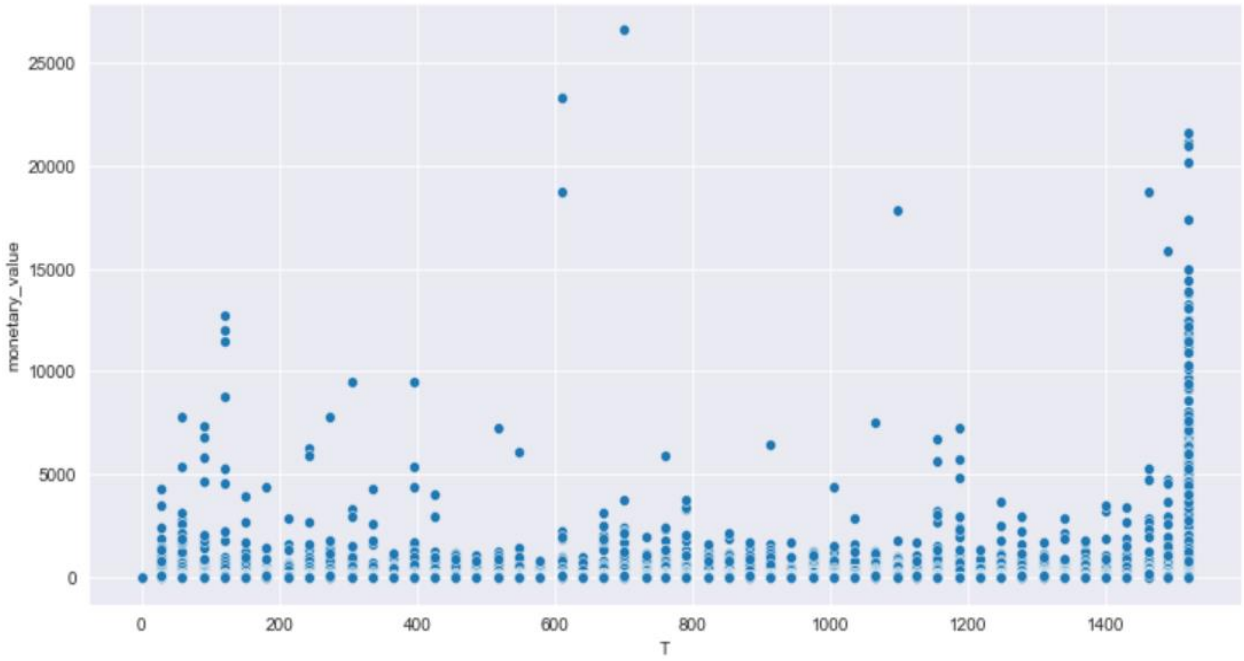
```
Οι πελάτες που χρησιμοποίησαν epass για ένα μόνο μήνα: 18839

Ο μέσος αριθμός μηνών για τον οποίο είχαμε ενεργούς πελάτες: 25
```



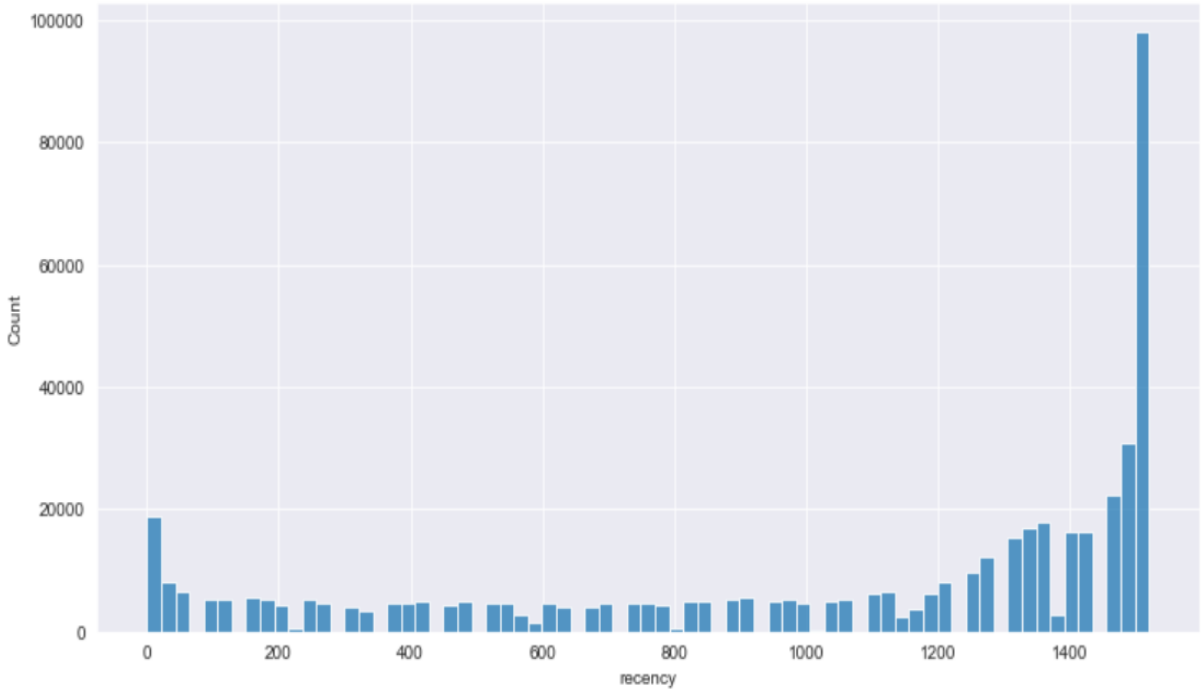
Για τις σχέσεις των monetary_value με τα υπόλοιπα μεγέθη, ακολούθησαν οι παρακάτω γραφικές αναπαραστάσεις:

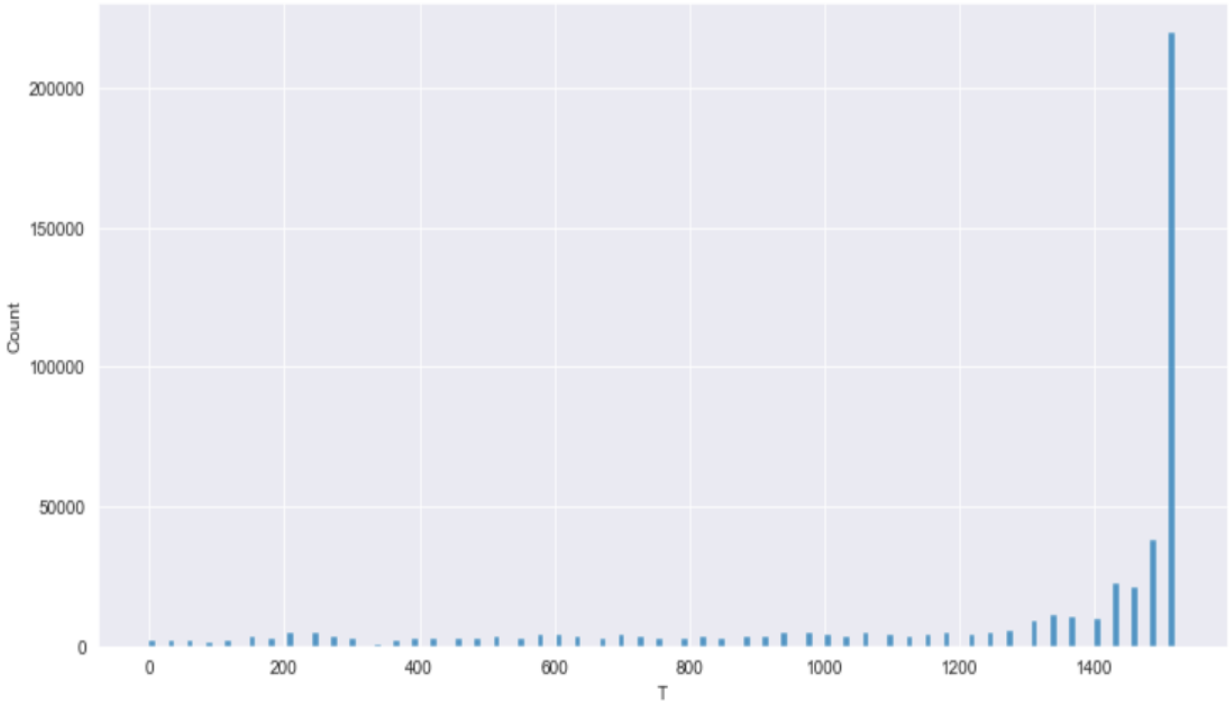

monetary_value
25000
20000
15000
10000
5000
0
0
10
20
30
40
50
frequency


monetary_value
25000
20000
15000
10000
5000
0
0
200
400
600
800
1000
1200
1400
recency



Και οι 3 γραφικές αναπαραστάσεις είχαν το εξής κοινό στοιχείο: Παρατηρήθηκε σχετικά ομοιόμορφη κατανομή του χρηματικού ποσού των account σύμφωνα με το χρονικό πλαίσιο, με εξαίρεση τα account τα οποία ήταν ενεργά καθ'όλη τη χρονική διάρκεια παρατήρησης. Εκείνα τα account δηλαδή, έκαναν αισθητά υψηλότερο λογαριασμό από όλα τα υπόλοιπα. Ο λόγος που συνέβη αυτό έγινε καλύτερα αντιληπτός από τα επόμενα 2 γραφήματα:





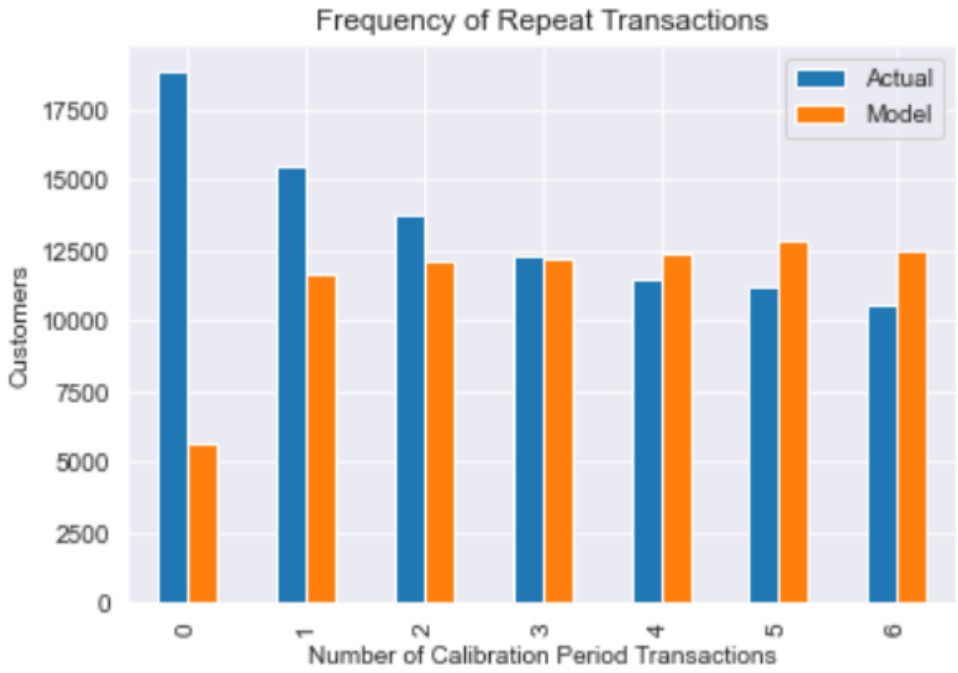
Φάνηκε λοιπόν ότι η πλειονότητα των account παρουσίαζε μεγάλη χρονική διαφορά μεταξύ πρώτου και τελευταίου μήνα σε διελεύσεις (recency), αλλά και μεγάλη χρονική διαφορά μεταξύ πρώτου μήνα διελεύσεων και ημερομηνίας παρατήρησης (T). Συνεπώς τα account αυτά ήταν λογικό να κάνουν το μεγαλύτερο λογαριασμό αφού είχαν και τις περισσότερες διελεύσεις.

