ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΣΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

Διπλωματική Εργασία

«ΠΡΟΒΛΕΨΗ ΕΓΚΡΙΣΕΩΝ ΠΙΣΤΩΤΙΚΩΝ ΚΑΡΤΩΝ»

του

ΘΩΜΑ ΠΕΛΤΕΚΗ

Υποβλήθηκε ως απαιτούμενο για την απόκτηση του
ΜεταπτυχιακούΔιπλώματος Ειδίκευσης στα
Πληροφοριακά Συστήματα

Επιβλέπων Καθηγητής
Κωνσταντίνος Ταραμπάνης

Μάρτιος 2024

Περίληψη

Στην παρούσα διπλωματική εργασία θα μελετηθεί με τη βοήθεια της μηχανικής μάθησης η έγκριση ή όχι, της αίτησης ενός πελάτη για την έκδοση πιστωτικής κάρτας.

Τα τελευταία χρόνια έχει παρατηρηθεί το φαινόμενο της αύξησης της χρήσης των πιστωτικών καρτών όχι μόνο για την αγορά καταναλωτικών αγαθών, αλλά και για την πληρωμή φόρων εισοδήματος, του ΕΝΦΙΑ και των τελών κυκλοφορίας των οχημάτων.

Αυτή η αυξητική τάση έκδοσης και χρήσης πιστωτικών καρτών, οδήγησε τις τράπεζες στο να υιοθετήσουν ένα αυτοματοποιημένο σύστημα αποδοχής, ανάλυσης και επεξεργασίας των αιτήσεων των πελατών, για την έκδοση των πιστωτικών καρτών, το οποίο με τη συμβολή της τεχνολογίας της στατιστικής ανάλυσης και της επιστήμης της ανάλυσης των δεδομένων η εκδότρια της κάρτας τράπεζα, θα επιτυγχάνει αυτόματα την έγκριση ή όχι της αίτησης για την έκδοση πιστωτικής κάρτας.

Έτσι στο πλαίσιο της διπλωματικής εργασίας, λήφθηκε ένα σύνολο δεδομένων από την πλατφόρμα Kaggle, όπου μετά από ανάλυση αυτών μέσω της εφαρμογής Tableau, δημιουργήθηκαν χρήσιμες απεικονίσεις των δεδομένων για την καλύτερη κατανόησή τους.

Στη φάση της προεργασίας και έπειτα από σχετική επεξεργασία το σύνολο των δεδομένων έπρεπε να μετατραπεί σε μορφή κατανοητή από τους σχετικούς αλγόριθμους της μηχανικής μάθησης.

Έτσι, για τα νέα δεδομένα που σχηματίστηκαν αξιοποιήθηκαν οι αλγόριθμοι decision trees και random forest. Για το μοντέλο που δημιουργήθηκε, αξιοποιήθηκε η ενισχυτική διαβάθμιση δέντρων αποφάσεων (GBDT) με τη χρήση της βιβλιοθήκης XGBoost (Extreme Gradient Boosting). Η διαχείριση των πινάκων έγινε με συναρτήσεις της βιβλιοθήκης NumPy η οποία διαθέτει μια μεγάλη συλλογή μαθηματικών συναρτήσεων. Για τη δημιουργία γραφημάτων, χρησιμοποιήθηκε η συλλογή συναρτήσεων pyplot της βιβλιοθήκης matplotlib. Το μοντέλο αξιολογήθηκε με τις μετρικές f1-score, accuracy, precision, recall, το ποσοστό του AUC ROC score (Area Under the ROC (Receiver Operating Characteristic) Curve), αλλά και μέσω των SHAP values όπου ανάλογα με την σπουδαιότητα των γνωρισμάτων και των τιμών αυτών, προέβλεπαν αν έχουν θετική ή αρνητική επίδραση στο τελικό αποτέλεσμα του στόχου του μοντέλου μας. Η μηχανική μάθηση, αξιοποιώντας σωστά τα ακριβή και μεγάλα σύνολα δεδομένων, μπορεί να συμβάλλει σημαντικά στη πρόγνωση των τομέων της οικονομίας.

Abstract

In this thesis, the approval or not of a customer's application for the issuance of a credit card will be studied with the help of machine learning.

In recent years, the phenomenon of increasing the use of credit cards has been observed not only for the purchase of consumer goods, but also for the payment of income taxes, ENFIA and vehicle registration fees.

This increasing trend and use of credit cards, led banks to adopt an automated system of acceptance, analysis and processing of customer requests, for the issuance of credit cards, which with the contribution of the technology of statistical analysis and its science. analysis of the data the issuance of the bank card, will carry out the approval or not of the application for the issuance of a credit card.

Thus, in the context of the diploma work, a data set was obtained from the Kaggle platform, where after analyzing them through Tableau, useful visualizations of the data were created for a better understanding.

In the pre-processing phase and after relevant processing of the data set, it had to be converted into a format understandable by the relevant machine learning algorithms.

Thus, decision trees and random forest algorithms were used for the new data that was formed. For the generated model, gradient boosting decision trees (GBDT) was leveraged using the XGBoost (Extreme Gradient Boosting) library. The tables were managed with functions from the NumPy library which has a large collection of mathematical functions. To generate graphs, the pyplot function collection of the matplotlib library was used. The model was evaluated with the metrics f1-score, accuracy, precision, recall and with the percentage of the AUC ROC score (Area Under the ROC (Receiver Operating Characteristic) Curve), but also through the SHAP values where depending on the importance of the features and these values, they predicted whether they have a positive or negative effect on the final result of our model's goal. Machine learning, making good use of accurate and large data sets, can significantly contribute to the forecasting of sectors of the economy.

Ευχαριστίες

Θα ήθελα να εκφράσω τις θερμές ευχαριστίες μου στον επιβλέποντα καθηγητή μου κ. Κωνσταντίνο Ταραμπάνη που μου παρείχε την πολύτιμη καθοδήγηση και υποστήριξη, καθ' όλη τη διάρκεια της διπλωματικής μου εργασίας.

Ένα μεγάλο ευχαριστώ στη σύζυγο μου Ζωή που εδώ και 20 χρόνια είναι στήριγμα δίπλα μου, σε ότι και αν κάνω.

Επίσης στα παιδιά μου Γεωργία, Δημήτρη και Μιχαήλ που με τον τρόπο τους συνέβαλαν στην εκπλήρωση του στόχου μου.

Πίνακας περιεχομένων

Λίστα πινάκων

Λίστα Σχημάτων

Λίστα εικόνων

1.Εισαγωγή

Οι συνεχείς αυξανόμενες τάσεις των ατόμων για υπερβάλλουσα κατανάλωση και ειδικότερα για την αγορά όχι τόσο προϊόντων πρώτης ανάγκης αλλά κυρίως προϊόντων πολυτελείας, οδήγησε τις τράπεζες στο να δημιουργήσουν τραπεζικά προϊόντα, με τα οποία θα ικανοποιούνταν οι ανάγκες των ατόμων. Στην αρχή, δημιουργήθηκαν τραπεζικά προϊόντα όπως τα δάνεια (καταναλωτικά), τα οποία ναι μεν ικανοποιούσαν τις ανάγκες των ατόμων, αλλά η διαδικασία από την αίτηση του έως και τη χορήγηση του ήταν χρονοβόρα και πολύπλοκη. Με την πάροδο του χρόνου, οι τράπεζες δημιούργησαν προϊόντα που επείχαν θέση δανείου αλλά με πιο πρακτικό τρόπο. Έτσι, εκδόθηκαν οι πρώτες πιστωτικές κάρτες, οι οποίες χορηγούνταν σε εύπορα άτομα, ή άτομα με υψηλό εισόδημα, ώστε να εξασφαλίζεται πέραν της πίστωσης του ποσού και η ασφαλής αποπληρωμή του. Βέβαια, ο ανταγωνισμός μεταξύ των τραπεζών οδήγησε στην έκδοση πιστωτικών καρτών με ανταγωνιστικά επιτόκια μεταξύ τους, ώστε ο πελάτης να έχει δικαίωμα επιλογής του προϊόντος μέσα από μία ποικίλα γκάμα καρτών.

Επίσης, η αύξηση της εγκληματικότητας και ειδικότερα των κλοπών και ληστειών ανά την εποχή, οι οποίες σχετίζονταν με την αφαίρεση χρημάτων ή και πολύτιμων αντικειμένων, ήταν ένας επιπλέον λόγος που οι τράπεζες έπρεπε να δώσουν τη δυνατότητα οι πελάτες τους να μπορούν να καταναλώνουν – αγοράζουν, χωρίς να απαιτείται η κατοχή χαρτονομισμάτων μαζί τους, αλλά απλά να έχουν μαζί τους την πιστωτική τους κάρτα, η οποία σε περίπτωση κλοπής ή απώλειας η εκδότρια τράπεζα κατόπιν ανακοίνωσης της από τον πελάτη, ενεργοποιεί μία αυτοματοποιημένη διαδικασία ακύρωσης της πιστωτικής κάρτας και δέσμευσης οιαδήποτε συναλλαγής που πραγματοποιήθηκε με αυτή και δεν έγινε από τον κάτοχό της.

Επειδή, η αύξηση των αγορών μέσω των πιστωτικών καρτών, ολοένα και συνέχιζε, οι τράπεζες προκειμένου να διευρύνουν το πελατολόγιο τους και να αυξήσουν τα κέρδη τους, επέτρεψαν στους καταναλωτές με μεσαίο εισόδημα να έχουν δικαίωμα αίτησης για χορήγηση πιστωτικής κάρτας, ανάλογα με την οικογενειακή τους κατάστασή, το ετήσιο εισόδημα τους και των περιουσιακών στοιχείων που μπορεί να έχουν στην κατοχή τους.

1.1 Το πρόβλημα

Οι τράπεζες έπρεπε στην αρχή να αντιμετωπίσουν το μεγάλο όγκων δεδομένων που προέκυπτε από το μεγάλο αριθμό των αιτήσεων των πελατών, κάνοντας χρονοβόρα την όλη διαδικασία, με αποτέλεσμα τη δυσαρέσκεια από την μεριά των πελατών. Ο ανταγωνισμός μεταξύ των τραπεζών, ανάγκασε αυτές να βελτιώσουν τη διαδικασία τόσο της αίτησης όσο και της αξιολόγησης της αίτησης από τη μεριά της τράπεζας.

Μετά από την όλη βελτίωση των διαδικασιών και το μηδενισμό του χρόνου αναμονής από τη μεριά του πελάτη, έκανε την εμφάνιση του ένα άλλο πρόβλημα, το οποίο άρχισε να μεγεθύνεται. Μεγάλος αριθμός πελατών ενώ δανειζόταν για τις αγορές τους μέσω των πιστωτικών καρτών, αυξάνοντας με γεωμετρική πρόοδο το πιστωτικό υπόλοιπο, ενώ στο τέλος αδυνατούσαν να αποπληρώσουν το χρέος τους. Παρά το κέρδος των τραπεζών από τα επιτόκια του χρέους, οδηγήθηκαν στο συμπέρασμα ότι πρέπει η αξιολόγηση της αίτησης αλλά και η έκδοση των πιστωτικών καρτών, να γίνεται με φειδώ και με αυστηρότερα κριτήρια, καθόσον στο τέλος δεν λάμβαναν ούτε τους τόκους του χρέους και εμφάνιζαν ζημία σε αυτόν τον κλάδο.

1.2 Στόχος

Οι τράπεζες καθημερινά δέχονται εκατοντάδες αιτήσεις, διά ζώσης ή και ηλεκτρονικά πλέον, για τη χορήγηση πιστωτικής κάρτας. Αποτέλεσμα της αίτησης μετά από την αξιολόγηση της είναι η έγκριση ή η απόρριψη. Γνώμονας της αξιολόγησης, είναι *το κριτήριο* του ετησίου εισοδήματος του ατόμου που αιτείται τη χορήγηση, η οικογενειακή του κατάσταση, τα περιουσιακά του στοιχεία και η ύπαρξη προηγούμενων χρεών ή καθυστέρηση στην πληρωμή κάποιας προγραμματισμένης δόσης ή χρόνος αποπληρωμής του χρέους γενικότερα.

Στόχος της παρούσας διπλωματικής εργασίας είναι η δημιουργία ενός μοντέλου μηχανικής μάθησης, το οποίο θα επεξεργάζεται και θα αξιοποιεί τα δεδομένα των αιτήσεων στις τράπεζες. Το εν λόγω μοντέλο, θα προβλέπει την έγκριση ή όχι της αίτησης του πελάτη για την έκδοση – χορήγηση πιστωτικής κάρτας από το τραπεζικό ίδρυμα.

1.3 Περιεχόμενο μελέτης

Στο κεφάλαιο 2, γνωρίζουμε τις απαιτούμενες γνώσεις αρχικά για τις πιστωτικές κάρτες, όπως είναι τα χαρακτηριστικά της, το έντυπο – υπόδειγμα αίτησης για τη χορήγηση της πιστωτικής κάρτας, αλλά και για τη γλώσσα Python, την εφαρμογή Tableau και τη μηχανική μάθηση ενώ γίνεται και μία βιβλιογραφική επισκόπηση με σκοπό την πληροφόρηση για αντίστοιχες προσπάθειες που έχουν καταγραφεί, τη μέθοδο που χρησιμοποίησαν για την αντιμετώπισή του και το αποτέλεσμα που είχαν.

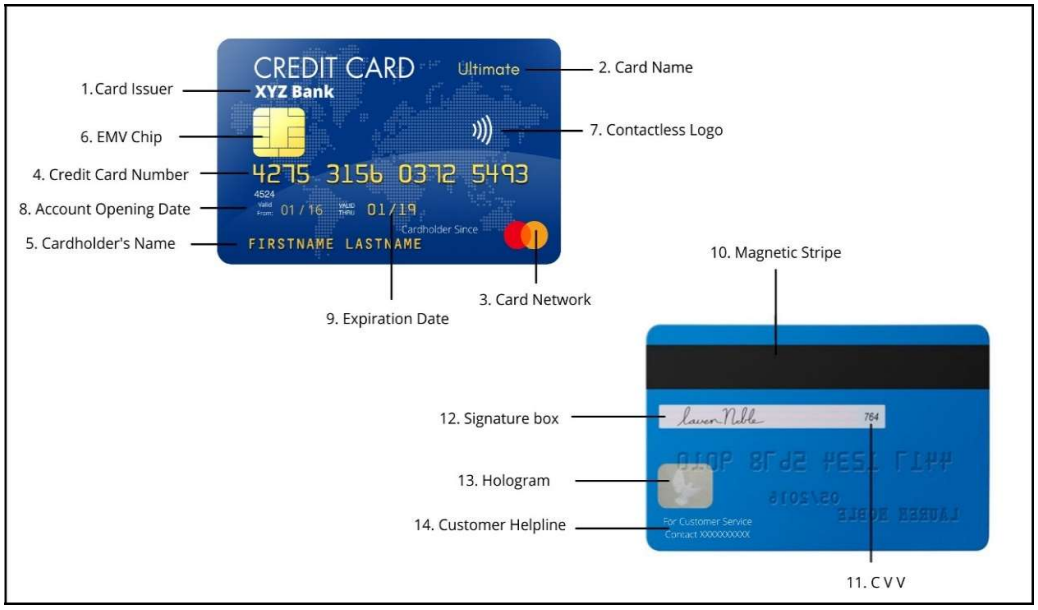
Στο κεφάλαιο 3, αναφέρεται η μεθοδολογία που θα ακολουθηθεί στην παρούσα εργασία.

Στο κεφάλαιο 4, ξεκινάει η υλοποίηση της μεθοδολογίας συλλογή και την εξερεύνηση των δεδομένων. Ακολουθεί η επεξεργασία τους, ώστε να είναι έτοιμα να εισαχθούν στον αλγόριθμο της μηχανικής μάθησης, Για αυτό μετά την επεξεργασία τους, προβαίνουμε στη δημιουργία μοντέλων – υλοποίηση της μηχανικής μάθησης (Machine Learning), όπου αρχικά γίνεται ο συντονισμός υπερπαραμέτρων με την Αναζήτηση Πλέγματος (Grid Search) και η δημιουργία των μοντέλων Decision Trees Classifier, Random Forest Classifier και XGBoost. Ακολουθεί ο πίνακας σύγχυσης (Confusion matrix) και οι οπτικοποιήσεις των καμπυλών ROC και AUC. Τέλος, δημιουργούνται τα 3 διαγράμματα σπουδαιότητας χαρακτηριστικών και η Εφαρμογή SHapley Additive exPlanations (SHAP) στο μοντέλο XGBoost .

Στο κεφάλαιο 5, αναφέρονται τα συμπεράσματα.

2. Θεωρητικό υπόβαθρο

2.1 Χαρακτηριστικά της πιστωτικής κάρτας



Εικόνα 1 Χαρακτηριστικά της πιστωτικής κάρτας

Μπροστινή όψη πιστωτικής κάρτας (Card Insider, 2021)

1. Όνομα εκδότη της κάρτας

Το πρώτο πράγμα στην επάνω γωνία (μπορεί να είναι δεξιά ή αριστερά και τα δύο) στην μπροστινή πλευρά της πιστωτικής σας κάρτας είναι το όνομα του εκδότη της κάρτας. Είναι το όνομα της τράπεζας στην οποία γίνεται η αίτηση για πιστωτική κάρτα. Στην εικόνα που φαίνεται εδώ, ότι η XYZ Bank είναι ο εκδότης της κάρτας.

2. Όνομα της πιστωτικής κάρτας

Σε κάθε πιστωτική κάρτα που εκδίδεται από τράπεζα δίνεται διαφορετικό όνομα. Το πλήρες όνομα μιας πιστωτικής κάρτας αποτελείται από το όνομα του εκδότη της κάρτας και το όνομα της συγκεκριμένης κάρτας. Στο παραπάνω παράδειγμα, το όνομα της συγκεκριμένης κάρτας είναι Ultimate και εκδίδεται από την XYZ Bank. Έτσι, το πλήρες όνομά του θα ήταν XYZ Bank Ultimate Credit Card.

3. Δίκτυο καρτών

Τα Δίκτυα Πιστωτικών Καρτών είναι τα χρηματοπιστωτικά ιδρύματα που εργάζονται στα παρασκήνια για να κάνουν όλες τις πληρωμές με πιστωτική κάρτα δυνατές. Τα δίκτυα πιστωτικών καρτών αποφασίζουν πού θα γίνει αποδεκτή η πιστωτική σας κάρτα και πού όχι. Λειτουργούν ως γέφυρες μεταξύ της εταιρείας έκδοσης καρτών και των καταστημάτων όπου γίνονται δεκτές οι πιστωτικές κάρτες. Ακολουθούν τα 5 κύρια

δίκτυα πιστωτικών καρτών στην Ινδία:

- Visa
- MasterCard
- American Express
- Diners Club
- RuPay

4. Αριθμός πιστωτικής κάρτας

Είναι ένας μοναδικός 16ψήφιος αριθμός (15ψήφιος για κάρτες American Express) που βοηθά στην αναγνώριση ενός συγκεκριμένου εκδότη κάρτας και του δικτύου καρτών. Κάθε φορά που χρησιμοποιείτε την πιστωτική σας κάρτα για να κάνετε συναλλαγές, ζητείται ο αριθμός της πιστωτικής κάρτας για να λάβετε πληροφορίες σχετικά με τον εκδότη της κάρτας σας, το δίκτυο της κάρτας σας και άλλες λεπτομέρειες. Τα πρώτα 6 ψηφία ενός αριθμού πιστωτικής κάρτας είναι γνωστά ως Αριθμός Αναγνώρισης Τράπεζας (BIN). Ονομάζεται επίσης αριθμός αναγνώρισης εκδότη (IIN) καθώς παρέχει πληροφορίες για τον εκδότη της κάρτας. Στην παραπάνω περίπτωση, το 4275 31 είναι το BIN.

5. Όνομα κατόχου κάρτας

Είναι το όνομα του κατόχου της κάρτας, δηλαδή στον οποίο ανήκει η πιστωτική κάρτα.

6. Τσιπ EMV

Το EMV, που σημαίνει EuroPay, Mastercard, Visa, αποτελεί μέρος του σύγχρονου τρόπου πληρωμής. Είναι ένα τσιπ ενσωματωμένο στην πιστωτική κάρτα και περιέχει πληροφορίες για τον κάτοχο της πιστωτικής κάρτας. Σε αυτήν την περίπτωση, το τσιπ κάτω από το όνομα της πιστωτικής κάρτας είναι το τσιπ EMV. Λόγω της τεχνολογίας τσιπ EMV, οι πιστωτικές κάρτες έχουν γίνει ο πιο ασφαλής τρόπος πληρωμής.

Κάθε φορά που σύρετε την κάρτα σας για να πραγματοποιήσετε μια πληρωμή ή τη χρησιμοποιείτε για ανάληψη μετρητών, το μηχάνημα ολίσθησης/ATM λαμβάνει τα στοιχεία του λογαριασμού σας μέσω του τσιπ και, στη συνέχεια, πρέπει να εισαγάγετε το PIN σας για επαλήθευση και ως εκ τούτου να κάνετε τη συναλλαγή σας με επιτυχία.

7. Τεχνολογία ανέπαφων πληρωμών

Το σύμβολο που μοιάζει με το σήμα WiFi στην πιστωτική σας κάρτα είναι ένα τσιπ που επιτρέπει την ανέπαφη πληρωμή στην πιστωτική σας κάρτα. Σημαίνει ότι δεν χρειάζεται πάντα να σύρετε μια κάρτα για να πραγματοποιήσετε μια συναλλαγή. Το τσιπ εκπέμπει ραδιοκύματα που καθιστούν δυνατή τη συναλλαγή χωρίς φυσική επαφή. Απλώς πρέπει να ακουμπήσετε την πιστωτική σας κάρτα και η συναλλαγή θα ολοκληρωθεί, με την προϋπόθεση ότι η συσκευή ανάγνωσης καρτών είναι επίσης ενεργοποιημένη για ανέπαφες πληρωμές.

8. Ημερομηνία ανοίγματος λογαριασμού

Η ημερομηνία ανοίγματος λογαριασμού μιας πιστωτικής κάρτας είναι η ημερομηνία έκδοσης της κάρτας σε εσάς. Είναι γραμμένο σε μορφή MM/YY. Στο παραπάνω παράδειγμα, η ημερομηνία ανοίγματος λογαριασμού είναι η 16/01, δηλαδή η κάρτα έχει εκδοθεί στον κάτοχο της κάρτας τον Ιανουάριο του 2016.

9. Ημερομηνία λήξης

Είναι η ημερομηνία λήξης της κάρτας σας και πρέπει να εκδώσετε μια νέα. Είναι γενικά 3 χρόνια μετά την ημερομηνία ανοίγματος του λογαριασμού. Γενικά, οι εκδότες της κάρτας σας στέλνουν τη νέα πιστωτική κάρτα μόνοι τους ή αλλιώς μπορείτε να επικοινωνήσετε μαζί τους για να το κάνετε. Η νέα πιστωτική κάρτα συνοδεύεται από νέο βιογραφικό σημείωμα και νέα ημερομηνία λήξης. Σε ορισμένες περιπτώσεις, ο αριθμός λογαριασμού μπορεί επίσης να είναι διαφορετικός από την προηγούμενη κάρτα. Δίνεται επίσης σε μορφή MM/YY. Στο παραπάνω παράδειγμα, η ημερομηνία λήξης είναι η 19/01, δηλαδή η κάρτα θα λήξει τον Ιανουάριο του 2019.

2.2 Έντυπο αίτησης για χορήγηση πιστωτικής κάρτας

4160. Ε. 4284

ΑΙΤΗΣΗ ΓΙΑ ΧΟΡΗΓΗΣΗ ΑΤΟΜΙΚΗΣ ΠΙΣΤΩΤΙΚΗΣ ΚΑΡΤΑΣ MASTERCARD

ΕΘΝΙΚΗ ΤΡΑΠΕΖΑ ΤΗΣ ΕΛΛΑΔΟΣ (ΚΥΠΡΟΥ) ΛΤΔ ΥΠΟΚ/ΜΑ ( )	Αριθμός Κάρτας :_____ Αριθμός Λ/μού Κάρτας :_____ Αριθμός Επιπρόσθετης Κάρτας :_____ Αριθμός Λ/μού Επιπρόσθετης Κάρτας:_____

(Α) ΠΡΟΣΩΠΙΚΑ ΣΤΟΙΧΕΙΑ			
CIF:			
ΟΝΟΜΑΤΕΠΩΝΥΜΟ :			
ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΠΑΤΕΡΑ :			
ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΣΥΖΥΓΟΥ :			
ΗΜΕΡΟΜΗΝΙΑ ΓΕΝΝΗΣΗΣ :		ΑΡ.ΤΑΥΤΟΤΗΤΑΣ / ΔΙΑΒΑΤΗΡΙΟΥ:	
☐ ΕΓΓΑΜΟΣ/Η	☐ ΑΓΑΜΟΣ/Η	☐ ΧΗΡΟΣ/Α	☐ ΔΙΑΖΕΥΓΜΕΝΟΣ/Η
ΕΞΑΡΤΩΜΕΝΟΙ:	ΚΑΤΩ ΤΩΝ 18 ΕΤΩΝ:	ΑΝΩ ΤΩΝ 18 ΕΤΩΝ:	
Ονοματεπώνυμο με λατινικούς χαρακτήρες όπως εμφανίζεται στο διαβατήριο σας:			
ΔΙΕΥΘΥΝΣΗ ΚΑΤΟΙΚΙΑΣ			
ΤΑΧΥΔΡΟΜΙΚΟΣ ΚΩΔΙΚΑΣ:			
ΤΗΛΕΦΩΝΟ :		ΧΡΟΝΙΑ ΠΑΡΑΜΟΝΗΣ ΣΤΗΝ ΠΙΟ ΠΑΝΩ Δ/ΝΣΗ:	
ΔΙΕΥΘΥΝΣΗ ΑΛΛΗΛΟΓΡΑΦΙΑΣ (αν διαφέρει από την πιο πάνω)			
ΤΑΧΥΔΡΟΜΙΚΟΣ ΚΩΔΙΚΑΣ:			

(Β) ΕΠΑΓΓΕΛΜΑΤΙΚΑ ΣΤΟΙΧΕΙΑ		
ΕΠΑΓΓΕΛΜΑ:	ΕΡΓΟΔΟΤΗΣ:	
ΔΙΕΥΘΥΝΣΗ ΕΡΓΑΣΙΑΣ:		
ΘΕΣΗ:	ΕΤΟΣ ΠΡΟΣΛΗΨΗΣ:	ΤΗΛ. :
ΠΡΟΗΓΟΥΜΕΝΟΣ ΕΡΓΟΔΟΤΗΣ: (Συμπληρώνεται αν τα χρόνια υπηρεσίας στο σημερινό εργοδότη είναι λιγότερα από 2)		
ΕΠΑΓΓΕΛΜΑ ΣΥΖΥΓΟΥ:	ΕΡΓΟΔΟΤΗΣ ΣΥΖΥΓΟΥ:	
ΘΕΣΗ:	ΕΤΟΣ ΠΡΟΣΛΗΨΗΣ:	ΤΗΛ. :

(Γ) ΟΙΚΟΝΟΜΙΚΗ ΚΑΤΑΣΤΑΣΗ (ΠΕΛΑΤΗ ΚΑΙ ΣΥΖΥΓΟΥ)			
ΜΗΝΙΑΙΑ ΕΙΣΟΔΗΜΑΤΑ	**€**	**Μείον ΜΗΝΙΑΙΕΣ ΠΛΗΡΩΜΕΣ**	**€**
Καθαρό μηνιαίο εισόδημα		Ενοίκιο	
Καθαρό μηνιαίο εισόδημα συζύγου		Δόσεις Στεγαστικού Δανείου	
Άλλα μηνιαία εισοδήματα :		Δόσεις Άλλων Δανείων (α)Προσωπικά Δάνεια	
- Ενοίκια		(β) Ενοικιαγορών	
- Μερίσματα		(γ) Καρτών	
- Τόκοι Εισπρακτέοι		(δ) Άλλα	
- Άλλα		Ασφάλιστρα	
		Σπουδές Τέκνων	
		Άλλα	
		Σύνολο Μηνιαίων Π.ληρωμών	
Σύνολο Καθαρών Μηνιαίων Εισοδημάτων		**ΜΗΝΙΑΙΟ ΔΙΑΘΕΣΙΜΟ ΠΟΣΟ**	

(Δ) ΣΥΝΕΡΓΑΣΙΑ ΜΕ ΤΡΑΠΕΖΕΣ				
ΣΧΕΣΗ	**ΕΤΕ (ΚΥΠΡΟΥ) ΛΤΔ**		**ΑΛΛΕΣ ΤΡΑΠΕΖΕΣ**	
	ΠΟΣΟ / ΟΡΙΟ €	**ΥΠΟΛΟΙΠΟ €**	**ΠΟΣΟ / ΟΡΙΟ €**	**ΥΠΟΛΟΙΠΟ €**
ΚΑΤΑΘΕΣΕΙΣ				
ΣΤΕΓΑΣΤΙΚΟ ΔΑΝΕΙΟ				
ΠΡΟΣΩΠΙΚΟ ΔΑΝΕΙΟ				
ΤΡΕΧΟΥΜΕΝΟΣ				
ΠΙΣΤΩΤΙΚΗ ΚΑΡΤΑ				
ΑΛΛΑ				

Αρχικά Κύριου Κατόχου Κάρτας / Κατόχου Κάρτας : _____ / _____

Εικόνα 2 Υπόδειγμα εντύπου αίτησης για χορήγηση πιστωτικής κάρτας

Το παραπάνω έντυπο αποτελεί την αίτηση για τη χορήγηση ατομικής πιστωτικής κάρτας του συστήματος Mastercard. Επίσης, από το έντυπο προκύπτει ότι πρέπει να συμπληρωθούν προσωπικές πληροφορίες, πέραν από τα προσωπικά στοιχεία, επαγγελματικά στοιχεία καθώς και η οικονομική κατάσταση τόσο του πελάτη όσο και συζύγου αυτού, αλλά και τυχόν συνεργασία με τράπεζες.