7.3.2 Ανάλυση Επιβίωσης

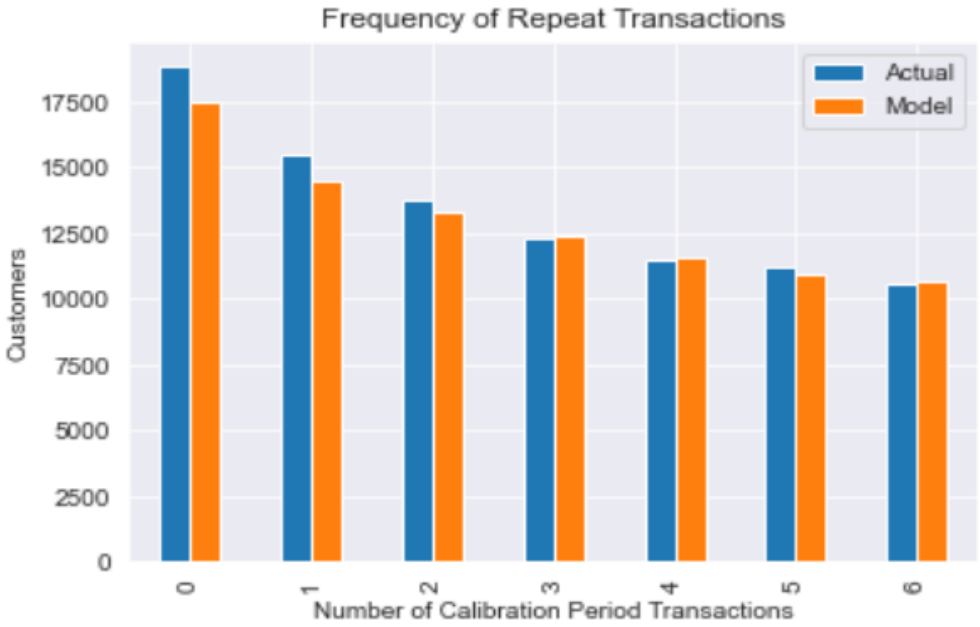
Η ανάλυση επιβίωσης (Survival Analysis) αποτέλεσε το ενδιάμεσο βήμα για το προσδιορισμό του CLV, αλλά παράλληλα προσέφερε και μια εναλλακτική μέθοδο για το προσδιορισμό της μετάπτωσης πελάτη σε αδράνεια (Churn Prediction).

Προκειμένου να επιτευχθεί αυτό, χρησιμοποιήθηκε το μοντέλο πρόβλεψης MBG/NBD. Το μοντέλο αυτό μαζί με το μοντέλο Pareto/NBD αποτελούν τα 2 κύρια στατιστικά μοντέλα της ευρύτερης οικογένειας στατιστικών μοντέλων Buy Till You Die (BTYD), η οποία έχει σχεδιαστεί για να αποτυπώνει χαρακτηριστικά συμπεριφοράς πελατών χωρίς συμβατική σχέση ή όταν η εταιρεία αδυνατεί να παρατηρήσει άμεσα πότε ένας πελάτης σταματά να είναι ανήκει ή να είναι ενεργός. Συγκεκριμένα το μοντέλο MBG/NBD μοντελοποιούσε τη διαδικασία μετάπτωσης σε αδράνεια βάση μιας μίξης γεωμετρικής και βήτα κατανομής, ενώ για τη διαδικασία συχνότητας διελεύσεων μοντελοποιήθηκε ως αρνητική διωνυμική κατανομή. Για την εφαρμογή του μοντέλου χρησιμοποιήθηκε αρχικά η συνάρτηση BetaGeoFitter() ή

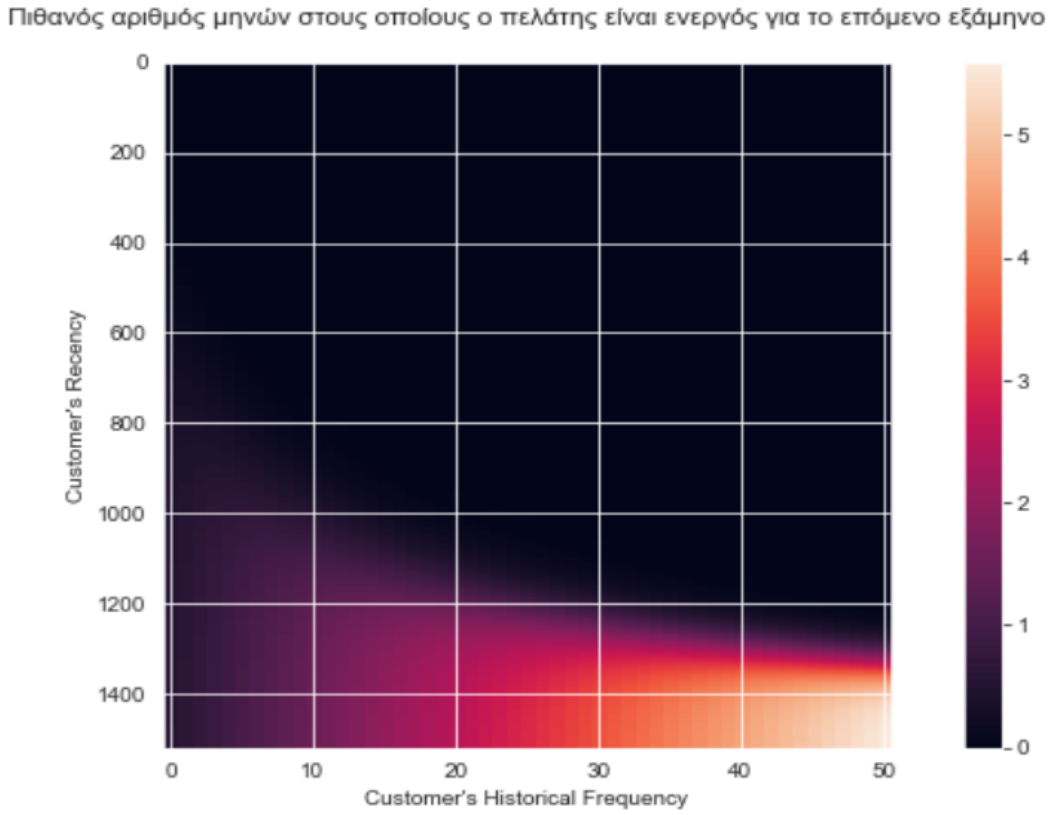
οποία ως παραμέτρους δεχόταν τις στήλες frequency, recency και T, δε λάμβανε δηλαδή υπόψη το monetary_value και δημιουργούσε αντικείμενο ονόματος bgf. Για την αξιολόγηση της απόδοσης του μοντέλου χρησιμοποιήθηκε η συνάρτηση plot_period_transactions() η οποία γραφικά παρουσίαζε εάν οι πραγματικοί υπολογισμοί συμφωνούσαν με υπολογισμούς συγκρίνοντας ομοιότητα με ένα τεχνιτό dataset το οποίο δημιουργούσε η ίδια η συνάρτηση:



Όπως διαπιστώθηκε, τα αποτελέσματα του αλγορίθμου δεν ήταν καθόλου ικανοποιητικά. Συνεπώς μετά από αλλεπάλληλες δοκιμές, χρησιμοποιήθηκε ο ελαφρώς τροποποιημένος αλγόριθμος Modified_Beta_Geo_Fitter(), ο οποίος χρησιμοποιώντας κατάλληλη διορθωτική μεταβλητή (penalizer_coef) με τιμή (0.0001) της οποίας η τάξη μεγέθους έδινε καλύτερα αποτελέσματα, μετά από την εφαρμογή του και αξιολόγησή του λήφθηκε το εξής ικανοποιητικό αποτέλεσμα:

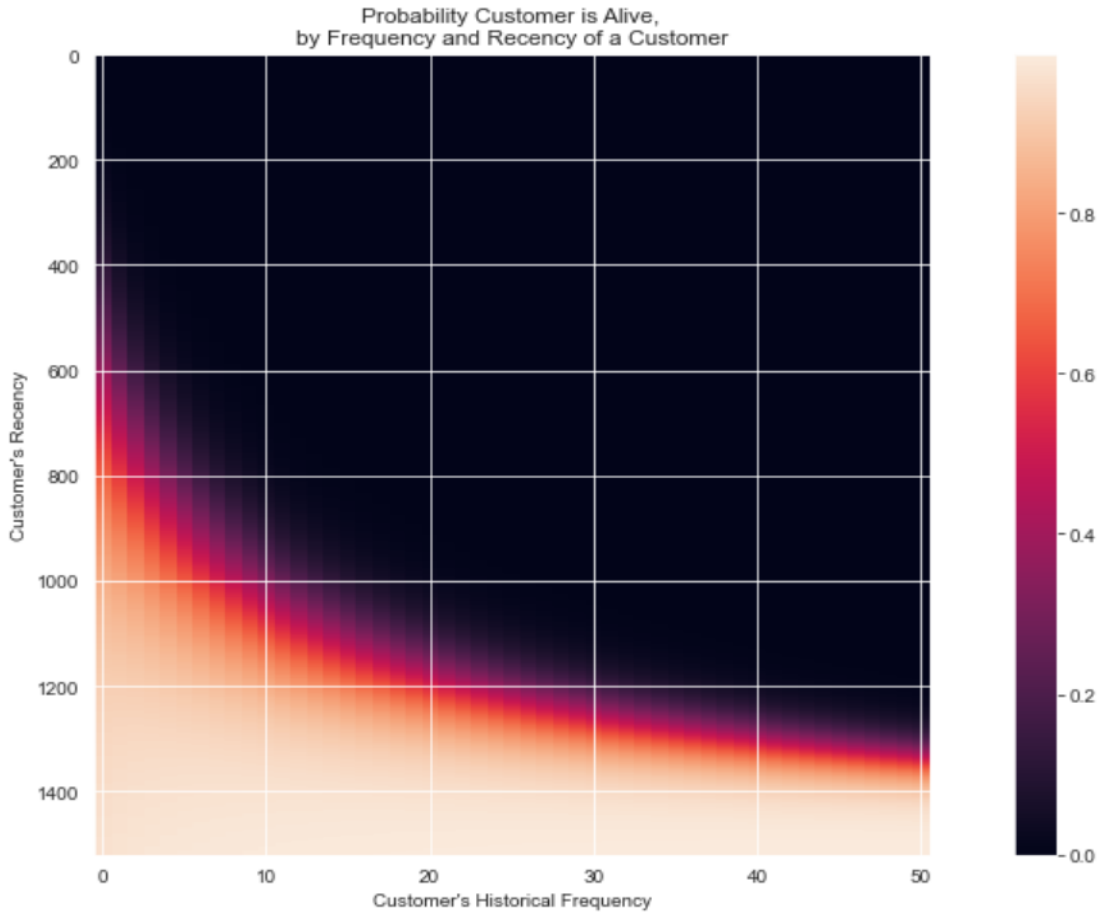


Στη συνέχεια, με τη βοήθεια της συνάρτησης plot_frequency_recency_matrix(), η οποία δεχόταν ως παραμέτρους το αντικείμενο mbgf αλλά και το επιθυμητό χρονικό διάστημα για την οποίο έγινε η πρόβλεψη, δηλαδή 180 ημέρες, εκτιμήθηκε η κατανομή σε μορφή heatmap του πιθανού αριθμού μηνών για τους οποίους το τα account θα ήταν ενεργά στους επόμενους έξι μήνες:



Από το παραπάνω γράφημα φάνηκε ότι account τα οποία είχαν υψηλό frequency (δηλαδή επαναλαμβανόμενους ενεργούς μήνες) και recency, (δηλαδή μεγάλη χρονική διαφορά από τη πρώτο μήνα που ήταν ενεργοί μέχρι το πιο πρόσφατο) ήταν σχεδόν βέβαιο ότι θα παρέμεναν ενεργοί για όλους τους επόμενους έξι μήνες. Αντίθετα, όσο μίκραιναν οι τιμές, τόσο πιο απίθανο ήταν να έχει ενεργούς μήνες στο επόμενο εξάμηνο. Ιδιαίτερο ενδιαφέρον είχε το γεγονός ότι το frequency έπαιξε σημαντικότερο ρόλο μια και για μεσαίες προς χαμηλές τιμές του, όσο υψηλό και να ήταν το recency, οι εκτιμώμενοι ενεργοί μήνες στο επόμενο εξάμηνο ήταν λίγοι ή μηδενικοί.

Χρησιμοποιώντας τη συνάρτηση plot_probability_alive_matrix() η οποία δεχόταν μόνο το αντικείμενο mbgf ως όρισμα, έδινε τη κατανομή της πιθανότητα το account να είναι ζωντανό (δηλαδή να υπάρξει ενεργό στο μέλλον) σε μορφή heatmap:



Ουσιαστικά, το παραπάνω heatmap απεικόνιζε τη πιθανότητα το εκάστοτε account να μη περνούσε ποτέ σε αδράνεια, άρα στην ουσία υλοποιήθηκε ένας

διαφορετικός τρόπος για το προσδιορισμό του Churn Prediction, χάρη δηλαδή στο μοντέλο MBG/NBD. Σε σύγκριση ωστόσο με τις μεθόδους του προηγούμενου κεφαλαίου, δε λήφθηκαν υπόψη σημαντικά χαρακτηριστικά, με κυριότερο το monetary_value, δηλαδή το συνολικό λογαριασμό που έκανε κάθε account. Επιπλέον, η γνώση της πιθανότητας αυτής δε περικλείεται σε ένα χρονικό πλαίσιο, συνεπώς και εδώ η αξία της δεν έχει πρακτική σημασία.

Λύση στο παραπάνω πρόβλημα έδωσε η μέθοδος conditional_expected_number_of_purchases_up_to_time() η οποία λάμβανε υπόψη το επιθυμητό χρονικό πλαίσιο ως όρισμα για την εκτίμηση του αριθμού των μηνών που το εκάστοτε account θα ήταν ενεργό. Συγκεκριμένα λοιπόν, χρησιμοποιώντας στρογγυλοποίηση, δημιουργήθηκε νέα στήλη με όνομα Πιθανοί ενεργοί μήνες για το επόμενο εξάμηνο, η οποία κατά φθίνουσα σειρά παρουσίαζε το πιθανό αριθμός μηνών για το οποίο το κάθε account θα ήταν ενεργό. Συνολικά λοιπόν:

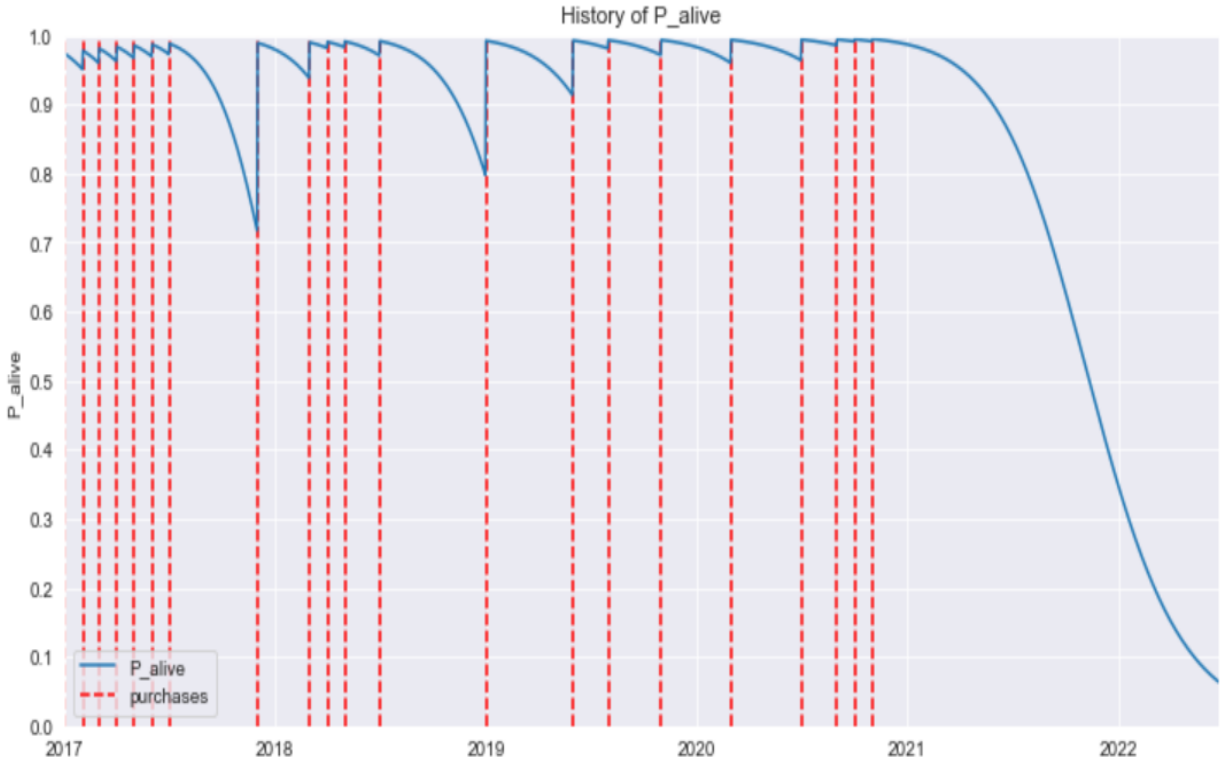
ACCOUNT_ID	frequency	recency	T	monetary_value	Πιθανοί ενεργοί μήνες για το επόμενο εξάμηνο
10100079	50.0	1520.0	1520.0	1704.913800	6.0
14902100	50.0	1520.0	1520.0	46.509000	6.0
10980110	50.0	1520.0	1520.0	53.884000	6.0
10980171	50.0	1520.0	1520.0	87.266000	6.0
14903746	50.0	1520.0	1520.0	72.894000	6.0
...	...	...	...	...	...
14901372	16.0	789.0	1520.0	75.212500	0.0
13066665	6.0	699.0	1430.0	59.000000	0.0
11319051	3.0	365.0	1339.0	8.516667	0.0
11319087	28.0	1155.0	1520.0	11.555357	0.0
14444918	10.0	822.0	1339.0	19.475000	0.0

485111 rows × 5 columns

Από το παραπάνω αποτέλεσμα φάνηκε ότι οι «καλύτεροι πελάτες», δηλαδή εκείνοι με υψηλές και τις 4 τιμές των βασικών μεγεθών, ήταν εξαιρετικά πιθανό να είναι ενεργοί και για τους 6 επόμενους μήνες, ενώ αντίστοιχα οι χειρότεροι, θα ήταν ανενεργοί. Συνεπώς για τους πελάτες που δεν είχαν ούτε ένα πιθανό ενεργό μήνα στο επόμενο εξάμηνο, ήταν ασφαλές να θεωρηθεί ότι περάσανε σε αδράνεια. Συνεπώς έτσι

δημιουργήθηκε ένας πιο αξιόπιστος τρόπος υπολογισμού του Churn μια και πλέον οριζόταν το χρονικό πλαίσιο στο οποίο αναφερόταν η εκτίμηση.

Τέλος, για την οπτικοποίηση της κυμάτωσης της πιθανότητας να είναι ζωντανό ένα account, χρησιμοποιήθηκε η συνάρτηση plot_history_alive() η οποία δεχόταν ως ορίσματα το αντικείμενο mbgf αλλά και τις χρονικές στιγμές που το συγκεκριμένο account ήταν ενεργό (στην προκειμένη περίπτωση τους μήνες). Για παράδειγμα το account με id 10107449:



Σημειώνεται ότι εσκεμμένα επιλέχθηκε χρονικό διάστημα 2000 ημερών ώστε να εξεταστεί η καμπύλη και για μελλοντικό χρόνο στον οποίο υποθετικά ο χρήστης ήταν ανενεργός. Σύμφωνα λοιπόν με το παραπάνω αποτέλεσμα, παρατηρήθηκε ότι σε βάθος ενός χρόνου πλήρους αδράνειας, η πιθανότητα ο χρήστης να είναι ζωντανός άγγιζε μόλις το 10%.

7.3.3 Υπολογισμός του CLV

Έχοντας προσδιορίσει τη σχέση που είχαν οι χρονικές μεταβλητές (recency,T) με το frequency, δηλαδή το πλήθος των επαναλαμβανόμενων ενεργών μηνών, για το καθορισμό της αξίας του πελάτη χρειάστηκε να εισαχθεί και η 4η μεταβλητή, ο συνολικός λογαριασμός ή αλλιώς το monetary_value. Το βασικό μοντέλο προσδιορισμού του CLV ήταν το ειδικά σχεδιασμένο για το σκοπό αυτό Gamma-Gamma, το οποίο είχε τις ακόλουθες 3 ιδιότητες:

- Η νομισματική αξία των συναλλαγών των χρηστών ήταν τυχαία γύρω από τη μέση αξία συναλλαγών τους.
- Η μέση τιμή συναλλαγής διέφερε μεταξύ των χρηστών, αλλά για έναν μεμονωμένο χρήστη δεν μεταβαλλόταν με την πάροδο του χρόνου.
- Οι μέσες τιμές συναλλαγών των πελατών κατανέμονταν βάση της κατανομής γάμα. (Gamma Distribution)

Για την ικανοποίηση της δεύτερης ιδιότητας, ελέγχθηκε το εάν υπήρχε συσχέτιση μεταξύ των monetary_value και frequency και συγκεκριμένα για τα accounts που ήταν ενεργά τουλάχιστον για 2 μήνες, προκειμένου να μη συμπεριληφθούν τα accounts που ενδεχομένως έκαναν μια διέλευση. Αυτό πραγματοποιήθηκε με τη μέθοδο corr() και τα αποτελέσματα που πάρθηκαν ήταν τα ακόλουθα:

	monetary_value	frequency
monetary_value	1.000000	0.089471
frequency	0.089471	1.000000

Βάσει των παραπάνω αποτελεσμάτων, συσχέτιση σχεδόν 0.09 σήμαινε ότι οι μεταβλητές ήταν πλήρως ασυσχέτιστες, συνεπώς τα δεδομένα μπορούσαν να εφαρμοστούν στον αλγόριθμο με ασφάλεια. Μέσω της συνάρτησης GammaGammaFitter() δημιουργήθηκε το αντικείμενο ggf το οποίο βάσει της μεθόδου conditional_expected_average_profit() η οποία ως όρισμα δεχόταν τα πεδία frequency και monetary_value, απέδιδε την εκτίμηση του συνολικού λογαριασμού ανά account. Αυτή η διαδικασία έγινε προκειμένου να διαπιστωθεί εάν το μοντέλο gamma-gamma απέδιδε σωστά σύμφωνα με τα δεδομένα που του δόθηκαν. Έτσι δημιουργήθηκε νέα στήλη με όνομα Μέσο εκτιμώμενο ποσό ανά account και τα αποτελέσματα που πάρθηκαν από τη συνάρτηση προσδιορισμού του εκτιμώμενου ποσού ανά account ήταν τα εξής:

ACCOUNT_ID	frequency	recency	T	monetary_value	Πιθανοί ενεργοί μήνες για το επόμενο εξάμηνο	Μέσο εκτιμώμενο ποσό ανά account
10100079	50.0	1520.0	1520.0	1704.913800	6.0	1694.965819
10100100	9.0	273.0	1520.0	14958.772222	0.0	14476.600844
10100111	50.0	1520.0	1520.0	447.966800	6.0	445.530393
10100147	50.0	1520.0	1520.0	960.491600	6.0	954.992321
10100160	50.0	1520.0	1520.0	384.504400	6.0	382.447247
...	...	...	...	...	...	...
21174848	1.0	28.0	28.0	3.900000	4.0	12.306267
21174861	1.0	28.0	28.0	44.750000	4.0	43.714878
21175010	1.0	28.0	28.0	25.500000	4.0	28.914003
21175034	1.0	28.0	28.0	120.400000	4.0	101.880397
21175060	1.0	28.0	28.0	58.650000	4.0	54.402264

466272 rows × 6 columns

Προκειμένου να παρθεί μια εκτίμηση για την ακρίβεια του παραπάνω αποτελέσματος, δημιουργήθηκε νέα στήλη με όνομα percentage difference στην οποία υπολογίστηκε η απόλυτη τιμή της ποσοστιαίας διαφοράς του μέσου εκτιμώμενου ποσού ανά ACCOUNT με το monetary_value, δηλαδή το πραγματικό συνολικό λογαριασμό. Η μέση τιμή του συνόλου των διαφορών των account προέκυψε 12.65. Έτσι, μόλις 12 τοις 100 διαφορά, σήμαινε ότι η εκτίμηση ήταν πολύ ικανοποιητική.

Τέλος, έχοντας υπολογίσει όλα τα απαραίτητα μεγέθη και λάβει πολύ ικανοποιητικές εκτιμήσεις για τα ενδιάμεσα στάδια, για τον τελικό προσδιορισμό του CLV χρησιμοποιήθηκε η μέθοδος customer_lifetime_value στο αντικείμενο ggf, η οποία ως ορίσματα δεχόταν το αντικείμενο mbgf που προέκυψε από το μοντέλο MBG/NBD, τις στήλες με τα 4 βασικά πεδία, δηλαδή τα frequency, recency, T και monetary_value, όπως και το χρονικό πλαίσιο t σε μήνες, έτσι ώστε να δοθεί συγκεκριμένη εκτίμηση. Συνεπώς για t=6 μήνες, η εκτιμώμενη αξία κάθε πελάτη για το εξάμηνο εκείνο ήταν:

ACCOUNT_ID	frequency	recency	T	monetary_value	Πιθανοί ενεργοί μήνες για το επόμενο εξάμηνο	Μέσο εκτιμώμενο ποσό ανά account	percentage_difference	CLV
15803794	20.0	609.0	609.0	23289.495000	5.0	22945.234846	1.478178	116372.729477
13797740	50.0	1520.0	1520.0	21563.639200	6.0	21435.014545	0.596489	115802.808382
10977572	50.0	1520.0	1520.0	21128.903000	6.0	21002.876349	0.596466	113468.178908
13738622	50.0	1520.0	1520.0	20935.191400	6.0	20810.322378	0.596455	112427.904805
14595597	50.0	1520.0	1520.0	20165.091400	6.0	20044.824532	0.596411	108292.297608
...	...	...	...	...	...	...	...	...
12952690	12.0	365.0	1520.0	6.930000	0.0	7.744787	11.757386	0.000016
13676295	10.0	304.0	1520.0	4.914000	0.0	5.945821	20.997576	0.000016
20909282	14.0	424.0	1520.0	6.010714	0.0	6.730873	11.981252	0.000012
10267906	15.0	455.0	1520.0	4.426667	0.0	5.130879	15.908403	0.000010
14753360	13.0	396.0	1520.0	2.713846	0.0	3.562660	31.277166	0.000007

466272 rows × 8 columns

Από το παραπάνω παρατηρήθηκε άμεσα ότι για τα account τα οποία είχαν μηδενικούς πιθανούς ενεργούς μήνες, το CLV ήταν και εκείνο μηδενικό, πράγμα που σήμαινε ότι με μια πρώτη ματιά η εκτίμηση ήταν σωστή.

Τέλος, εξετάστηκε η διαφορά μεταξύ εμπειρικού τρόπου προσδιορισμού του CLV, με το αποτέλεσμα που πάρθηκε από την εφαρμογή των αλγορίθμων. Για τον εμπειρικό προσδιορισμό του CLV δηλαδή, δημιουργήθηκε συνάρτηση με όνομα diafora η οποία για κάθε account υπολόγιζε τη ποσοστιαία διαφορά του μέσου εξαμηνιαίου (συνολικός λογαριασμός (TOTAL_AMT) δια τον αριθμό μηνών που ήταν ενεργός (frequency+1), και πολλαπλασιασμός του αποτελέσματος επί 6 ώστε να βρεθεί ο μέσος λογαριασμός ανά εξάμηνο) με το CLV. Στη συνέχεια, κλήθηκε η συνάρτηση με ορίσματα τις στήλες «TOTAL_AMT» από το revenue_by_accounts DataFrame και τις «frequency» και «CLV» από το sub_part DataFrame, δημιουργώντας νέα στήλη με όνομα diafora. Ακολούθως, μετά την απομάκρυνση των Nan στοιχείων τα οποία προέκυψαν λόγω του διαφορετικού μεγέθους των DataFrame που χρησιμοποιήθηκαν, η μέση ποσοστιαία διαφορά μεταξύ του CLV και εμπειρικού τρόπου προσδιορισμού ανήλθε σε 47.33 τοις εκατό. Η τεράστια αυτή απόκλιση οφειλόταν στο γεγονός ότι οι αλγόριθμοι προσδιορισμού του CLV έκαναν πιο σύνθετη ανάλυση, συμπεριλαμβάνοντας μάλιστα 4 απαραίτητα μεγέθη, ενώ η εμπειρική ανάλυση μονάχα 2, πράγμα που σαφέστατα μείωνε την αξιοπιστία του αποτελέσματος. Συμπεραίνοντας, η χρήση αλγορίθμων και εξειδικευμένων μοντέλων για το προσδιορισμό του CLV αποτέλεσε μια απαραίτητη διαδικασία, προκειμένου να δημιουργηθεί μια πραγματική και αξιόπιστη πρόβλεψη.

Κεφάλαιο 8
Μελλοντικές Επεκτάσεις

8.1 Εισαγωγή

Οι τρόποι και οι τεχνικές που χρησιμοποιήθηκαν για το προσδιορισμό της Κατάτμησης Πελατών, της Πρόβλεψης Μετάπτωσης σε Αδράνεια και της Αξίας του Πελάτη, αποτέλεσαν στοχευμένες διαδικασίες για την ανάπτυξη μέρους ενός Συστήματος Διαχείρισης Πελατών. Μολονότι οι διαδικασίες αυτές υλοποιήθηκαν και εφαρμόστηκαν σε προκαθορισμένα δεδομένα τα οποία είχαν συγκεκριμένες παραμέτρους και ιδιαιτερότητες, πέραν του κομματιού της ανάλυσης και της επεξεργασίας, οι κύριες διαδικασίες που χρησιμοποιήθηκαν αποτέλεσαν γενικές μεθόδους, οι οποίες μπορούσαν να εφαρμοστούν αποτελεσματικά σε κάθε σύνολο δεδομένων συνδρομητών. Αυτό είχε ταυτόχρονα και θετικό, αλλά και αρνητικό αντίκτυπο. Το θετικό ήταν ότι με μικρές τροποποιήσεις στην ανάλυση και την επεξεργασία, το σύστημα αυτό μπορούσε να εξάγει πολύ ικανοποιητικά αποτελέσματα για κάθε Dataset που αφορούσε στοιχεία συνδρομητών. Από την άλλη, δεν εμβάθυνε πλήρως στα εκάστοτε δεδομένα, πράγμα που σήμαινε ότι ενδεχομένως μπορούσαν να εξαχθούν ακόμα πιο ακριβή αποτελέσματα. Ακόμη, με την μελλοντική προσθήκη δεδομένων θα μπορούσαν να αναλυθούν θέματα εμπορικής πολιτικής.