Συγκεκριμένα, στα προσωπικά στοιχεία περιλαμβάνεται η οικογενειακή κατάσταση, ενώ στα επαγγελματικά περιλαμβάνονται πληροφορίες για την εργασία του πελάτη. Η οικονομική κατάσταση του πελάτη και συζύγου αποτελούν βασικό μέρος της αίτησης αφού εκεί αναγράφονται τα μηνιαία εισοδήματα αλλά και οι μηνιαίες πληρωμές, όπου από τη διαφορά τους προκύπτει το καθαρό μηνιαίο διαθέσιμο ποσό. (DocPlayer, 2015)

2.3 Στατιστικά στοιχεία των πιστωτικών καρτών

Στατιστικά στοιχεία της ΕΚΤ έτους 2021 για τις συναλλαγές πληρωμών χωρίς τη χρήση μετρητών (Ελληνική Ένωση Τραπεζών, 2022)

Στις 22 Ιουλίου 2022, η ΕΚΤ δημοσίευσε τα στατιστικά στοιχεία έτους 2021 για τις συναλλαγές πληρωμών χωρίς τη χρήση μετρητών.

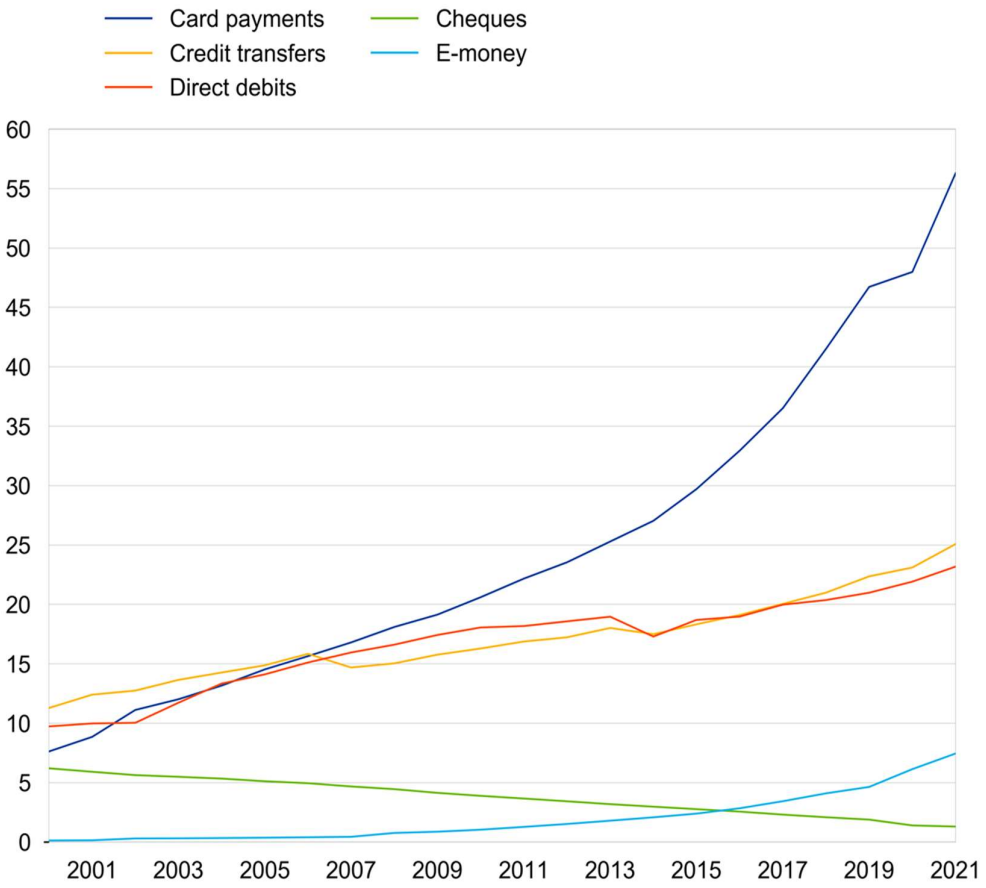
Όπως προκύπτει, ο αριθμός των συναλλαγών που έχουν πραγματοποιηθεί με τη χρήση:

- επιταγών,
- χρεωστικών και πιστωτικών καρτών που έχουν εκδοθεί από παρόχους υπηρεσιών πληρωμών που λειτουργούν στην Ελλάδα,
- υπηρεσιών μεταφοράς πιστώσεων,
- υπηρεσιών άμεσων χρεώσεων,
- πληρωμών ηλεκτρονικού χρήματος (εδώ εμπίπτουν και οι συναλλαγές με προπληρωμένες κάρτες), και
- άλλων υπηρεσιών πληρωμών (π.χ. εμβάσματα)

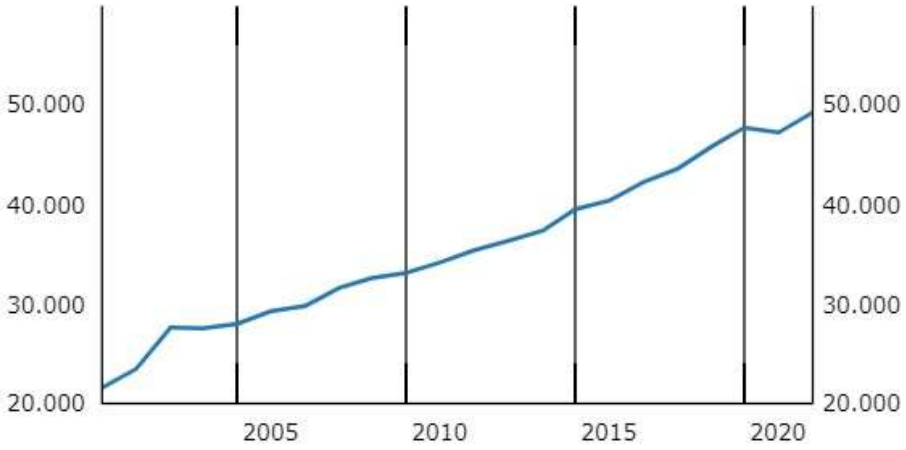
στην Ελλάδα ανήλθαν, το 2021, σε 2,146 δισ. (+379 εκατ. σε σχέση με το 2020) και στις 201 συναλλαγές ανά κάτοικο (+36 συναλλαγές σε σχέση με το 2020), έναντι 333 συναλλαγών ανά κάτοικο στην ευρωζώνη (+37 συναλλαγές σε σχέση με το 2020).

Σε όρους αξίας συναλλαγών, οι μεταφορές πιστώσεων ανήλθαν σε 739,4 δισ. ευρώ (78% επί του συνόλου της αξίας για την Ελλάδα), οι επιταγές στα 69,4 δισ. ευρώ (7,3%), οι άμεσες χρεώσεις στα 4,3 δισ. ευρώ (0,5%), οι κάρτες πληρωμών στα 44,8 δισ. ευρώ (4,7%), το ηλεκτρονικό χρήμα στα 4,2 δισ. ευρώ (0,4%) και οι λοιπές υπηρεσίες πληρωμών (π.χ. εμβάσματα) στα 86,1 δισ. ευρώ (9,1%).

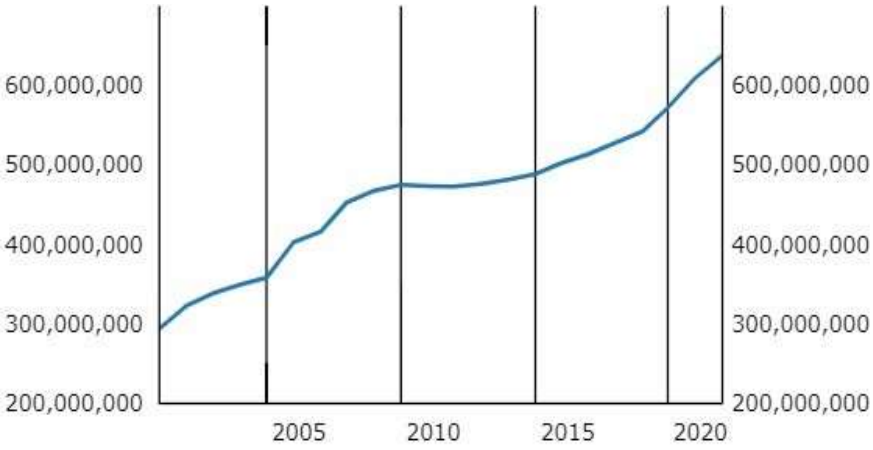
Τέλος, η κατανομή καρτών πληρωμών (χρεωστικών και πιστωτικών), υπηρεσιών μεταφοράς πιστώσεων και υπηρεσιών άμεσων χρεώσεων ανήλθε το 2021 στην Ελλάδα αντίστοιχα σε: 69,5% (Ευρωζώνη: 49%), 22,4% (Ευρωζώνη: 22%) και 1,2% (Ευρωζώνη: 20%). (European Central Bank, 2023)



Σχήμα 1 Ετήσια απεικόνιση συναλλαγών (σε δις) (European Central Bank, 2022)



Σχήμα 2 Μερίδιο πληρωμών με κάρτα στον αριθμό των συνολικών συναλλαγών πληρωμών - με κάρτες που εκδίδονται από PSP κατοίκους - από τη ζώνη του ευρώ (μεταβαλλόμενη σύνθεση) (European Central Bank, 2023)



Σχήμα 3 Αριθμός καρτών με λειτουργία πληρωμής (εκτός από κάρτες με λειτουργία ηλεκτρονικού χρήματος μόνο) - που εκδίδονται από PSP κατοίκους - από τη ζώνη του ευρώ (μεταβαλλόμενη σύνθεση)

2.4 Βιβλιογραφική επισκόπηση του προβλήματος

Εδώ και τρεις δεκαετίες, οι πιστωτικές κάρτες γίνονται όλο και πιο διαθέσιμες στα νοικοκυριά, καθώς οι εκδότες πιστωτικών καρτών έχουν χορηγήσει πίστωση σε πελάτες. Αυτός ο μηχανισμός πληρωμής ήταν μια απάντηση στην ανάγκη να υπάρχει, πρώτον, ένας πιο ευέλικτος τρόπος πληρωμής για έναν πληθυσμό που κινείται όλο και περισσότερο και δεύτερον, να υπάρχει μια άμεσα διαθέσιμη μορφή πίστωσης που απευθύνεται στους Αμερικανούς καταναλωτές . (VISA, 2003)

Όπως έχουμε ήδη προαναφέρει ότι η αύξηση των αγορών μέσω των πιστωτικών καρτών, ολοένα και συνέχιζε να αυξάνεται και οι τράπεζες από τη μεριά τους προκειμένου να διευρύνουν το πελατολόγιο τους και να αυξήσουν τα κέρδη τους, επέτρεψαν στους καταναλωτές με μεσαίο εισόδημα να έχουν δικαίωμα αίτησης για χορήγηση πιστωτικής κάρτας, ανάλογα με την οικογενειακή τους κατάστασή, το ετήσιο εισόδημα τους και των περιουσιακών στοιχείων που μπορεί να έχουν στην κατοχή τους.

Στο πλαίσιο μιας μελέτης, που διενεργήθηκε το 2021, δημιουργήθηκε ένας αυτόματος προγνωστικός παράγοντας έγκρισης πιστωτικών καρτών χρησιμοποιώντας τεχνικές ML. Το σύνολο δεδομένων έγκρισης πιστωτικής κάρτας ανακτήθηκαν από το UCI ML Repository, χρησιμοποιήθηκαν συνολικά 690 παρατηρήσεις με συνολικά 16 features, από τα οποία τα 15 καλύπτουν διάφορες πληροφορίες σχετικά με τους αιτούντες, όπως η ηλικία, το φύλο, το εισόδημά τους, η οικογενειακή κατάσταση, ο αριθμός των ετών απασχόλησης κ.α. ενώ το εναπομείναντα χαρακτηριστικό αποτελεί το τελικό αποτέλεσμα, δηλαδή αν το αίτημα απορρίφθηκε ή εγκρίθηκε. Το προ-επεξεργασμένο σύνολο δεδομένων χωρίζεται σε ένα train set (σε ποσοστό 70% του συνόλου των δεδομένων) και σε ένα test set (σε ποσοστό 30% του συνόλου των δεδομένων). Το train set χρησιμοποιήθηκε για την εκπαίδευση των δύο μοντέλων και η απόδοση του αξιολογήθηκε με το test set. Το μοντέλο ML θα κατασκευαστεί και θα συγκριθεί χρησιμοποιώντας δύο τεχνικές τη λογιστική παλινδρόμηση και το ANN (Artificial Neural Network) και θα αξιολογεί την αίτηση με, αλλά και χωρίς την τεχνική αναζήτησης πλέγματος (Grid Search). Συγκεκριμένα, προτάθηκαν τρία μοντέλα ML. Στο πρώτο (M1) χρησιμοποιήθηκε LR (Logistic Regression) χωρίς αναζήτηση πλέγματος. Στο δεύτερο μοντέλο (M2) χρησιμοποιήθηκε LR με αναζήτηση πλέγματος και στο τρίτο χρησιμοποιήθηκε ANN. Η αξιολόγηση και των τριών μοντέλων έγινε με την ακρίβεια (accuracy).

Στο μοντέλο M2, που χρησιμοποιήθηκε grid search, χρησιμοποιήθηκαν οι υπερπαράμετροι: tol, max_iter, pentalty και solver. Μετά την αναζήτηση πλέγματος λήφθηκε ο συνδυασμός των υπερπαραμέτρων με τις ακόλουθες τιμές: tol=0.0001,

max_iter: 150, penalty: 11 και solver: ‘liblinear’. Ενώ, για το ANN προτάθηκε ένα μοντέλο two-layer με 50 νευρώνες το πρώτο layer και 30 νευρώνες το δεύτερο layer. Η απόδοση στο νευρωνικό δίκτυο, μετράται με τη βοήθεια μιας συνάρτησης απώλειας (loss function). Τέλος, η ακρίβεια του μοντέλου M1 χωρίς την αναζήτηση πλέγματος υπολογίστηκε στο 82% (0.82), ενώ στο M2 η ακρίβεια αυξήθηκε στο 86.69%. Η ακρίβεια του μοντέλου M3 υπολογίστηκε έως 94% στη δοκιμή (test). Ως συμπέρασμα εξάγουμε ότι, η εφαρμογή αναζήτησης πλέγματος βελτίωσε την ικανότητα του μοντέλου ML κατά 4%. Το μοντέλο ANN, ξεπέρασε σημαντικά και τα δύο μοντέλα λογιστικής παλινδρόμησης. Αυτή η μελέτη δείχνει ότι τα μοντέλα ML μπορούν να χρησιμοποιηθούν για την επίλυση πολύπλοκων προβλημάτων που είναι δύσκολο να αποκρυπτογραφηθούν με τη βοήθεια συμβατικών τεχνικών προγραμματισμού. (S. Hemkiran, 2021)

Μια άλλη μελέτη, χρησιμοποίησε το σύνολο δεδομένων και είναι διαθέσιμο στο Πανεπιστήμιο της Καλιφόρνια Irvine (UCI). Συγκεκριμένα, το σύνολο δεδομένων ήταν ένας καλός συνδυασμός κατηγορικών και συνεχών χαρακτηριστικών, αλλά υπήρχαν και μερικές τιμές που έλειπαν. Αυτό το σύνολο περιείχε 690 περιπτώσεις με 16 χαρακτηριστικά. Οι πρώτες δεκαπέντε μεταβλητές είναι χαρακτηριστικά του αιτούντος όπως είναι για παράδειγμα φύλο, ηλικία, οικογενειακή κατάσταση επίπεδο εκπαίδευσης, χρόνια απασχόλησης, αλλά η τελευταία μεταβλητή είναι η κατάσταση έγκρισης πιστωτικής κάρτας ή η μεταβλητή στόχος. Προτάθηκε το μοντέλο ML του Δέντρου Αποφάσεων (decision trees). Το προ-επεξεργασμένο σύνολο δεδομένων χωρίστηκε σε ένα train set (σε ποσοστό 70% του συνόλου των δεδομένων) και σε ένα test set (σε ποσοστό 30% του συνόλου των δεδομένων). Στην ανάπτυξη του μοντέλου του Δέντρου Αποφάσεων (Decision Tree), χρησιμοποιούνται παράμετροι όπως criterion, max_depth, min_samples_leaf και min_samples_split. Η κατανομή μιας κλάσης πριν και μετά τη διαίρεση του συνόλου των δεδομένων, συγκρίνεται με τη συνάρτηση απώλειας (P. Tan, 2005), δηλαδή με τη Gini Impurity και την Εντροπία.

Στο συγκεκριμένο μοντέλο, χρησιμοποιήθηκαν οι υπερπαράμετροι criterion=’gini’ και max_depth=3. Από τον πίνακα συσχέτισης που δημιουργήθηκε, διαπίστωσαν ότι το χαρακτηριστικό του στόχου (εγκεκριμένη κατάσταση) είναι σχεδόν ισορροπημένο έτσι ώστε 383 (55,5%) από τις 690 αιτήσεις απορρίφθηκαν ενώ 307 (44,5%) εγκρίθηκαν (Yengejeh, 2023).

Για την αξιολόγηση της απόδοσης του μοντέλου χρησιμοποιήθηκαν οι παρακάτω μετρήσεις: Model Accuracy, Precision, Sensitivity (recall) και F1-Score. Η accuracy του μοντέλου στο test set είναι 0,86, ενώ η Precision είναι 0,83, η Sensitivity είναι 0,89 και τέλος το F1-Score 0,86 .

Τέλος, σε μια μελέτη (Aji Gautama Putrada, 2022) δημιουργήθηκε ένα μοντέλο αποτελέσματος Tree-Based Pipeline Optimization (TPOT), σχετικά με την ταξινόμηση της έγκρισης ή της αίτησης για την έκδοση πιστωτικής κάρτας. Τα δεδομένα που χρησιμοποιήθηκαν είναι 690 και προέρχονται από την πλατφόρμα Kaggle (University of California). Η έξοδος των δεδομένων είναι 1 και 0, 'Εγκρίθηκε' και 'Δεν εγκρίθηκε' αντίστοιχα η αίτηση από την τράπεζα. Σε αυτά τα δεδομένα υπάρχουν 15 χαρακτηριστικά, όπως το φύλο, η ηλικία, το χρέος, η οικογενειακή κατάσταση, ο κλάδος εργασίας, η εθνικότητα, τα έτη απασχόλησης, η προηγούμενη προεπιλογή κ.α. ενώ μία εξαρτημένη μεταβλητή πλέον των προηγούμενων είναι αυτή του στόχου (εγκρίνεται ή όχι). Το tree του TPOT χρησιμοποιεί μέθοδο Generation Programming (GP). Ένα εξελισσόμενο πρόγραμμα, που ενώ στην αρχή είναι ακατάλληλο, στη συνέχεια περνά από μια γενετική διαδικασία, ώστε τελικά να καθίσταται κατάλληλο.

Στη συνέχεια, ακολούθησε η σύγκριση της μεθόδου TPOT με: (1) naive Bayes, (2) SVM (Support Vector Machine) και (3) Decision Trees. Για την σύγκριση χρησιμοποιήθηκαν οι μετρικές accuracy, recall και AUC. Επιπλέον, στο TPOT χρησιμοποιήθηκε cross validation για να μετρήσει την απόδοση κάθε προγράμματος αποτελέσματος της μεθόδου GP.

Από τον πίνακα συσχέτισης που δημοσιεύτηκε στη μελέτη, προκύπτει η συσχέτιση μεταξύ των παραμέτρων γενετικού προγραμματισμού και της εσωτερικής βαθμολογίας CV. Συγκεκριμένα, μέτρια συσχέτιση έχουν δύο παράμετροι, η generation και ο mutation_count, ενώ η generation έχει την ισχυρότερη συσχέτιση με μια τιμή PCC (συντελεστής συσχέτισης Pearson) ίση με 0.48. Ασθενή συσχέτιση έχει ο crossover_count.

Για την εύρεση της βέλτιστης ταξινόμησης για την έγκριση της πιστωτικής κάρτας χρησιμοποιώντας τη μέθοδο TPOT, είναι μεγάλης σημασίας η συσχέτιση μεταξύ της generation και της interval_CV.

Τα καλύτερα αποτελέσματα από τις πέντε συνεχόμενες generation είναι: 0.877, 0.880, 0.880, 0.883 και 0.883 . Έτσι από ένα σύνολο 294 μοντέλων που παράγονται από την TPOT, 21 μοντέλα επιτυγχάνουν την υψηλότερη απόδοση (accuracy>0.88). Ακολούθως, αναφέρεται το καλύτερο μοντέλο που δημιουργήθηκε από το TPOT. Το καλύτερο μοντέλο είναι το stack estimator .

No.	Specification	Value
1	Model	Stack estimator
2	Ensemble model 1	Random forest
3	Hyperparameters	6
4	Parameter 1	Bootstrap = True
5	Parameter 2	Criterion = Gini
6	Parameter 3	Max Features = 0.95
7	Parameter 4	Min. Samples Leaf = 19
8	Parameter 5	Min. Samples Split = 15
9	Parameter 6	N. Estimators = 100
10	Ensemble model 2	Gaussian naive Bayes
11	Hyperparameters	0
12	CV Score	0.89

Πίνακας 1 Προδιαγραφές καλύτερου μοντέλου TPOT

Οπότε συγκρίνοντας τα αποτελέσματα του TPOT με τις άλλες μελέτες: (1) naïve Bayes, (2) SVM και (3) Decision Trees, καταλήξανε ότι έχει την υψηλότερη accuracy 0.89 [(1) 0.823, (2) 0.852 και (3) 0.796 αντίστοιχα], ενώ το TPOT μπορεί να παράγει ROC με υψηλότερο AUC, δηλαδή 0.94, σε σχέση με τις άλλες μελέτες [(1) 0.897, (2) 0.98 και (3) 0.796 αντίστοιχα] .

2.5 Python

Η Python είναι αυτή τη στιγμή (από τον Νοέμβριο του 2022) η πιο δημοφιλής γλώσσα προγραμματισμού στον κόσμο και η βάση χρηστών της αυξάνεται συνεχώς. Χρησιμοποιείται συχνά από πολλές βιομηχανίες και εταιρείες για να αναλύσουν δεδομένα, να δημιουργήσουν μοντέλα μηχανικής εκμάθησης, να δημιουργήσουν ιστότοπους και να προγραμματίσουν λογισμικό. (datacamp, 2022)

2.5.1 Πλεονεκτήματα της Python

Τα κύρια πλεονεκτήματα της χρήσης της Python που την καθιστούν μια τόσο ισχυρή και διαδεδομένη γλώσσα προγραμματισμού:

- Έχει μια διαισθητική σύνταξη που μοιάζει με μια φυσική αγγλική γλώσσα και ως εκ τούτου είναι εύκολο στην εκμάθηση, ειδικά για άτομα που μόλις μπαίνουν στον κόσμο του προγραμματισμού.
- Λόγω της φιλικής προς τον άνθρωπο σύνταξης, είναι εύκολο να γραφτεί, να διαβαστεί και να εντοπιστεί σφάλματα.
- Παρέχει μια εκτεταμένη τυπική βιβλιοθήκη και μια ευρεία επιλογή από καλά τεκμηριωμένες και ολοκληρωμένες πρόσθετες βιβλιοθήκες και ενότητες
- Είναι δωρεάν τόσο για ιδιώτες όσο και για επιχειρήσεις.
- Χάρη στην τεράστια κοινότητα υποστήριξης, η Python αναπτύσσεται, βελτιώνεται και επεκτείνεται συνεχώς.

- Μπορεί να ενσωματωθεί σε οποιοδήποτε έργο και να χρησιμοποιηθεί για την επίλυση προηγμένων προβλημάτων.
- Ως γλώσσα γενικής χρήσης, έχει διάφορες εφαρμογές σε πολλούς τομείς.

Εικόνα 3 Χρησιμότητα της Python

2.6 Tableau

Το Tableau Software είναι μια αμερικανική εταιρεία λογισμικού διαδραστικής απεικόνισης δεδομένων που επικεντρώνεται στην επιχειρηματική ευφυΐα. Τα προϊόντα Tableau ερωτούν σχεσιακές βάσεις δεδομένων, διαδικτυακούς κύβους αναλυτικής επεξεργασίας, βάσεις δεδομένων cloud και υπολογιστικά φύλλα για να δημιουργήσουν

οπτικοποιήσεις δεδομένων τύπου γραφήματος. Το λογισμικό μπορεί επίσης να εξάγει, να αποθηκεύει και να ανακτά δεδομένα από μια μηχανή δεδομένων στη μνήμη. (Wikipedia, 2021)

Ιδρύθηκε το 2003 στο Mountain View της Καλιφόρνια και επί του παρόντος έχει την έδρα του στο Σιάτλ της Ουάσιγκτον. Το 2019 η εταιρεία εξαγοράστηκε από τη Salesforce για 15,7 δισεκατομμύρια δολάρια. Εκείνη την εποχή, αυτή ήταν η μεγαλύτερη εξαγορά από την Salesforce (ηγέτη στον τομέα του CRM) από την ίδρυσή της. Αργότερα ξεπεράστηκε από την εξαγορά του Slack από την Salesforce.

Εικόνα 4 Τί είναι το Tableau?

Το Tableau προσφέρει μεταφορά και απόθεση και άλλες δυνατότητες, όπως πολλαπλές μορφές γραφημάτων και δυνατότητες χαρτογράφησης.

- ✓ "Maps". Tableau. Retrieved November 8, 2022.
- ✓ "Best Dashboard Visualization Tools According to 30 Experts | Databox Blog". Databox. August 15, 2022. Retrieved November 8, 2022.

Πηγές δεδομένων

Το Tableau Software μπορεί να συνδεθεί με πηγές δεδομένων όπως κανονικά αρχεία κειμένου *(.txt, .csv)*, Microsoft Excel *(.xlsx)*, Microsoft Access *(.accdb)*, εισαγωγή από βιβλίο εργασίας Tableau *(.tbm)* ή Εξαγωγή δεδομένων πίνακα Tableau *(.tds)*. (Daniel, Tableau Your Data, 2013)

2.6.1 Τύπος δεδομένων

Το Tableau εκφράζει αυτόματα τύπους και πεδία δεδομένων.