8.2 Εξειδικευμένη Κατάτμηση Πελατών και Προσδιορισμός Αξίας Πελάτη

Οι κύριοι τρόποι προσδιορισμού του Customer’s Segmentation και CLV βασίστηκαν στο μοντέλο RFM και τους αλγορίθμους της βιβλιοθήκης lifetimes αντίστοιχα. Αν και οι μέθοδοι αυτοί έδωσαν πολύ ικανοποιητικά αποτελέσματα αποτελώντας πολλές φορές τη βασική επιλογή μιας εταιρίας για μια πρώτη εκτίμηση των μεγεθών αυτών, εντούτοις δε λάμβαναν υπόψη ιδιαιτερότητες και διαφορετικά πεδία των συνδρομητών. Η συμπερίληψη αυτών των πεδίων θα απαιτούσε νέα μοντέλα και αλγορίθμους οι οποίοι πιθανότατα θα έπρεπε να κατασκευαστούν εξολοκλήρου από τον αναλυτή, διαδικασία επίπονη αλλά και ριψοκίνδυνη μια και η αξιοπιστία της δε θα ήταν δεδομένη. Ωστόσο, μια τέτοια διαδικασία δεδομένου ότι γινόταν σωστά, θα προσέφερε πολύ καλύτερη ακρίβεια στα αποτελέσματα, μεγιστοποιώντας έτσι τα κέρδη μιας εταιρίας. Για παράδειγμα, οι τεχνικές αυτές θα μπορούσαν να επεκταθούν ώστε να λαμβάνουν κατάλληλα υπόψη πεδία όπως:

- PREAUTH_STATUS: Δηλαδή να προσμετράται το εάν τα accounts έχουν ενεργοποιημένη πάγια εντολή, δηλαδή εάν ανανεώνεται αυτόματα το υπόλοιπο του λογαριασμού του με χρέωση τραπεζικής κάρτας.

- E_INVOICE_STATUS: Δηλαδή να συμπεριλαμβάνεται το εάν το εκάστοτε account είχε επιλέξει να του στέλνονται οι ειδοποιήσεις για το υπόλοιπο του λογαριασμού του μέσω ηλεκτρονικού ταχυδρομείου.

- WEB_USER: Δηλαδή να λαμβάνεται υπόψη εάν ο εκάστοτε χρήστης είχε κάνει εγγραφή στην ηλεκτρονική υπηρεσία 'my e-Pass' στην ιστοσελίδα της Αττικής Οδού.

- CUSTOMER'S_ATTRIBUTE: Δηλαδή να συνυπολογίζεται το εάν ο χρήστης είναι ιδιώτης ή ελεύθερος επαγγελματίας, ή φυσικό ή νομικό πρόσωπο.

8.3 Ανάλυση Θεμάτων Εμπορικής Πολιτικής

Μια ακόμη μελλοντική επέκταση, η οποία ωστόσο προϋπέθετε αξιοποίηση πρόσθετων πληροφοριών οι οποίες δεν ήταν διαθέσιμες τη δεδομένη χρονική περίοδο εκπόνησης διπλωματικής, θα μπορούσε να αφορά θέματα εμπορικής πολιτικής. Συγκεκριμένα, μέσω ανάλυσης των δεδομένων αυτών θα μπορούσαν να ανιχνευτούν μοτίβα χρήσης υπηρεσιών σε συσχέτιση με χαρακτηριστικά των συνδρομητών, που ως στόχο θα είχαν τη διαμόρφωση στοχευμένων δράσεων για τον καθορισμό νέων προϊόντων. Ενδεικτικά παραδείγματα αποτελούν τα παρακάτω:

- Εύρεση αναντιστοιχιών συνδρομητικών πακέτων με συνδρομητές, δηλαδή εσφαλμένη χρήση των εκπτωτικών προγραμμάτων σε σχέση με τις πραγματοποιούμενες διαδρομές.

- Sensitivity analysis αλλαγών στα συνδρομητικά πακέτα, δηλαδή ανάλυση η οποία εξετάζει την επίδραση στη χρήση και στα έσοδα πριν και μετά την αλλαγή.

- Εξέταση συσχέτισης της εγγραφής στην υπηρεσία myEpass ή της ανάθεσης πάγια εντολής με αύξηση διελεύσεων και εσόδων.

Βιβλιογραφία

- Charu C. Aggarwal *"Data Mining"*. Springer, 2015.
- Ian H. Witten, Eibe Frank, Mark A.Hall, Christopher J. Pal *"Data Mining: Practical Machine Learning Tools and Techniques"*. Morgan Kaufmann, 2017.
- Christopher M. Bishop *"Pattern Recognition and Machine Learning"*. Springer, 2005.
- Aurelien Geron *"Hands-on machine learning with Scikit-Learn, Keras and TensorFlow: concepts, tools, and techniques to build intelligent systems"*. OReilly, 2019.
- Dave Kuhlman *"A Python Book: Beginning Python, Advanced Python, and Python Exercises"*. Platypus Global Media, 2011.
- *"Python (programming language)"* [Online]. Available: https://en.wikipedia.org/wiki/Python_(programming_language)
- *"Project Jupyter"* [Online]. Available: https://en.wikipedia.org/wiki/Project_Jupyter
- *"pandas documentation"* [Online]. Available: *https://pandas.pydata.org/docs/*
- *"NumPy Documentation"* [Online]. Available: https://numpy.org/doc/stable/
- *"scikit-learn"* [Online]. Available: https://scikit-learn.org/stable/index.html
- *"seaborn: statistical data visualization"* [Online]. Available: https://seaborn.pydata.org/
- *"Matplotlib 3.5.1 documentation"* [Online]. Available: https://matplotlib.org/stable/index.html#
- *"PostgreSQL+ time-series"* [Online]. Available: https://www.timescale.com/
- *"Διαχείριση κυκλοφορίας & συντήρησης"* [Online]. Available: https://www.aodos.gr/leitourgia-sudirisi/diaheirisi-kukloforias-sudirisis/

- *"lifetimes"* [Online]. Available: https://lifetimes.readthedocs.io/en/latest/index.html

- *"Customer relationship management"* [Online]. Available: https://en.wikipedia.org/wiki/Customer_relationship_management

- *"Customer Segmentation"* [Online]. Available: https://www.optimove.com/resources/learning-center/customer-segmentation

- Kevin Do Ruibin, Tobias Vintilescu Borglöv *"Predicting Customer Lifetime Value Understanding its accuracy and drivers from a frequent flyer program perspective"*.

- Saharon Rosset, Einat Neumann, Uri Eick, Nurit Vatnik, Yizhak Shuki Idan *"Customer lifetime value modeling and its use for customer retention planning".*

- Roudabeh Gharaee *"Modeling CLV for Financial Service Providers – Case of Karafarin Bank"*.

- Jan Johansson, Jorgen Sparredal *"CRM in e-Business"*.

- Vladislav Lazarov, Marius Capota *"Churn Prediction"*.