Το Tableau θα χρησιμοποιήσει τον τύπο δεδομένων που έχει ορίσει η πηγή δεδομένων, εάν εξέλθει, ή θα επιλέξει έναν τύπο δεδομένων εάν η πηγή δεδομένων δεν ορίζει έναν.

Στο Tableau, υποστηρίζονται οι ακόλουθοι τύποι δεδομένων (Daniel, Tableau Your Data (2nd ed.), 2016):

- ✓ Τιμή κειμένου
- ✓ Τιμή δεδομένων
- ✓ Αξία δεδομένων και χρόνου
- ✓ Αριθμητική αξία
- ✓ Γεωγραφικές τιμές (Γεωγραφικό πλάτος και γεωγραφικό μήκος που χρησιμοποιούνται για χάρτες)
- ✓ Τιμές Boolean (Συνθήκες True / False)

2.7 Μηχανική μάθηση

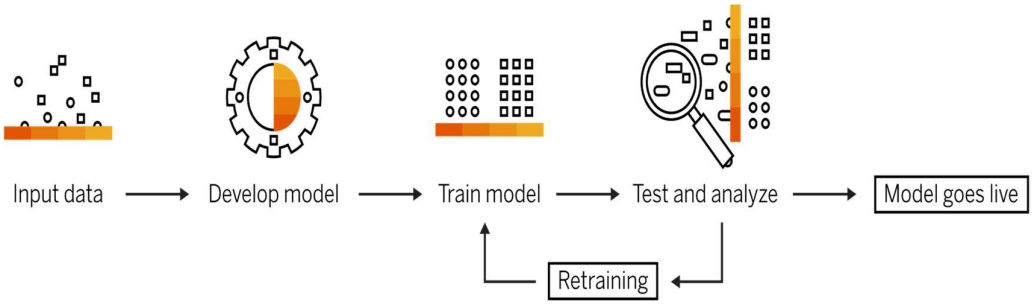
2.7.1 Έννοια μηχανικής μάθησης

Η μηχανική μάθηση είναι ένα υποσύνολο της τεχνητής νοημοσύνης (AI). Επικεντρώνεται στη διδασκαλία των υπολογιστών να μαθαίνουν από τα δεδομένα και να βελτιώνονται με την εμπειρία – αντί να είναι ρητά προγραμματισμένοι να το κάνουν. Στη μηχανική μάθηση, οι αλγόριθμοι εκπαιδεύονται για να βρουν μοτίβα και συσχετίσεις σε μεγάλα σύνολα δεδομένων και να λαμβάνουν τις καλύτερες αποφάσεις και προβλέψεις με βάση αυτή την ανάλυση. Οι εφαρμογές μηχανικής μάθησης βελτιώνονται με τη χρήση και γίνονται πιο ακριβείς όσο περισσότερα δεδομένα έχουν πρόσβαση.

Οι εφαρμογές της μηχανικής μάθησης βρίσκονται παντού γύρω μας – στα σπίτια μας, στα καλάθια αγορών μας, στα μέσα ψυχαγωγίας μας και στην υγειονομική μας περίθαλψη. (SAP, 2023)

2.7.2. Πώς λειτουργεί η μηχανική μάθηση;

Η μηχανική μάθηση αποτελείται από διαφορετικούς τύπους μοντέλων μηχανικής μάθησης, χρησιμοποιώντας διάφορες αλγοριθμικές τεχνικές. Ανάλογα με τη φύση των δεδομένων και το επιθυμητό αποτέλεσμα, ένα από τα τέσσερα μοντέλα μάθησης μπορεί να χρησιμοποιηθεί: επιβλεπόμενο, χωρίς επίβλεψη, ημι-εποπτευόμενο, ή ενίσχυση. Σε καθένα από τα εν λόγω μοντέλα, μπορεί να εφαρμοστεί μία ή περισσότερες αλγοριθμικές τεχνικές – σε σχέση με τα σύνολα δεδομένων που χρησιμοποιούνται και τα επιδιωκόμενα αποτελέσματα. Οι αλγόριθμοι μηχανικής μάθησης είναι βασικά σχεδιασμένοι για να ταξινομούν τα πράγματα, να βρίσκουν μοτίβα, να προβλέπουν τα αποτελέσματα και να λαμβάνουν τεκμηριωμένες αποφάσεις. Οι αλγόριθμοι μπορούν να χρησιμοποιηθούν ένας κάθε φορά ή να συνδυαστούν για να επιτευχθεί η καλύτερη δυνατή ακρίβεια όταν εμπλέκονται σύνθετα και πιο απρόβλεπτα δεδομένα.



Σχήμα 4 Πώς λειτουργεί η διαδικασία μηχανικής μάθησης

2.7.3 Τι είναι η επιβλεπόμενη μάθηση;

Η επιβλεπόμενη μάθηση είναι το πρώτο από τα τέσσερα μοντέλα μηχανικής μάθησης. Στους επιτηρούμενους αλγορίθμους μάθησης, η μηχανή διδάσκεται από το παράδειγμα. Τα επιτηρούμενα μοντέλα μάθησης αποτελούνται από ζεύγη δεδομένων «εισόδου» και «εξόδου», όπου η έξοδος επισημαίνεται με την επιθυμητή τιμή.

Επίσης, τα μοντέλα αυτά χρησιμοποιούνται σε πολλές από τις εφαρμογές με τις οποίες αλληλεπιδρούμε καθημερινά, όπως οι μηχανές συστάσεων για προϊόντα και εφαρμογές ανάλυσης κυκλοφορίας.

2.7.4 Τι είναι η μη επιβλεπόμενη μάθηση;

Η μη επιβλεπόμενη μάθηση είναι το δεύτερο από τα τέσσερα μοντέλα μηχανικής μάθησης. Η μηχανή μελετά τα δεδομένα εισόδου – πολλά από τα οποία είναι αδόμητα– και αρχίζει να εντοπίζει μοτίβα και συσχετίσεις, χρησιμοποιώντας όλα τα σχετικά, προσβάσιμα δεδομένα. Για τις μηχανές, η «εμπειρία» ορίζεται από την ποσότητα των δεδομένων που εισάγονται και καθίστανται διαθέσιμα. Κοινά παραδείγματα ανεπίβλεπτων εφαρμογών μάθησης περιλαμβάνουν αναγνώριση προσώπου, ανάλυση γονιδιακών ακολουθιών, έρευνα αγοράς και κυβερνοασφάλεια.

2.7.5 Τι είναι η ημι-επιβλεπόμενη μάθηση;

Η ημι-εποπτευόμενη μάθηση είναι το τρίτο από τέσσερα μοντέλα μηχανικής μάθησης. Σε έναν τέλειο κόσμο, όλα τα δεδομένα θα ήταν δομημένα και επισημασμένα πριν εισαχθούν σε ένα σύστημα. Αλλά δεδομένου ότι αυτό προφανώς δεν είναι εφικτό, η ημι-εποπτευόμενη μάθηση γίνεται μια εφαρμόσιμη λύση όταν υπάρχουν τεράστιες ποσότητες ακατέργαστων, μη δομημένων δεδομένων. Αυτό το μοντέλο αποτελείται από την εισαγωγή μικρών ποσοτήτων δεδομένων με ετικέτα για να αυξήσει τα σύνολα δεδομένων χωρίς ετικέτα. Ουσιαστικά, τα επισημασμένα δεδομένα ενεργούν για να δώσουν μια αρχή λειτουργίας στο σύστημα και μπορούν να βελτιώσουν σημαντικά την

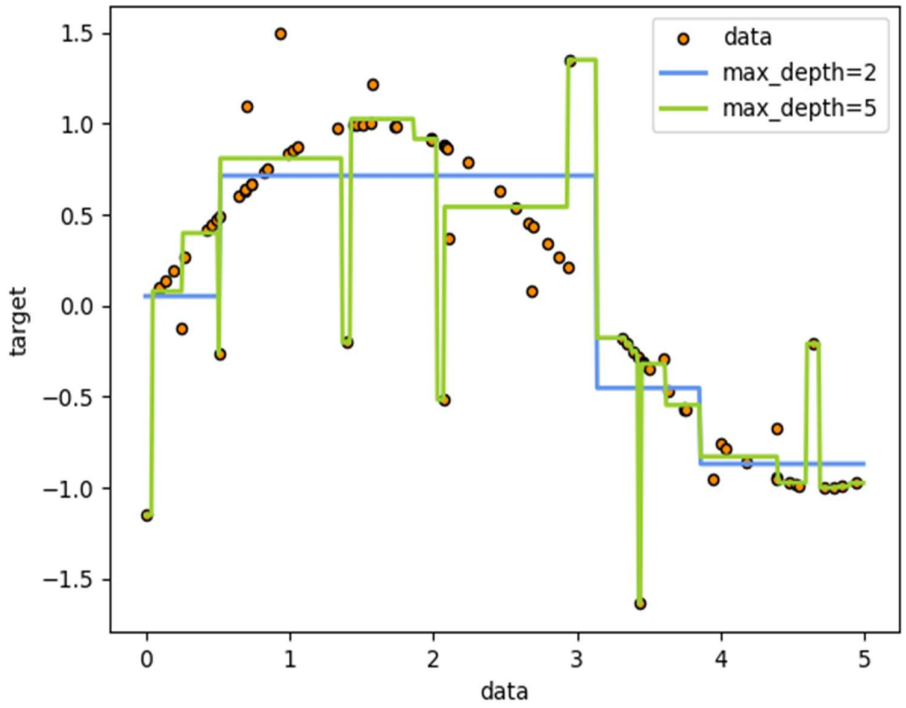
ταχύτητα και την ακρίβεια της μάθησης. Ένας ημι-εποπτευόμενος αλγόριθμος μάθησης καθοδηγεί τη μηχανή να αναλύσει τα επισημασμένα δεδομένα για τις συσχετιζόμενες ιδιότητες που θα μπορούσαν να εφαρμοστούν στα δεδομένα χωρίς ετικέτα (Olivier Chapelle, 2006).

2.7.6 Τι είναι η ενισχυμένη μάθηση;

Η ενισχυμένη μάθηση είναι το τέταρτο μοντέλο μηχανικής μάθησης. Στην επιβλεπόμενη μάθηση, η μηχανή λαμβάνει το κλειδί απάντησης και μαθαίνει βρίσκοντας συσχετίσεις μεταξύ όλων των σωστών αποτελεσμάτων. Το μοντέλο ενισχυμένης μάθησης δεν περιλαμβάνει ένα κλειδί απάντησης αλλά, αντίθετα, εισάγει ένα σύνολο επιτρεπόμενων ενεργειών, κανόνων και πιθανών τελικών καταστάσεων. Σε περιπτώσεις όπου το επιθυμητό αποτέλεσμα είναι μεταβλητό, το σύστημα πρέπει να μάθει από την εμπειρία και την ανταμοιβή. Στα μοντέλα ενισχυτικής μάθησης, η «ανταμοιβή» είναι αριθμητική και προγραμματίζεται στον αλγόριθμο ως κάτι που το σύστημα επιδιώκει να συλλέξει.

2.7.7 Decision Trees – Δέντρα αποφάσεων

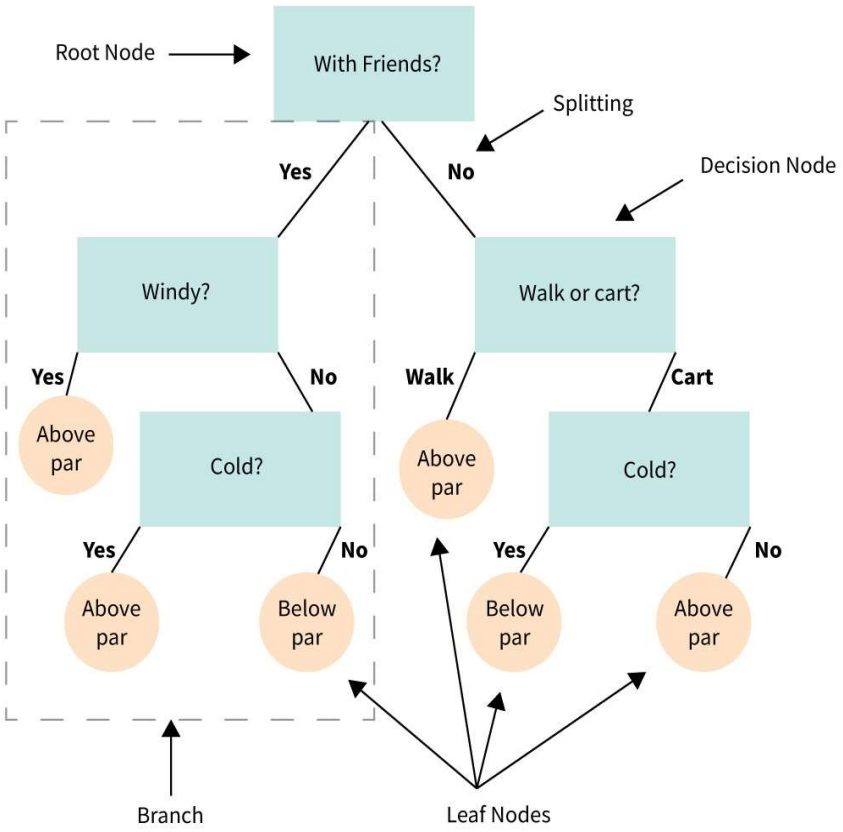
Τα Δέντρα Αποφάσεων (DTs), είναι μια μη παραμετρική εποπτευόμενη μέθοδος μάθησης που χρησιμοποιείται για ταξινόμηση και παλινδρόμηση . Ο στόχος είναι να δημιουργηθεί ένα μοντέλο που προβλέπει την τιμή μιας μεταβλητής στόχου μαθαίνοντας απλούς κανόνες απόφασης που συνάγονται από τα χαρακτηριστικά δεδομένων. Ένα δέντρο μπορεί να θεωρηθεί ως μια τμηματικά σταθερή προσέγγιση.
Για παράδειγμα, τα δέντρα αποφάσεων μαθαίνουν από δεδομένα να προσεγγίζουν μια καμπύλη ημιτόνου με ένα σύνολο κανόνων απόφασης εάν-τότε-άλλο. Όσο πιο βαθιά είναι το δέντρο, τόσο πιο περίπλοκοι είναι οι κανόνες απόφασης και τόσο πιο κατάλληλο είναι το μοντέλο. (developers i.-l. , 2023)



Σχήμα 5 Δέντρα αποφάσεων προσαρμόζουν τα δεδομένα στην καμπύλη ημιτόνου

2.7.8 Ορισμένοι βασικοί όροι ενός δέντρου αποφάσεων

- **Root node:** Η βάση του δέντρου αποφάσεων.
- **Διαίρεση (splitting):** Η διαδικασία διαίρεσης ενός κόμβου σε πολλαπλούς υπο-κόμβους.
- **Κόμβος απόφασης (Decision Node):** Όταν ένας υπο-κόμβος χωρίζεται περαιτέρω σε επιπλέον υπο-κόμβους.
- **Κόμβος φύλλου (Leaf Nodes):** Όταν ένας υπο-κόμβος δεν χωρίζεται περαιτέρω σε επιπλέον υπο-κόμβους.
- **Κλάδεμα:** Η διαδικασία αφαίρεσης υπο-κόμβων ενός δέντρου αποφάσεων.
- **Διακλάδωση (Branch):** Μια υποενότητα του δέντρου αποφάσεων που αποτελείται από πολλούς κόμβους. (MastersInDataScience.org, 2023)



Σχήμα 6 Λειτουργία των Δέντρων αποφάσεων

2.7.9 Μέτρα επιλογής χαρακτηριστικών

Εάν το σύνολο δεδομένων αποτελείται από **N** χαρακτηριστικά, τότε η λήψη απόφασης για το ποιο χαρακτηριστικό θα τοποθετηθεί στη ρίζα ή σε διαφορετικά επίπεδα του δέντρου ως εσωτερικοί κόμβοι, είναι ένα πολύπλοκο βήμα. Αν ακολουθηθεί μια τυχαία προσέγγιση, μπορεί να δώσει αποτελέσματα με χαμηλή ακρίβεια. (Chauhan, 2022)

2.7.10 Παράμετροι δέντρου αποφάσεων

- ✓ **κριτήριο** ***{"gini", "entropy", "log_loss"}, default="gini"***

  Η λειτουργία για τη μέτρηση της ποιότητας ενός διαχωρισμού.

- ✓ **splitter** ***{"best", "random"}, default="best"***

  Η στρατηγική που χρησιμοποιήθηκε για την επιλογή του διαχωρισμού σε κάθε κόμβο.

- ✓ **max_depth :** ***int, default=Καμία***

  Το μέγιστο βάθος του δέντρου.

- ✓ **min_samples_split :** ***int ή float, προεπιλογή=2***

  Ο ελάχιστος αριθμός δειγμάτων που απαιτούνται για τον διαχωρισμό ενός εσωτερικού κόμβου.

- ✓ **min_samples_leaf** ***int ή float, προεπιλογή=1***

  Ο ελάχιστος αριθμός δειγμάτων που απαιτείται να βρίσκονται σε έναν κόμβο φύλλου.

- ✓ **random_state** ***int, RandomState instance ή None, default=Καμία***

  Ελέγχει την τυχαιότητα του εκτιμητή. (scikit-learn, 2023)

2.7.11 Συντονισμός υπερπαραμέτρων με την Αναζήτηση Πλέγματος (Grid Search)

Στη μηχανική μάθηση, θα χρησιμοποιήσουμε ένα σύνολο υπερπαραμέτρων με βέλτιστο τρόπο, ώστε να συμβάλλουν με το βέλτιστο τρόπο στον αλγόριθμο εκμάθησης. Σε κάθε περίπτωση τίθεται το πρόβλημα της επιλογής των βέλτιστων υπερπαραμέτρων . Μια υπερπαράμετρος, είναι μία παράμετρος της οποίας η τιμή χρησιμοποιείται για τον έλεγχο της μαθησιακής διαδικασίας. Μια μεθοδολογία για την επιλογή των σωστών υπερπαραμέτρων είναι η Αναζήτηση Πλέγματος (Grid Search). Πρόκειται για μία τεχνική συντονισμού που επιχειρεί να υπολογίσει τις βέλτιστες τιμές των υπερπαραμέτρων. Βέβαια, στην πλειονότητα των περιπτώσεων είναι μια εξαντλητική αναζήτηση που εκτελείται σε συγκεκριμένες τιμές των παραμέτρων ενός μοντέλου. Το μοντέλο είναι επίσης γνωστό ως εκτιμητής.

2.7.12 Ακρίβεια εναντίον ανάκλησης

Η *Ακρίβεια* μας λέει πόσες από τις σωστά προβλεπόμενες περιπτώσεις αποδείχθηκαν πραγματικά θετικές. (Bhandari, 2023)

$$Precision = \frac{TP}{TP + FP}$$

Ο υπολογισμός της ακρίβειας θα καθόριζε εάν το μοντέλο μας είναι αξιόπιστο ή όχι.

Η *Ανάκληση* μας λέει πόσες από τις πραγματικές θετικές περιπτώσεις μπορέσαμε να προβλέψουμε σωστά με το μοντέλο μας.

2.7.13 Τι είναι το F1-Score;

Στην πράξη, όταν προσπαθούμε να αυξήσουμε την ακρίβεια του μοντέλου μας, η ανάκληση μειώνεται και αντίστροφα. Η βαθμολογία F1 καταγράφει και τις δύο τάσεις σε μία μόνο τιμή: (Bhandari, 2023)

$$F1 - score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

Η βαθμολογία F1 είναι ένας αρμονικός μέσος όρος Ακρίβειας και Ανάκλησης , και έτσι δίνει μια συνδυαστική ιδέα για αυτές τις δύο μετρήσεις.

- Είναι **μέγιστο** όταν η Ακρίβεια είναι ίση με την Ανάκληση.

Η ερμηνεία της βαθμολογίας F1 είναι κακή. Αυτό σημαίνει ότι δεν γνωρίζουμε τι μεγιστοποιεί ο ταξινομητής μας – ακρίβεια ή ανάκληση. Έτσι, το χρησιμοποιούμε σε συνδυασμό με άλλες μετρήσεις αξιολόγησης, δίνοντάς μας μια πλήρη εικόνα του αποτελέσματος.

2.7.14 Μήτρα Σύγχυσης (Confusion Matrix)

Σημαντικοί όροι σε μια μήτρα σύγχυσης

Αληθινό θετικό (TP)

- Η προβλεπόμενη τιμή ταιριάζει με την πραγματική τιμή ή η προβλεπόμενη κατηγορία ταιριάζει με την πραγματική κατηγορία.
- Η πραγματική τιμή ήταν θετική και το μοντέλο προέβλεψε μια θετική τιμή.

Αληθινό αρνητικό (TN)

- Η προβλεπόμενη τιμή ταιριάζει με την πραγματική τιμή ή η προβλεπόμενη κατηγορία ταιριάζει με την πραγματική κατηγορία.
- Η πραγματική τιμή ήταν αρνητική και το μοντέλο προέβλεψε μια αρνητική τιμή.

False Positive (FP) – Σφάλμα τύπου I

- Η προβλεπόμενη τιμή είχε προβλεφθεί λανθασμένα.
- Η πραγματική τιμή ήταν αρνητική, αλλά το μοντέλο προέβλεψε μια θετική τιμή.
- Γνωστό και ως σφάλμα τύπου I.

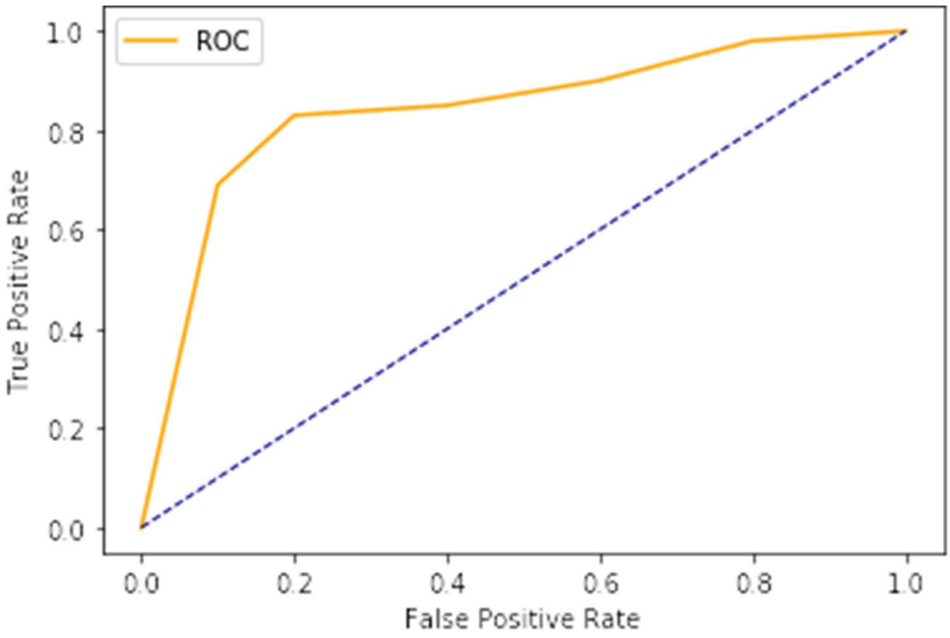
False Negative (FN) – Σφάλμα τύπου II

- Η προβλεπόμενη τιμή είχε προβλεφθεί λανθασμένα.
- Η πραγματική τιμή ήταν θετική, αλλά το μοντέλο προέβλεψε μια αρνητική τιμή.
- Γνωστό και ως σφάλμα τύπου II. (Bhandari, 2023)

2.7.15 The Receiver Operating Characteristic (ROC) Curve

Η καμπύλη AUC–ROC είναι η μέτρηση επιλογής μοντέλου για προβλήματα ταξινόμησης δυαδικών/πολλαπλών κατηγοριών. Το ROC είναι μια καμπύλη πιθανότητας για διαφορετικές κλάσεις. Το ROC μας λέει πόσο καλό είναι το μοντέλο για τη διάκριση των δεδομένων κλάσεων, όσον αφορά την προβλεπόμενη πιθανότητα.

Μια τυπική καμπύλη ROC έχει False Positive Rate (FPR) στον άξονα X και True Positive Rate (TPR) στον άξονα Y.



Σχήμα 7 Receiver Operating Characteristic (ROC) Curve

Η περιοχή που καλύπτεται από την καμπύλη είναι η περιοχή μεταξύ της πορτοκαλί γραμμής (ROC) και του άξονα. Αυτή η περιοχή που καλύπτεται είναι AUC. Όσο μεγαλύτερη είναι η περιοχή που καλύπτεται, τόσο καλύτερα τα μοντέλα μηχανικής μάθησης διακρίνουν τις συγκεκριμένες τάξεις. Η ιδανική τιμή για το AUC είναι 1. (Stack Abuse, 2023)

2.7.16 Random Forest Classifier

Οι αλγόριθμοι που βασίζονται σε δέντρα είναι δημοφιλείς μέθοδοι μηχανικής εκμάθησης που χρησιμοποιούνται για την επίλυση προβλημάτων εποπτευόμενης μάθησης. Αυτοί οι αλγόριθμοι είναι ευέλικτοι και μπορούν να λύσουν κάθε είδους πρόβλημα (ταξινόμηση ή παλινδρόμηση). (David, 2020)

Υπάρχουν διαφορετικοί αλγόριθμοι που βασίζονται σε δέντρα που μπορείτε να χρησιμοποιήσετε, όπως π.χ

- Δέντρα απόφασης
- Τυχαίο Δάσος
- Ενίσχυση κλίσης
- Bagging (Συγκέντρωση Bootstrap)

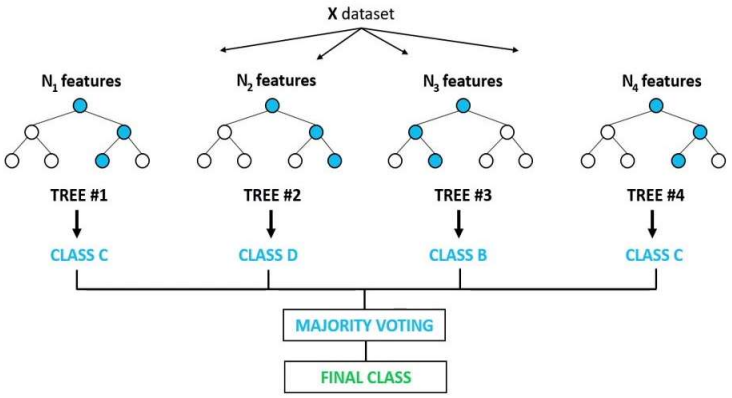
Το Random Forest είναι ένας από τους πιο δημοφιλείς αλγόριθμους εποπτευόμενης μάθησης βάσει δέντρων. Είναι επίσης το πιο ευέλικτο και εύκολο στη χρήση.

Ο αλγόριθμος μπορεί να χρησιμοποιηθεί για την επίλυση προβλημάτων τόσο ταξινόμησης όσο και παλινδρόμησης. Το τυχαίο δάσος τείνει να συνδυάζει εκατοντάδες **δέντρα αποφάσεων** και στη συνέχεια εκπαιδεύει κάθε δέντρο απόφασης σε διαφορετικό δείγμα των παρατηρήσεων. Οι τελικές προβλέψεις του τυχαίου δάσους γίνονται με τον μέσο όρο των προβλέψεων κάθε μεμονωμένου δέντρου.

2.7.17 Διαφορά μεταξύ Random Forest και Decision Trees

Μερικά κύρια χαρακτηριστικά σύγκρισης είναι τα ακόλουθα: (Banerjee, 2019)

- ❖ Τα τυχαία δάση είναι ένα σύνολο πολλαπλών δέντρων αποφάσεων.
- ❖ Τα δέντρα απόφασης είναι υπολογιστικά πιο γρήγορα σε σύγκριση με τα τυχαία δάση.
- ❖ Τα βαθιά δέντρα αποφάσεων μπορεί να υποφέρουν από υπερβολική προσαρμογή.
- ❖ Το τυχαίο δάσος αποτρέπει την υπερπροσαρμογή δημιουργώντας δέντρα σε τυχαία δάση.
- ❖ Το τυχαίο δάσος είναι δύσκολο να ερμηνευτεί.
- ❖ Όμως, ένα δέντρο αποφάσεων είναι εύκολα ερμηνεύσιμο και μπορεί να μετατραπεί σε κανόνες.