Παράρτημα Α
Πηγαίος Κώδικας Φόρτωσης, Επεξεργασίας και Ανάλυσης Δεδομένων

```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
get_ipython().run_line_magic('matplotlib', 'inline')
sns.set_style('darkgrid')

import os

##all revenues
path='../../Diploma/Data/all_revenues'
files = [f for f in os.listdir(path)]

dfs = []
months = []

date = pd.to_datetime('2017-01-01')

for file in files:

    months.append(date)
    if file=='revenue_data179.csv':
        df = pd.read_csv(path+'/'+file, sep=':', decimal=',',
encoding = "iso8859_7")
    else:
        df = pd.read_csv(path+'/'+file, sep=';', decimal=',',
encoding = "iso8859_7")
    if 'STATEMENT_ISSUE_ID' in df.columns:
        df.drop(labels=['STATEMENT_ISSUE_ID', 'DESCRIPTION'], axis=1,
inplace=True)
    elif 'STATEMENT_ISSUE_COUNT_ID' in df.columns:
        df.drop(labels=['STATEMENT_ISSUE_COUNT_ID', 'DESCRIPTION'],
axis=1, inplace=True)

    dfs.append(df)

    date += pd.DateOffset(months=1)

for df, month in zip(dfs, months):

    df['Month'] = month

revenue_data = pd.concat(dfs)
revenue_data.head(5)

# ομαδοποίηση μέσω πρόσθεσης των στοιχείων των έταιρων
αυτοκινητόδρομων και διαγραφή των επιμέρους στηλών

revenue_data['ELSE_CNT'] = revenue_data.iloc[:,2:9].sum(axis=1)
revenue_data['ELSE_AMT'] = revenue_data.iloc[:,10:17].sum(axis=1)
```

```
revenue_data.drop(['MOREAS_CNT','OLYMPIA_CNT','AIGAIO_CNT','GEFYRA_CN
T','NEA_CNT','KENTRIKI_CNT','EGNATIA_CNT','MOREAS_AMT','OLYMPIA_AMT',
'AIGAIO_AMT','GEFYRA_AMT','NEA_AMT','KENTRIKI_AMT','EGNATIA_AMT'],axi
s=1,inplace=True)
revenue_data.head(5)

# διαγραφή στοιχείων που ενώ δε πλήρωσαν αντίτιμο διοδίου είχαν
διέλευση

revenue_data = revenue_data[(revenue_data['AO_AMT']!=0) |
(revenue_data['AO_CNT']==0)]
revenue_data = revenue_data[(revenue_data['ELSE_AMT']!=0) |
(revenue_data['ELSE_CNT']==0)]

# δημιουργία στηλών total_amt και total_cnt

revenue_data['TOTAL_CNT'] = revenue_data['ELSE_CNT'] +
revenue_data['AO_CNT']
revenue_data['TOTAL_AMT'] = revenue_data['ELSE_AMT'] +
revenue_data['AO_AMT']

# έλεγχος για nan στοιχεία σε ολοκληρο το Dataframe

revenue_data.isnull().sum().sum()

# έλεγχος αν οι τύποι δεδομένων κάθε στήλης είναι σωστοί

revenue_data.dtypes

# συγκριση μεγέθους AO_AMT ΜΕ ELSE_AMT

revenue_data[['AO_AMT', 'ELSE_AMT']].sum(axis=0)

# συγκριση μεγέθους AO_AMT ΜΕ ELSE_AMT

revenue_data[['AO_CNT', 'ELSE_CNT']].sum(axis=0)

# δημιουργία νέου dataframe βάσει του groupby account_id στο
revenue_data με στήλες συνολικά διόδια, συνολικές διελεύσεις και το
μήνα που εμφανίστηκε πιο πρόσφατα, για κάθε account_id

revenue_data_byaccounts = pd.DataFrame(data =
revenue_data.groupby(by='ACCOUNT_ID').agg({'Month': lambda x:
x.max(), 'TOTAL_CNT': lambda x: x.sum(), 'TOTAL_AMT': lambda x:
x.sum()}))

# συσχέτιση συνολικών διελεύσεων με συνολικά έσοδα ανά account

plt.figure(figsize=(12,6))
plt.scatter(x=revenue_data_byaccounts['TOTAL_CNT'],
y=revenue_data_byaccounts['TOTAL_AMT'])

# συσχέτιση συνολικών διελεύσεων με συνολικά έσοδα ανά account (σε
λογαριθμική κλίμακα)

plt.figure(figsize=(12,6))
plt.scatter(x=np.log(revenue_data_byaccounts['TOTAL_CNT']),
y=np.log(revenue_data_byaccounts['TOTAL_AMT']))

# προσδιορισμός και αφαίρεση των κάτω ακραίων outliers
```

```
revenue_data_byaccounts =
revenue_data_byaccounts[revenue_data_byaccounts['TOTAL_AMT'] >= 1.4]

# προσδιορισμός 10 πιο ακραίων τιμών λογαριασμών

revenue_data_byaccounts['TOTAL_AMT'].sort_values(ascending=False).hea
d(10).index

revenue_data_byaccounts['TOTAL_AMT'].sort_values(ascending=True).head
(10).index

# απομάκρυνση των 10 outliers

revenue_data_byaccounts.drop([14780871, 10100067, 14352993, 12448055,
10100123, 15042429,
            14992530, 14982339, 10155957, 10100043], inplace=True)

# συσχέτιση συνολικών διελεύσεων με συνολικά έσοδα ανά account χωρις
τα 10 outliers

plt.figure(figsize=(12,6))
plt.scatter(x=revenue_data_byaccounts['TOTAL_CNT'],
y=revenue_data_byaccounts['TOTAL_AMT'])

# προσδιορισμός 4 πιο ακραίων τιμών διελεύσεων

revenue_data_byaccounts['TOTAL_CNT'].sort_values(ascending=False).hea
d(4).index

# απομάκρυνση των 4 outliers

revenue_data_byaccounts.drop([10113757, 14254594, 10160920,
10395246], inplace=True)

# συσχέτιση συνολικών διελεύσεων με συνολικά έσοδα ανά account χωρίς
τα outliers

plt.figure(figsize=(12,6))
plt.scatter(x=revenue_data_byaccounts['TOTAL_CNT'],
y=revenue_data_byaccounts['TOTAL_AMT'])

# συνάρτηση υπολογισμού των μηνών που πέρασαν από τη τελευταία φορά
που ήταν ενεργός

def months_passed(date):
    if date.year == 2017:
        return 3-date.month + 48
    elif date.year == 2018:
        return 3-date.month + 36
    elif date.year == 2019:
        return 3-date.month + 24
    elif date.year == 2020:
        return 3-date.month + 12
    else:
        return 3-date.month

# δημιουργία του αντίστοιχου πεδίου

revenue_data_byaccounts['Months_passed_since_the_most_recent_transit'
] = revenue_data_byaccounts['Month'].apply(months_passed)
```

```
# πλήθος accounts βάσει των πόσων μηνών βρίσκονται σε αδράνεια από τη
πιο πρόσφατη ημερομηνία

plt.figure(figsize=(12,6))
sns.countplot(x='Months_passed_since_the_most_recent_transit',
data=revenue_data_byaccounts)

# μέση τιμή διοδίου σε ευρώ ανά διέλευση κατά τη διάρκεια των μηνών

plt.figure(figsize=(12,6))
(revenue_data.groupby(by='Month')['TOTAL_AMT'].sum()/revenue_data.gro
upby(by='Month')['TOTAL_CNT'].sum()).plot()

# μέγιστος αριθμός διελεύσεων ανά μήνα

plt.figure(figsize=(12,6))
(revenue_data.groupby(by='Month')['TOTAL_CNT'].sum()).plot()


# μέγιστο κέρδος σε ευρώ ανά μήνα

plt.figure(figsize=(12,6))
(revenue_data.groupby(by='Month')['TOTAL_AMT'].sum()).plot()

# συσχέτιση συνολικών διελεύσεων με συνολικά έσοδα ανά μήνα

plt.figure(figsize=(12,6))
plt.scatter(x=revenue_data.groupby(by='Month')['TOTAL_AMT'].sum(),
y=revenue_data.groupby(by='Month')['TOTAL_CNT'].sum())
plt.xlabel('Εσοδα σε ευρώ')
plt.ylabel('Διελεύσεις')

## account_data

path='../../Diploma/Data/account_data.csv'
account_data=pd.read_csv(path, sep=';', decimal=',', encoding =
'iso8859_7', error_bad_lines=True , low_memory=False)
account_data.head(5)

# είναι τα accounts μοναδικά?

account_data.shape[0] == account_data['ACCOUNT_ID'].nunique()

# αλλαγή δείκτη

account_data.set_index('ACCOUNT_ID',inplace=True)
account_data.head(5)

# τροποποίηση ημερομηνίας σε datetime

account_data['CREATE_LDATETIME']=pd.to_datetime(account_data['CREATE_
LDATETIME'])

# έλεγχος ύπαρξης μηδενικών τιμών ανά πεδίο

account_data.isnull().sum()

# εύρεση των 3 προβληματικών account

account_data[account_data['ACCOUNT_PROFILE_NAME'].isnull()]
```

```
account_data['Unnamed: 14'].value_counts()

account_data['MAILING_POSTAL_CODE'].nunique()

account_data['COUNTRY_NAME'].value_counts().head(5)

account_data['PROFESSION'].nunique()

account_data['PROFESSION'].value_counts()

account_data['BILLING_TYPE'].value_counts()

account_data['PREAUTH_STATUS'].value_counts()

account_data['E_INVOICE_STATUS'].value_counts()

account_data['WEB_USER'].value_counts()

# διαγραφή περιττών στηλών-διαγραφή λαθών

account_data.drop(['ACCOUNT_STATUS','ACCOUNT_STATUS_LDATETIME','MAILI
NG_POSTAL_CODE','COUNTRY_NAME','PROFESSION','WEB_USER_ACTIVATION_DATE
','Unnamed: 14','BILLING_TYPE'],axis=1,inplace=True)
account_data.drop([14797841,16091262,16108170],inplace=True)

account_data

account_data['PROVINCE_NAME'].unique()

# εντός ή εκτός Αττικής?

def attiki(prov_name):
    if (prov_name == 'ΑΤΤΙΚΗΣ') | (prov_name == 'ΑΤΤΙΚΗΣ_') |
(prov_name == '    ΑΤΤΙΚΗΣ') | (prov_name == 'ΑΤΤΙΚΗΣ') | (prov_name
== '     ΑΤΤΙΚΗΣ') | (prov_name == '    ΑΤΤΙΚΗΣ') | (prov_name ==
'ΑΤΤΙΚΗΣ__') | (prov_name == 'ΑΤΤΙΚΗΣ ') | (prov_name ==
'__ΑΤΤΙΚΗΣ__'):
        return 'ΑΤΤΙΚΗ'
    else:
        return 'ΕΚΤΟΣ_ΑΤΤΙΚΗΣ'

account_data['PROVINCE_NAME'] =
account_data['PROVINCE_NAME'].apply(attiki)

account_data['ACCOUNT_PROFILE_NAME'].unique()

# κατηγοριοποίηση συνδρομητικών πακέτων και ξεσκαρτάρισμα των OTHER

def paketa(name):
    if (name == 'EXPRESS 1.2')  | (name == 'EXPRESS 1.1') | (name ==
'EXPRESS  PASS '):
        return 'EXPRESS ΙΔΙΩΤΗΣ'
    elif (name == 'EXPRESS 2.1'):
        return 'EXPRESS ΕΛ.ΕΠΑΓΓΕΛΜΑΤΙΑΣ'
    elif (name == 'EXPRESS 3.1'):
        return 'EXPRESS ΝΟΜΙΚΟ_ΠΡΟΣΩΠΟ'
    elif (name == 'BONUS 1.1'):
        return 'BONUS ΙΔΙΩΤΗΣ'
    elif (name == 'BONUS 3.1 '):
        return 'BONUS ΝΟΜΙΚΟ_ΠΡΟΣΩΠΟ'
    elif (name == 'BONUS 2.1'):
```

```
        return 'BONUS ΕΛ.ΕΠΑΓΓΕΛΜΑΤΙΑΣ'
    elif (name == 'FRIENDLY 3.1'):
        return 'FRIENDLY ΝΟΜΙΚΟ_ΠΡΟΣΩΠΟ'
    elif (name == 'FRIENDLY 1.1'):
        return 'FRIENDLY ΙΔΙΩΤΗΣ'
    elif (name == 'FRIENDLY 2.1'):
        return 'FRIENDLY ΕΛ.ΕΠΑΓΓΕΛΜΑΤΙΑΣ'
    elif (name == 'MOTO 1.2') | (name == 'MOTO 1.1'):
        return 'MOTO ΙΔΙΩΤΗΣ'
    elif (name == 'MOTO 3.1'):
        return 'MOTO ΝΟΜΙΚΟ_ΠΡΟΣΩΠΟ'
    elif (name == 'MOTO 2.1'):
        return 'MOTO ΕΛ.ΕΠΑΓΓΕΛΜΑΤΙΑΣ'
    elif (name == 'EXPRESSTRUCK3.1'):
        return 'EXPRESS_TRUCK ΝΟΜΙΚΟ_ΠΡΟΣΩΠΟ'
    elif (name == 'EXPRESSTRUCK2.1'):
        return 'EXPRESS_TRUCK ΕΛ.ΕΠΑΓΓΕΛΜΑΤΙΑΣ'
    elif (name == 'EXPRESSTRUCK1.2'):
        return 'EXPRESS_TRUCK ΙΔΙΩΤΗΣ'
    elif (name == 'FRIEND.TRUCK2.1'):
        return 'FRIENDLY_TRUCK ΕΛ.ΕΠΑΓΓΕΛΜΑΤΙΑΣ'
    elif (name == 'FRIEND.TRUCK3.1'):
        return 'FRIENDLY_TRUCK ΝΟΜΙΚΟ_ΠΡΟΣΩΠΟ'
    elif (name == 'FRIEND.TRUCK1.1'):
        return 'FRIENDLY_TRUCK ΙΔΙΩΤΗΣ'
    elif (name == 'Business no fee') | (name == 'BUSINESS 1') | (name
== 'BUSINESS 2') | (name == 'BUSINESS 3') | (name == 'ON HOLD
B/NESS') | (name == 'BUSINESS 4') | (name == 'BUSINESS 5'):
        return 'BUSINESS ΙΔΙΩΤΗΣ'
    else:
        return 'OTHER'

account_data['ACCOUNT_PROFILE_NAME'] =
account_data['ACCOUNT_PROFILE_NAME'].apply(paketa)
account_data = account_data[account_data['ACCOUNT_PROFILE_NAME'] !=
'OTHER']

# διαχωρισμός σε column Πακέτου και Ιδιότητας

account_data[["CUSTOMER'S_PACKAGE","CUSTOMER'S_ATTRIBUTE"]] =
account_data['ACCOUNT_PROFILE_NAME'].str.split(' ', expand=True)
account_data.drop('ACCOUNT_PROFILE_NAME',axis=1,inplace=True)

# πιο παλιό account

account_data['CREATE_LDATETIME'].min()

# πιο νέο account

account_data['CREATE_LDATETIME'].max()

# πόσο παλιός είναι ο κάθε χρήστης? => ομαδοποίηση ανά 3 χρόνια

def years(date):
    if (date.year == 2002) | (date.year == 2003) | (date.year ==
2004) | (date.year == 2005):
        return 6
    elif (date.year == 2006) | (date.year == 2007) | (date.year ==
2008):
        return 5
```

```
    elif (date.year == 2009) | (date.year == 2010) | (date.year ==
2011):
        return 4
    elif (date.year == 2012) | (date.year == 2013) | (date.year ==
2014):
        return 3
    elif (date.year == 2015) | (date.year == 2016) | (date.year ==
2017):
        return 2
    else:
        return 1

account_data['HOW_OLD'] =
account_data['CREATE_LDATETIME'].apply(years)
account_data.drop('CREATE_LDATETIME', axis=1, inplace=True)

account_data

account_data['PROVINCE_NAME'].value_counts()

# some visualization

plt.figure(figsize=(12,6))
sns.countplot(x="CUSTOMER'S_PACKAGE",
data=account_data,hue="CUSTOMER'S_ATTRIBUTE")

plt.figure(figsize=(12,6))
sns.countplot(x="HOW_OLD", data=account_data, hue='E_INVOICE_STATUS')

plt.figure(figsize=(12,6))
sns.countplot(x="HOW_OLD", data=account_data, hue='WEB_USER')
```