Σχήμα 8 Διαίσθηση αλγόριθμου Random Forest

2.7.18 Η σημασία του συντονισμού υπερπαραμέτρων

Ο συντονισμός υπερπαραμέτρων είναι **σημαντικός** για τους αλγόριθμους. Βελτιώνει τη συνολική τους απόδοση ενός μοντέλου μηχανικής μάθησης και τίθεται πριν από τη διαδικασία εκμάθησης και συμβαίνει εκτός του μοντέλου.

Εάν δεν πραγματοποιηθεί συντονισμός υπερπαραμέτρων, το μοντέλο θα παράγει σφάλματα και ανακριβή αποτελέσματα καθώς η συνάρτηση απώλειας δεν ελαχιστοποιείται.

2.7.19 XGB Classifier

Το XGBoost σημαίνει **Extreme Gradient Boosting** , είναι μια αποτελεσματική βιβλιοθήκη μηχανικής εκμάθησης που βασίζεται στο χαρτί **Greedy Function Approximation: A Gradient Boosting Machine, του Friedman** (II.Friedman, 2001)**.**

Το XGBoost εφαρμόζει έναν αλγόριθμο **Gradient Boosting** που βασίζεται σε δέντρα αποφάσεων. (Wikipedia, 2023).

Τα δέντρα αυτά είναι φτωχά μοντέλα μεμονωμένα, αλλά όταν είναι ομαδοποιημένα μπορούν να έχουν πραγματικά απόδοση. (Lemagnen, 2017)

2.7.20 Η διαφορά μεταξύ XGBoost και Random Forest

Η διαφορά μεταξύ XGBoost και Random Forest, έγκειται στον τρόπο κατασκευής και συνδυασμού αυτών των δέντρων.

- Το Random Forest δημιουργεί παράλληλα δέντρα απόφασης με πλήρη ανάπτυξη σε υποδείγματα των δεδομένων.
- Κάθε δέντρο είναι εξαιρετικά εξειδικευμένο στην πρόβλεψη στο υπόδειγμα του και δεν γενικεύεται σε καλό βαθμό (υψηλή διακύμανση).
- Συνδυάζοντας τις προβλέψεις που γίνονται από κάθε μεμονωμένο δέντρο, ο αλγόριθμος Random Forest μειώνει τη διακύμανση και δίνει καλή απόδοση. (Lemagnen, 2017)

- **Το XGBoost** από την άλλη πλευρά,
  - δημιουργεί πολύ σύντομα και απλά δέντρα αποφάσεων επαναληπτικά.
  - Κάθε δέντρο αποκαλείται *«αδύναμος μαθητής»* για την υψηλή του προκατάληψη.
  - Το XGBoost ξεκινά δημιουργώντας ένα πρώτο απλό δέντρο που έχει από μόνο του κακή απόδοση. Στη συνέχεια χτίζει ένα άλλο δέντρο το οποίο εκπαιδεύεται να προβλέπει τι δεν μπόρεσε να κάνει το πρώτο δέντρο και είναι και ο ίδιος αδύναμος μαθητής.
  - Ο αλγόριθμος συνεχίζει με τη διαδοχική δημιουργία περισσότερων αδύναμων μαθητών, ο καθένας διορθώνοντας το προηγούμενο δέντρο μέχρι να επιτευχθεί μια συνθήκη διακοπής, όπως ο αριθμός των δέντρων (εκτιμητές) που θα δημιουργηθούν

Το XGBoost χρησιμοποιείται ευρέως **στη Μηχανική Εκμάθηση** και έγινε ιδιαίτερα διάσημο στο **Kaggle** , τον ιστότοπο του διαγωνισμού μηχανικής μάθησης. Όπως είπε ο Anthony Goldbloom CEO της Kaggle το 2016 , όταν το XGBoost γινόταν μεγάλο στην ανταγωνιστική Machine Learning: (NVIDIA, 2023)

> ''*…Σχεδόν πάντα ήταν σύνολα δέντρων αποφάσεων που κέρδιζαν διαγωνισμούς. Παλιότερα ήταν το τυχαίο δάσος που ήταν ο μεγάλος νικητής, αλλά τους τελευταίους έξι μήνες εμφανίστηκε ένας νέος αλγόριθμος που ονομάζεται* ***XGboost*** *και κερδίζει σχεδόν κάθε διαγωνισμό στην κατηγορία δομημένων δεδομένων…*''

Από τα παραπάνω προκύπτυει ότι το XGBoost παρέχει παράλληλη ενίσχυση δέντρων και είναι η κορυφαία βιβλιοθήκη μηχανικής εκμάθησης για προβλήματα παλινδρόμησης, ταξινόμησης και κατάταξης.

Είναι σημαντική η κατανόηση των εννοιών και τους αλγόριθμους μηχανικής μάθησης στους οποίους βασίζεται το XGBoost: (Mitchell, 2017)

i. εποπτευόμενη μηχανική εκμάθηση,
ii. δέντρα αποφάσεων,
iii. εκμάθηση συνόλου και
iv. ενίσχυση κλίσης .

2.7.21 Οι παράμετροι XGBoost μπορούν να ταξινομηθούν σε τέσσερις διακριτές κατηγορίες:

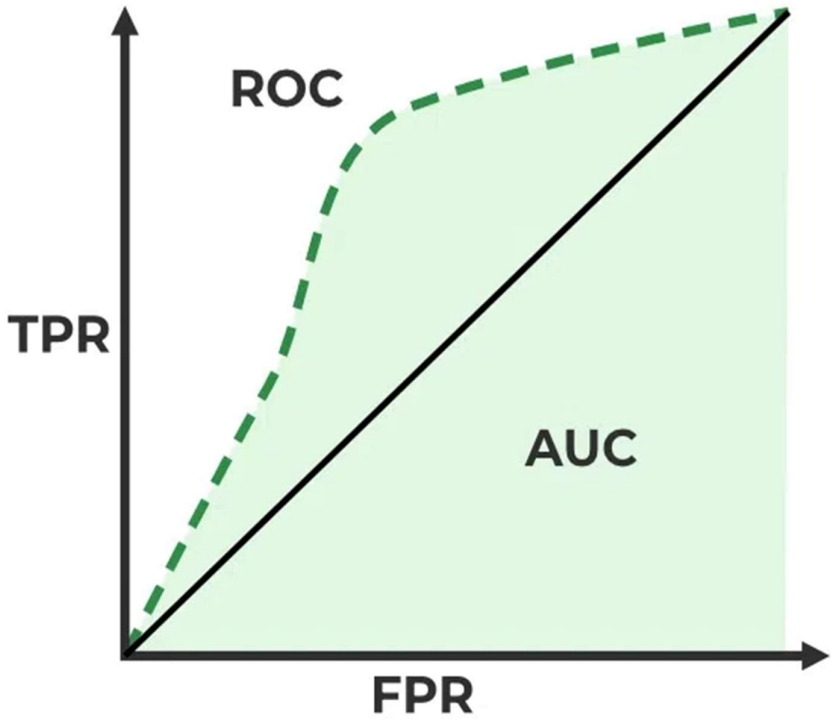
1. **General Parameters**— Καθοδηγούν τη συνολική λειτουργία του μοντέλου και σχετίζονται με τον ενισχυτή που χρησιμοποιείται.
2. **Booster Parameters** — Εξαρτάται από την επιλογή του ενισχυτή.
3. **Learning Task Parameters**— χρησιμοποιούνται για τον καθορισμό του στόχου βελτιστοποίησης,.
4. **Command Line Parameters**— Απαιτούνται για την έκδοση γραμμής εντολών του XGBoost. (developers s.-l. , 2023)

2.7.22 The Area Under Curve (AUC)

Η καμπύλη AUC αντιπροσωπεύει την περιοχή κάτω από την καμπύλη ROC. Μετρά τη συνολική απόδοση του μοντέλου δυαδικής ταξινόμησης. Καθώς τόσο το True Positive Rate (TPR) όσο και το False Positive Rate (FPR) κυμαίνονται μεταξύ 0 και 1, έτσι, η περιοχή θα βρίσκεται πάντα μεταξύ 0 και 1, και μια μεγαλύτερη τιμή AUC υποδηλώνει καλύτερη απόδοση του μοντέλου.

Ο κύριος στόχος είναι να μεγιστοποιήσουμε αυτήν την περιοχή ώστε να έχουμε το υψηλότερο TPR και το χαμηλότερο FPR στο δεδομένο όριο. Η AUC μετρά την πιθανότητα το μοντέλο να εκχωρήσει σε μια τυχαία επιλεγμένη θετική περίπτωση μια υψηλότερη προβλεπόμενη πιθανότητα σε σύγκριση με μια τυχαία επιλεγμένη αρνητική περίπτωση.

Αντιπροσωπεύει την πιθανότητα το μοντέλο μας, να είναι σε θέση να διακρίνει μεταξύ των δύο κλάσεων που υπάρχουν στον στόχο μας. (geeksforgeeks, 2023)



Σχήμα 9 AUC Curve

2.7.23 SHapley Additive exPlanations (SHAP)

Οι εταιρείες προσπαθούν να κάνουν τα μοντέλα μηχανικής εκμάθησης τους πιο διαφανή και κατανοητά.

Ένα από τα πιο αποτελεσματικά εργαλεία για αυτή τη διαδικασία είναι οι τιμές SHAP, οι οποίες μετρούν πόσο συνεισφέρει κάθε χαρακτηριστικό (όπως εισόδημα, ηλικία, πιστωτική βαθμολογία κ.λπ.) στην πρόβλεψη του μοντέλου. Οι τιμές SHAP μπορούν να βοηθήσουν να δούμε ποια χαρακτηριστικά είναι πιο σημαντικά για το μοντέλο και πώς επηρεάζουν το αποτέλεσμα. (Awan, 2023).

Στην παρούσα εργασία θα κάνουμε χρήση του εργαλείου SHAP στο μοντέλο με τα καλύτερα αποτελέσματα μετρικών αξιολόγησης.

3. Μεθοδολογία

3.1. Συλλογή δεδομένων

Η πηγή των δεδομένων μας, είναι η πλατφόρμα Kaggle (https://www.kaggle.com/datasets/rikdifos/credit-card-approval-prediction/data) (Kaggle, 2020). Μια πλατφόρμα δηλαδή, που δίνει τη δυνατότητα και σε άλλους προγραμματιστές να συμμετέχουν σε διαγωνισμούς μηχανικής εκμάθησης, να γράφουν και να μοιράζονται κώδικα και να φιλοξενούν σύνολα δεδομένων.

Την πλατφόρμα αυτή εμπιστεύονται μερικές από τις μεγαλύτερες εταιρείες επιστήμης δεδομένων στον κόσμο, όπως η Walmart, το Facebook και η Winton Capital.

3.2 Εξερεύνηση δεδομένων

Αρχικά, εισάγουμε τα δεδομένα μας από το Kaggle, μέσω της εφαρμογής Jupyter, Notebook, η οποία είναι μια εφαρμογή διακομιστή-πελάτη που επιτρέπει την επεξεργασία και την εκτέλεση εγγράφων-σημειωματάριων, μέσω ενός προγράμματος περιήγησης ιστού. Η εφαρμογή, μπορεί να εκτελεστεί σε μια τοπική επιφάνεια εργασίας που δεν απαιτεί πρόσβαση στο Διαδίκτυο ή μπορεί να εγκατασταθεί σε έναν απομακρυσμένο διακομιστή και να έχει πρόσβαση μέσω Διαδικτύου (contributors, 2015).

Τα δύο αρχεία των δεδομένων μας που εισάγονται στο Jupyter Notebook, υποβάλλονται σε διάφορα στάδια επεξεργασίας, προκειμένου να οριστικοποιηθούν σε τέτοια μορφή, ώστε να είναι έτοιμα για την εισαγωγή τους στη μηχανική μάθηση.

Έπειτα προς το τέλος της εξερεύνησης των δεδομένων (exploration) και πριν την επεξεργασία τους και την εισαγωγή τους στον αλγόριθμο της μηχανικής μάθησης, έχουμε συγχωνεύσει τα δύο αρχεία -πίνακες σε έναν και εξάγουμε το νέο αρχείο .csv, από το περιβάλλον του Jupyter Notebook και το εισάγουμε στην εφαρμογή Tableau, ώστε μέσα από τη δημιουργία ενός πλήθους οπτικοποιήσεων, να βοηθηθούμε στη λήψη αποφάσεων. Έτσι, θα μπορέσουμε να δούμε τα χαρακτηριστικά των αιτούντων για την έκδοση πιστωτικής κάρτας και μέσω αυτών θα έχουμε μια πιο πλήρη εικόνα για τη λήψη της απόφασης, αν θα εκδοθεί ή όχι.

3.3 Επεξεργασία δεδομένων

Στη συνέχεια, το νέο σύνολο δεδομένων, που προέκυψε από την εξερεύνηση και τη συγχώνευση τους, μετά την κατάλληλη επεξεργασία του, θα εισαχθεί στον αλγόριθμο μηχανικής μάθησης, όπου αυτό είναι και το τελευταίο στάδιο της μεθοδολογίας.

Επίσης, αναλύουμε τις τιμές των γνωρισμάτων του dataset, ενώ ταυτόχρονα δημιουργούμε διαγράμματα για την καλύτερη κατανόησή τους, αφαιρώντας τις ακραίες τιμές τους.

Έπειτα, προβαίνουμε στο μετασχηματισμό των τιμών των στηλών σε μορφή που η μηχανική μάθηση μπορεί να αναγνωρίσει και να επεξεργαστεί, ενώ στη συνέχεια αφαιρούνται οι τιμές null από το νέο dataset.

Μετά την ανάλυση, αφαιρούνται πολλές στήλες οι οποίες δεν είναι χρήσιμες ή θα προκαλέσουν σύγχυση στον αλγόριθμο, κατά την εισαγωγή τους στο μοντέλο της μηχανικής μάθησης.