Παράρτημα Β
Πηγαίος Κώδικας Τμηματοποίησης Πελατών (Customer's Segmentation)

```
## CUSTOMER SEGMENTATION (RFM model)

# πιο πρόσφατη ημερομηνία διελεύσεων

Latest_Date = pd.to_datetime('2021-03-01')

# υπολογισμός του πόσο πρόσφατα περασε τελευταία φορά(Recency), του
πόσες φορές συνολικά πέρασε(Frequency) και του συνολικού ποσού που
πλήρωσε στα διόδια(Monetary)

RFMScores = revenue_data.groupby(by='ACCOUNT_ID').agg({'Month':
lambda x: (Latest_Date - x.max()).days, 'TOTAL_CNT': lambda x:
x.sum(), 'TOTAL_AMT': lambda x: x.sum()})

# μετατροπή του τύπου των στοιχείων της στήλης month σε integers

RFMScores['Month'] = RFMScores['Month'].astype(int)

# αλλαγή ονόματος των στηλών

RFMScores.rename(columns={'Month': 'Recency', 'TOTAL_CNT':
'Frequency', 'TOTAL_AMT': 'Monetary'}, inplace=True)

# απομάκρυνση των γνωστών outliers

RFMScores.drop([14780871, 10100067, 14352993, 12448055, 10100123,
15042429,
            14992530, 14982339, 10155957, 10100043, 10113757,
14254594, 10160920, 10395246], inplace=True)

RFMScores = RFMScores[RFMScores['Monetary'] >= 1.4]

# διαχωρισμός του RFMScores σε 5 τμήματα

quantiles = RFMScores.quantile(q=[0.2, 0.4, 0.6, 0.8])

# μετατροπή σε dictionary

quantiles = quantiles.to_dict()

# συναρτήσεις για απόδοση value ανά segment

def RScoring(x,p,d):
    if x <= d[p][0.2]:
        return 5
    elif x <= d[p][0.4]:
        return 4
    elif x <= d[p][0.6]:
        return 3
    elif x <= d[p][0.8]:
```

```
        return 2
    else:
        return 1

def FMScoring(x,p,d):
    if x <= d[p][0.2]:
        return 1
    elif x <= d[p][0.4]:
        return 2
    elif x <= d[p][0.6]:
        return 3
    elif x <= d[p][0.8]:
        return 4
    else:
        return 5

# δημιουργία R,F,M columns βάσει της αξίας που θεωρήσαμε στις
παραπάνω συναρτήσεις

RFMScores['R'] = RFMScores['Recency'].apply(RScoring,
args=('Recency',quantiles,))
RFMScores['F'] = RFMScores['Frequency'].apply(FMScoring,
args=('Frequency',quantiles,))
RFMScores['M'] = RFMScores['Monetary'].apply(FMScoring,
args=('Monetary',quantiles,))

# υπολογισμός συνολικού σκορ για κάθε χρήστη

RFMScores['RFM'] = RFMScores[['R','F','M']].sum(axis=1)

# απόδοση loyalty σε κάθε χρήστη => οι Diamond θα ανανεώσουν ενω οι
Bronze θα κάνουν churn out

Loyalty_Lvl = ['Diamond', 'Platinum', 'Gold', 'Silver', 'Bronze']

def Loyalty(x,l):
    if (x == 3) | (x == 4):
        return l[4]
    elif (x == 5) | (x == 6) | (x == 7):
        return l[3]
    elif (x == 8) | (x == 9) | (x == 10):
        return l[2]
    elif (x == 11) | (x == 12) | (x == 13):
        return l[1]
    else:
        return l[0]

RFMScores['RFM_Loyalty_Lvl'] = RFMScores['RFM'].apply(Loyalty,
args=(Loyalty_Lvl,))

RFMScores

# visualization των αποτελεσμάτων

plt.figure(figsize=(16,8))
sns.countplot(x='RFM', data=RFMScores, hue='RFM_Loyalty_Lvl')

plt.figure(figsize=(12,6))
sns.scatterplot(x='Monetary', y='Recency', data=RFMScores,
hue='RFM_Loyalty_Lvl', edgecolor='none', alpha=0.5)
```

```
plt.figure(figsize=(12,6))
sns.scatterplot(x='Frequency', y='Recency', data=RFMScores,
hue='RFM_Loyalty_Lvl', edgecolor='none', alpha=0.5)

plt.figure(figsize=(12,6))
sns.scatterplot(x='Frequency', y='Monetary', data=RFMScores,
hue='RFM_Loyalty_Lvl', edgecolor='none', alpha=0.5)

## CUSTOMER SEGMENTATION (K-means model)

#standardize the variables

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaled_data =
scaler.fit_transform(RFMScores[['Recency','Frequency','Monetary']])
scaled_data = pd.DataFrame(scaled_data, index=RFMScores.index,
columns=[['Recency','Frequency','Monetary']])

scaled_data

# elbow method with k-means

from sklearn.cluster import KMeans

error_rate = []
for k in range(1,20):
    km = KMeans(n_clusters=k)
    km.fit(scaled_data)
    error_rate.append(km.inertia_)

# visualization

plt.figure(figsize=(12,6))
plt.plot(range(1,20), error_rate, 'bx-')
plt.xlabel('Number of Clusters(k)')
plt.ylabel('error_rate')
plt.title('The Elbow Method showing the optimal k')

# k-means για k=5 (βλέπε παραπάνω γραφική)

km_cluster = KMeans(n_clusters=5)
km_cluster.fit(scaled_data)
RFMScores['Cluster'] = km_cluster.labels_

RFMScores

# κατανομή του πλήθους των account ανά cluster

RFMScores['Cluster'].value_counts()

plt.figure(figsize=(12,6))
sns.scatterplot(x='Frequency', y='Recency', data=RFMScores,
hue='Cluster', palette='bright')

plt.figure(figsize=(12,6))
sns.scatterplot(x='Monetary', y='Recency', data=RFMScores,
hue='Cluster', palette='bright')

plt.figure(figsize=(12,6))
```

```
sns.scatterplot(x='Frequency', y='Monetary', data=RFMScores ,
hue='Cluster', palette='bright')
```

Παράρτημα Γ
Πηγαίος Κώδικας Πρόβλεψης Μετάπτωσης Πελατών σε Αδράνεια (Churn Prediction)

```
## ΠΡΟΒΛΕΨΗ ΠΟΙΑ ACCOUNT ΘΑ ΚΑΝΟΥΝ CHURN OUT

# πρώτα πρώτα κάνουμε merge τα 2 dataframes ώστε να ενώσουμε τα
attributes του account_data με τα του revenue_data_byaccounts

accounts = pd.merge(account_data, revenue_data_byaccounts,
left_index=True, right_index=True)

# συνάρτηση για το αν ο χρήστης είναι ενεργός το τελευταίο 9μηνο

def active(x):
    if (x >= pd.to_datetime('2020-07-01')):
        return 1
    else:
        return 0

accounts['Active_last_9_months'] = accounts['Month'].apply(active)

accounts['Active_last_9_months'].value_counts()

# διαγραφή πλεονάζοντων πεδίων

accounts.drop(['Months_passed_since_the_most_recent_transit',
'Month'], axis=1, inplace=True)

# one hot encoding για να χρησιμοποιηθούν στο classification

accounts = pd.get_dummies(accounts, columns=['PROVINCE_NAME',
'PREAUTH_STATUS', 'E_INVOICE_STATUS', 'WEB_USER',
"CUSTOMER'S_PACKAGE", "CUSTOMER'S_ATTRIBUTE"], drop_first=True)

# standardize ορισμένα columns

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
accounts[['HOW_OLD', 'TOTAL_CNT', 'TOTAL_AMT']] =
scaler.fit_transform(accounts[['HOW_OLD', 'TOTAL_CNT', 'TOTAL_AMT']])

accounts

# διαχωρισμός σε feature variable X και target variable y

X = accounts.drop(['Active_last_9_months'], axis=1)
y = accounts['Active_last_9_months']

# διαχωρισμός σε training set και test set

from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=101)

# Classification models

from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import LinearSVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics

# Logistic Regression

#fit the model

logmodel = LogisticRegression(max_iter=200, class_weight='balanced')
logmodel.fit(X_train, y_train)

# predict for new data

logmodel_pred = logmodel.predict(X_test)

# evaluate the model

print(metrics.classification_report(y_test, logmodel_pred))

# Support Vector Machine

#fit the model

svcmodel = LinearSVC(class_weight='balanced')
svcmodel.fit(X_train, y_train)

# predict for new data

svcmodel_pred = svcmodel.predict(X_test)

# evaluate the model

print(metrics.classification_report(y_test, svcmodel_pred))

# K Neighbors Classifier

# εύρεση optimal k με elbow method

error_rate = []

for i in range(1,20):
    knnmodel = KNeighborsClassifier(n_neighbors=i, n_jobs=-1)
    knnmodel.fit(X_train, y_train)
    knnmodel_pred = knnmodel.predict(X_test)
    error_rate.append(np.mean(knnmodel_pred != y_test))

# visualization of elbow method

plt.figure(figsize=(12,6))
plt.plot(range(1,20), error_rate, 'bx-')
plt.xlabel('Number of Neighbors(k)')
plt.ylabel('error_rate')
plt.title('The Elbow Method showing the optimal k')
```

```
# fit the model for k=9

knnmodel = KNeighborsClassifier(n_neighbors=9, n_jobs=-1)
knnmodel.fit(X_train, y_train)

# predict for new data
knnmodel_pred = knnmodel.predict(X_test)

# evaluate the model
print(metrics.classification_report(y_test, knnmodel_pred))

# Decision Tree Classification

# fit the model

dtmodel = DecisionTreeClassifier(class_weight='balanced')
dtmodel.fit(X_train, y_train)

# predict for new data

dtmodel_pred = dtmodel.predict(X_test)

# evaluate the model

print(metrics.classification_report(y_test, dtmodel_pred))

# Random Forest Classification

# fit the model

rfmodel = RandomForestClassifier(n_jobs=-1, class_weight='balanced')
rfmodel.fit(X_train, y_train)

# predict for new data

rfmodel_pred = rfmodel.predict(X_test)

# evaluate the model

print(metrics.classification_report(y_test, rfmodel_pred))

# πρόβλεψη πιθανότητας να κάνει churn out (για κάθε row προκύπτουν 2
τιμές, εμείς κρατάμε τη πιθανότητα να είναι 0, δηλαδή inactive, άρα
να κανει Churn Out)

# εφόσον k-neighbors αναδείχθηκε καλύτερος αλγόριθμος, χρησιμοποιούμε
αυτόν

accounts['Probability_of_Churn'] =
round(pd.Series(knnmodel.predict_proba(X)[:,0], index=X.index)*100,
2)
accounts['Probability_of_Churn']
```

Παράρτημα Δ
Πηγαίος Κώδικας Προσδιορισμού της Αξίας των Πελατών (Customer's Lifetime Value)

```
## Customer's Lifetime Value (CLV) calculation

#Εδώ frequency είναι το πόσες επαναλβανόμενες φορές πέρασε (ανά μήνα)
(εφοσον έχουμε σπαν 51 μην΄ών θα είναι max=50)
#Εδώ recency είναι η χρονική διαφορά της ημερομηνίας πιο πρόσφατης
διέλευσης με την ημερομηνία πρώτης διέλευσης(σε μέρες)
#Εδώ T είναι η χρονική διαφορά της latest_Date (μέχρι εκεί που έχουμε
data), με την ημερομηνία πρώτης διέλευσης(σε μέρες)
#Εδώ monetary_value είναι το μέσο ποσό για κάθε account για όλο το
διάστημα μελέτης

#Η βιβλιοθήκη lifetimes προσφέρει έτοιμες συναρτήσεις για παραγωγή
των ανωτέρω 4 μεγεθών, τα οποία και θα χρησιμοποιήσουμε:

from lifetimes.utils import summary_data_from_transaction_data

summary = summary_data_from_transaction_data(revenue_data,
'ACCOUNT_ID', 'Month', monetary_value_col='TOTAL_AMT',
observation_period_end=Latest_Date)
summary

# απομάκρυνση των γνωστών outliers

summary.drop([14780871, 10100067, 14352993, 12448055, 10100123,
15042429,
              14992530, 14982339, 10155957, 10100043, 10113757,
14254594, 10160920, 10395246], inplace=True)

# some visualization

plt.figure(figsize=(12,6))
sns.histplot(x='frequency', data=summary)
print('\nΟι πελάτες που χρησιμοποίησαν epass για ένα μόνο μήνα:
{}\n'.format((summary['frequency'] == 0).sum()))
print('Ο μέσος αριθμός μηνών για τον οποίο είχαμε ενεργούς πελάτες:
{}\n'.format(round(summary['frequency'].mean())))

plt.figure(figsize=(12,6))
sns.scatterplot(x='frequency', y='monetary_value', data=summary)

plt.figure(figsize=(12,6))
sns.scatterplot(x='recency', y='monetary_value', data=summary)

plt.figure(figsize=(12,6))
sns.scatterplot(x='T', y='monetary_value', data=summary)

plt.figure(figsize=(12,6))
sns.histplot(x='recency', data=summary)

plt.figure(figsize=(12,6))
```

```
sns.histplot(x='T', data=summary)

# BetaGeoFitter

from lifetimes import BetaGeoFitter

bgf = BetaGeoFitter()
bgf.fit(summary['frequency'], summary['recency'], summary['T'])
bgf.summary

# αξιολόγηση του μοντέλου μας

from lifetimes.plotting import plot_period_transactions

plt.figure(figsize=(12,6))
plot_period_transactions(bgf)

# Frequency/Recency Analysis Using MBG/NBD Model (επιλέχθηκε αυτό
διότι το κανονικό μοντέλο στην αξιολόγηση δε συνέκλινε)

from lifetimes import ModifiedBetaGeoFitter

mbgf = ModifiedBetaGeoFitter(penalizer_coef=0.0001)
mbgf.fit(summary['frequency'], summary['recency'], summary['T'])
mbgf.summary

# αξιολόγηση του μοντέλου μας

from lifetimes.plotting import plot_period_transactions

plt.figure(figsize=(16,8))
plot_period_transactions(mbgf)

# visualization of Frequency/Recency matrix

from lifetimes.plotting import plot_frequency_recency_matrix

plt.figure(figsize=(12,6))
plot_frequency_recency_matrix(mbgf, T=180, title='Πιθανός αριθμός
μηνών στους οποίους ο πελάτης είναι ενεργός για το επόμενο
εξάμηνο\n')

# πρόβλεψη εάν ο πελάτης είναι ενεργός (ζωντανός), όπου ενεργός είναι
πελάτης με τουλάχιστον μια διέλευση ανά μήνα (Survival Analysis)

from lifetimes.plotting import plot_probability_alive_matrix

plt.figure(figsize=(16,8))
plot_probability_alive_matrix(mbgf)

# κατηγοριοποίηση των account από καλύτερο πελάτη σε χειρότερο

t = 180 # πρόβλεψη για το επόμενο εξάμηνο
summary['Πιθανοί ενεργοί μήνες για το επόμενο εξάμηνο'] =
round(mbgf.conditional_expected_number_of_purchases_up_to_time(t,
summary['frequency'], summary['recency'], summary['T']))
summary.sort_values(by='Πιθανοί ενεργοί μήνες για το επόμενο
εξάμηνο', ascending=False)

# πιθανότητα για μεμονωμένο τυχαίο account (για σπαν 2000 ημερών από
τη μέρα πρώτου ενεργού μήνα)
```

```
from lifetimes.plotting import plot_history_alive


account = revenue_data[revenue_data['ACCOUNT_ID'] == 10107449]
plt.figure(figsize=(12,6))
plot_history_alive(mbgf, 2000, account, 'Month')

# κρατάμε το κομμάτι πελατών που έχουν τουλάχιστον 2 μήνες
ενεργούς(δηλαδή frequency>0)

sub_part = summary[summary['frequency'] > 0]

# έλεγχος για το αν υπάρχει συσχέτιση μεταξύ frequency και
monetary_value προκειμένου να χρησιμοποιηθεί το μοντέλο Gamma-Gamma

sub_part[['monetary_value', 'frequency']].corr()

# εφόσον δεν υπάρχει συσχέτιση, μπορούμε με ασφάλεια να
χρησιμοποιήσουμε το Gamma-Gamma

from lifetimes import GammaGammaFitter

ggf = GammaGammaFitter(penalizer_coef=0.0001)
ggf.fit(sub_part['frequency'], sub_part['monetary_value'])

# υπολογίζουμε τώρα το μέσο εκτιμώμενο ποσό ανά πελάτη (για το σύνολο
πλέον)

sub_part['Μέσο εκτιμώμενο ποσό ανά account'] =
ggf.conditional_expected_average_profit(sub_part['frequency'],
sub_part['monetary_value'])
sub_part

# ποσοστιαία διαφορά των στοιχείων των στηλών monetary_value και
μέσο_εκτιμ´´ωμενο ποσό ανα account

sub_part['percentage_difference'] =
np.absolute((sub_part['monetary_value']-sub_part['Μέσο εκτιμώμενο
ποσό ανά account'])/sub_part['monetary_value'])*100

# η μέση ποσοστιαία διαφορά

sub_part['percentage_difference'].mean()

# Υπολογισμός του CLV για κάθε account

mbgf = ModifiedBetaGeoFitter(penalizer_coef=0.0001)
mbgf.fit(sub_part['frequency'], sub_part['recency'], sub_part['T'])
sub_part['CLV'] = ggf.customer_lifetime_value(mbgf,
sub_part['frequency'], sub_part['recency'], sub_part['T'],
sub_part['monetary_value'], time=6)
sub_part.sort_values(by='CLV', ascending=False)

# συνάρτηση εύρεσης ποσοστιαίας διαφοράς μεταξύ μέσου εξαμηνιαίου
λογαριασμού με CLV

def diafora(a,b,c):
    temp = (6*a)/(b+1)
    return np.absolute(((temp-c)/temp)*100)
```

```
sub_part['diafora'] = diafora(revenue_data_byaccounts['TOTAL_AMT'],
sub_part['frequency'], sub_part['CLV'])

# απομάκρυνση των Nan μοντέλων

sub_part.dropna(inplace=True)

# η μέση ποσοστιαία διαφορά

sub_part['diafora'].mean()
```

Παράρτημα Ε
Πηγαίος Κώδικας Φόρτωσης Στοιχείων από τη Βάση Δεδομένων

```
from sqlalchemy import create_engine
import pandas as pd

USERNAME = ""
PASSWORD = ""
HOST = "attica-db.dslab.os.grnetcloud.net:55234"
DB = "attiki"

sql = create_engine(
    "postgresql+psycopg2://{}:{}@{}/{}".format(USERNAME, PASSWORD,
HOST, DB)
)
sql.connect()

df = pd.read_sql(
    """
    SELECT * FROM statements WHERE account_id = %(account)s
    ORDER BY statement_id ASC
    """,
    sql,
    params={"account": "24ec64f41368ad63e9c0"},
)

## Result is a DataFrame containing data returned by SQL, use at will

print(df.to_string())
```

Παράρτημα Στ’
Πηγαίος Κώδικας SQL για τη Δημιουργία του TABLE

```
CREATE TABLE statements (
    id SERIAL,

    account_id VARCHAR(20) NOT NULL,
    statement_id INT NOT NULL,
    statement_date DATE NOT NULL,

    ao_cnt INT,
    moreas_cnt INT,
    olympia_cnt INT,
    aigaio_cnt INT,
    gefyra_cnt INT,

    nea_cnt INT,
    kentriki_cnt INT,
    egnatia_cnt INT,

    ao_amt FLOAT,
    moreas_amt FLOAT,
    olympia_amt FLOAT,
    aigaio_amt FLOAT,
    gefyra_amt FLOAT,

    nea_amt FLOAT,
    kentriki_amt FLOAT,
    egnatia_amt FLOAT
);
```