Για το σκοπό αυτό αναφέρονται ενδεικτικά, οι βιβλιοθήκες της γλώσσας python, που χρησιμοποιήθηκαν σε αυτή τη διαδικασία:

- **Pandas :** χειρίζεται δεδομένα σε μορφή Dataframe. (Activate State, 2020) Διενεργήσαμε εργασίες όπως:
    1. Καθαρισμός δεδομένων
    2. Κανονικοποίηση δεδομένων
    3. Συγχώνευση δεδομένων
    4. Οπτικοποίηση δεδομένων
    5. Στατιστική ανάλυση
    6. Επιθεώρηση δεδομένων
    7. Φόρτωση και αποθήκευση δεδομένων
- **Numpy :** είναι το βασικό πακέτο για επιστημονικούς υπολογισμούς στην Python (NumPy, n.d.). Υπολογίσαμε αθροίσματα τιμών γνωρισμάτων κ.α. .
- **Matplotlib :** είναι μια ολοκληρωμένη βιβλιοθήκη για τη δημιουργία στατικών, κινούμενων και διαδραστικών απεικονίσεων στην Python . (John Hunter, 2012). Δημιουργήσαμε διαγράμματα σημαντικών γνωρισμάτων του dataset κατά κατηγορίες, του γνωρίσματος STATUS.
- **Seaborn :** είναι μια βιβλιοθήκη για τη δημιουργία στατιστικών γραφικών στην Python. (Waskom, 2023). Απεικονίσαμε το συνολικό εισόδημα και το πλήθος αυτών, σε πίνακα pairplot τα γνωρίσματα συνολικό εισόδημα, αριθμός παιδιών και μελών της οικογένειας, ημέρες από την γέννηση και ημέρες εργασίας.
- **Plotly :** είναι μια βιβλιοθήκη γραφημάτων που δημιουργεί διαδραστικά

γραφήματα ποιότητας δημοσίευσης (Plotly, 2020). Δημιουργήσαμε και εμφανίσαμε διαγράμματά των γνωρισμάτων του dataset.

- **Sklearn :** είναι μια από τις πιο ισχυρές βιβλιοθήκες για μηχανική μάθηση στην Python. Παρέχει μια σειρά εργαλείων για μηχανική μάθηση και στατιστική μοντελοποίηση, συμπεριλαμβανομένης της μείωσης διαστάσεων, της ομαδοποίησης, της παλινδρόμησης και της ταξινόμησης. Επιπλέον, παρέχει πολλά άλλα εργαλεία για αξιολόγηση, επιλογή, ανάπτυξη μοντέλων και προεπεξεργασία δεδομένων. (Simpliearn, 2022). Θα κάνουμε χρήση της συνάρτησης LabelEncoder της βιβλιοθήκης scikitlearn, για τη μετατροπή στηλών από μη αριθμητικά δεδομένα σε αριθμητικά. Από την ίδια βιβλιοθήκη θα χρησιμοποιήσουμε τις μετρικές αξιολόγησης όπως η f1-score, classification report και το διάγραμμα confusion matrix.

3.4 Δημιουργία μοντέλων

Στο στάδιο αυτό χρησιμοποιούνται πολλές τεχνικές της μηχανικής μάθησης, του ζητήματος της ταξινόμησης, των μοντέλων εκπαίδευσης που δημιουργούνται και των μετρικών αξιολόγησης .

Μετά το πέρας της διαδικασίας της επεξεργασίας των δεδομένων, δημιουργείται το τελικό dataset με το οποίο θα ξεκινήσουμε τη διαδικασία της μηχανικής μάθησης. Έτσι το χωρίζουμε σε δύο μέρη X και y, όπου το τελευταίο είναι η εξαρτημένη μεταβλητή που θα γίνει η πρόβλεψη και το πρώτο οι ανεξάρτητες μεταβλητές, ενώ μέσω της εντολής split, διαχωρίζουμε τα δεδομένα μας σε train και σε test, σε 70% (train) και 30% (test).

Ακολούθως, με τη μέθοδο SMOTE προχωράμε στην εκπαίδευση 3 διαφορετικών μοντέλων, ώστε στο τέλος να προβούμε στην σύγκριση των αποτελεσμάτων τους

Σε αυτό το σημείο άξιο αναφοράς, είναι οι βιβλιοθήκες Python, που χρησιμοποιήσαμε προκειμένου να εφαρμόσουμε την μέθοδο SMOTE.

- **Imblearn :** Το Imbalanced-learn είναι ένα πακέτο που χρησιμοποιείται για το χειρισμό μη ισορροπημένων συνόλων δεδομένων στη μηχανική μάθηση . (Kariuki, 2021)

Ενώ, ένα από τα μοντέλα που δημιουργούνται θα αξιολογηθεί μέσω του αλγόριθμου XGboost.

3.5 Αξιολόγηση μοντέλου πρόβλεψης

Στην αρχή δημιουργήθηκαν και παρουσιάστηκαν οπτικοποιήσεις τόσο στην Python μέσω της πλατφόρμας Kaggle, όσο και με την εισαγωγή των δεδομένων μας στην εφαρμογή Tableau. Τα συμπεράσματα που εξάγονται από αυτές, οδηγούν στον προσδιορισμό των σχέσεων περί έγκρισης ή μη της αίτησης για την έκδοση της πιστωτικής κάρτας από το χρηματοπιστωτικό ίδρυμα.

Επίσης, εμφανίζονται τα αποτελέσματα των μοντέλων που έχουν εκπαιδευτεί και γίνεται σύγκριση μεταξύ τους, για την καλύτερη πρόβλεψη σύμφωνα με τις με τις μετρικές αξιολόγησης f1 – score, precision, recall, Accuracy και AUC που εμφανίζονται.

Στην παρούσα εργασία θα χρησιμοποιήσουμε την βιβλιοθήκη SHAP κάνοντας χρήση της οπτικοποίησης, SHAP Summary Plot, μέσω της οποίας θα δούμε την σημασία/αξία του κάθε χαρακτηριστικού, για το μοντέλο που έχουμε φτιάξει και τις προβλέψεις του, μέσω χρωματικών εφέ αλλά και σε σύγκριση με τις υπόλοιπες μεταβλητές.

4. Υλοποίηση

4.1 Συλλογή δεδομένων

Στο κεφάλαιο γίνεται αναλυτική περιγραφή των ενεργειών που εκτελούνται με σκοπό την ολοκλήρωση της εργασίας. Επίσης, γίνεται μία βιβλιογραφική αναφορά τεχνολογιών που χρησιμοποιούνται με στόχο την καλύτερη κατανόηση του αναγνώστη.

Το κύριο σύνολο δεδομένων που επιλέγεται - εισάγεται για την ανάλυση ονομάζεται “credit-card-approval-prediction” (https://www.kaggle.com/datasets/rikdifos/credit-card-approval-prediction/data).

Για την ανάλυση των δεδομένων γίνεται εγκατάσταση των βιβλιοθηκών της γλώσσας Python, προκειμένου να γίνει χρήση των κατάλληλων συναρτήσεων της.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import itertools
import warnings
warnings.filterwarnings('ignore')
```

Το σύνολο των δεδομένων αποτελείται από δύο αρχεία. Τα “application_record” και “credit_record”, τα οποία εισάγουμε στο Jupyter Notebook με τη χρήση της συνάρτησης .read_csv(), εμπεριέχουσα στην βιβλιοθήκη Pandas.

```
application=pd.read_csv('/kaggle/input/credit-card/application_record.csv')
credit=pd.read_csv('/kaggle/input/credit-card/credit_record.csv')
```

Το αρχείο application_record είναι διαστάσεων 438557 x 18 columns ενώ το credit_record είναι 1048575 x 3 columns αντίστοιχα.

4.2 Εξερεύνηση δεδομένων

Όπως προαναφέραμε, τα δεδομένα μας, αποτελούνται από δύο αρχεία με κατάληξη .csv (comma separated values) (Ltd., 2023), ως ακολούθως:

- ✓ application_record.csv
- ✓ credit_record.csv

Τα δεδομένα που περιέχονται στα ανωτέρω αρχεία, κατά την πλειοψηφία είναι εξίσου σημαντικά και μπορούν να λειτουργήσουν, μέσω της μηχανικής μάθησης, θετικά για την λήψη της τελικής απόφασης.

Ακολουθεί μία σύντομη περιγραφή για το καθένα από τα αρχεία .

4.2.1 Αρχείο application_record.csv

Η τράπεζα σε κάθε αίτηση για έκδοση πιστωτικής κάρτας, δίνει έναν μοναδικό κωδικό πελάτη αλλά καταχωρεί και τα δημογραφικά στοιχεία του, όπως είναι το φύλο του και η ηλικία του .

Όνομα γνωρίσματος	***Επεξήγηση***	***Τιμές***
ID	Κωδικός πελάτη	Αύξοντες αριθμοί από το σύστημα της τράπεζας
CODE_GENDER	Φύλο πελάτη	M (Male): Άνδρας F (Female): Γυναίκα
DAYS_BIRTH	Ηλικία σε ημέρες	Μετράει αντίστροφα από την τρέχουσα ημέρα (0), -1 σημαίνει χθες

Πίνακας 2 Στοιχεία πελάτη (μοναδικός κωδικός πελάτη και δημογραφικά στοιχεία)

Επιπλέον, η τράπεζα θέλει να γνωρίζει τα περιουσιακά στοιχεία του πελάτη. Δηλαδή, αν έχει κάποιο σπίτι στην ιδιοκτησία του ή κάποιο αυτοκίνητο.

Όνομα γνωρίσματος	***Επεξήγηση***	***Τιμές***
NAME_HOUSING_TYPE	Ποια είναι η στεγαστική κατάσταση του πελάτη (νοικιάζει, συμβιώνει με γονείς κ.ά.).	-Σπίτι / Διαμέρισμα -Με τους γονείς -Άλλο
FLAG_OWN_CAR	Αν υπάρχει αμάξι	N (No): Όχι Y (Yes): Ναι
FLAG_OWN_REALTY	Αν υπάρχει ιδιοκτησία	N (No): Όχι Y (Yes): Ναι

Πίνακας 3 Στοιχεία πελάτη (περιουσιακά στοιχεία)

Επίσης, την τράπεζα ενδιαφέρει η οικογενειακή κατάσταση του πελάτη, αν έχει οικογένεια, πόσα είναι τα μέλη της, ο αριθμός των παιδιών του, αν υπάρχουν.

Όνομα γνωρίσματος	***Επεξήγηση***	***Τιμές***
NAME_FAMILY_STATUS	Οικογενειακή κατάσταση	-Παντρεμένος -Εργένης / Όχι παντρεμένος -Άλλο

CNT_FAM_MEMBERS	Μέλη οικογένειας	Δεκαδικοί αριθμοί
CNT_CHILDREN	Αριθμός παιδιών	Ακέραιοι αριθμοί

Πίνακας 4 Στοιχεία πελάτη (οικογενειακή κατάσταση)

Σημαντικό κομμάτι στην αίτηση για έκδοση πιστωτικής κάρτας, είναι η καταγραφή από την τράπεζα της επαγγελματικής και οικονομικής κατάστασης του πελάτη. Συγκεκριμένα, το ετήσιο εισόδημα του, η κατηγορία του εισοδήματος του, το είδος της εργασίας στην οποία απασχολείται και το χρονικό διάστημα απασχόλησης του.

Όνομα γνωρίσματος	***Επεξήγηση***	***Τιμές***
AMT_INCOME_TOTAL	Ετήσιο εισόδημα	Δεκαδικοί αριθμοί
OCCUPATION_TYPE	Είδος εργασίας	-Εργάτης -Null -Άλλο
DAYS_EMPLOYED	Ημερομηνία έναρξης απασχόλησης	Μετράει αντίστροφα από την τρέχουσα ημέρα(0). Εάν είναι θετικό, σημαίνει ότι το άτομο είναι σήμερα άνεργο.
NAME_INCOME_TYPE	Κατηγορία εισοδήματος	-Εργαζόμενος -Εμπορικός συνεργάτης -Άλλο

Πίνακας 5 Στοιχεία πελάτη (επαγγελματική και οικονομική κατάσταση)

Επιπλέον, η τράπεζα ζητά από τον πελάτη να αναφέρει, αν υπάρχουν, τρόποι επικοινωνίας μαζί του, όπως είναι για παράδειγμα, αν έχει αριθμό κινητού τηλεφώνου, αριθμό τηλεφώνου στην οικία του, διεύθυνση ηλεκτρονικού ταχυδρομείου και το τηλέφωνο της εργασίας του.

Όνομα γνωρίσματος	***Επεξήγηση***	***Τιμές***
FLAG_MOBIL	Υπάρχει κινητό τηλέφωνο	0: Όχι 1: Ναι

FLAG_PHONE	Υπάρχει τηλέφωνο	0: Όχι 1: Ναι
FLAG_EMAIL	Υπάρχει email	0: Όχι 1: Ναι
FLAG_WORK_PHONE	Υπάρχει τηλέφωνο εργασίας	0: Όχι 1: Ναι

Πίνακας 6 Στοιχεία πελάτη (Στοιχεία επικοινωνίας)

Τέλος, η τράπεζα ενδιαφέρεται έστω και στατιστικά για το μορφωτικό επίπεδο του πελάτη.

Όνομα γνωρίσματος	***Επεξήγηση***	***Τιμές***
NAME_EDUCATION_TYPE	Επίπεδο μόρφωσης	-Δευτεροβάθμια / Δευτεροβάθμια Ειδική -Ανώτατη εκπαίδευση -Άλλο

Πίνακας 7 Στοιχεία πελάτη (μορφωτικό επίπεδο)

4.2.2 Αρχείο credit_record.csv

Στο αρχείο credit_record.csv παρέχονται πληροφορίες για μηνιαία υπόλοιπα προηγούμενων δανείων που αναφέρθηκαν στο Γραφείο Πιστώσεων. Για κάθε δάνειο στα τρέχοντα σύνολα δεδομένων, υπάρχουν τόσες σειρές όσοι και οι μήνες για τους οποίους υπάρχει ιστορικό.

Όνομα γνωρίσματος	***Επεξήγηση***
ID	Κωδικός πελάτη
MONTHS_BALANCE	Ο μήνας των εξαγόμενων δεδομένων είναι το σημείο εκκίνησης, προς τα πίσω, 0 είναι ο τρέχων μήνας, -1 είναι ο προηγούμενος μήνας και ούτω καθεξής

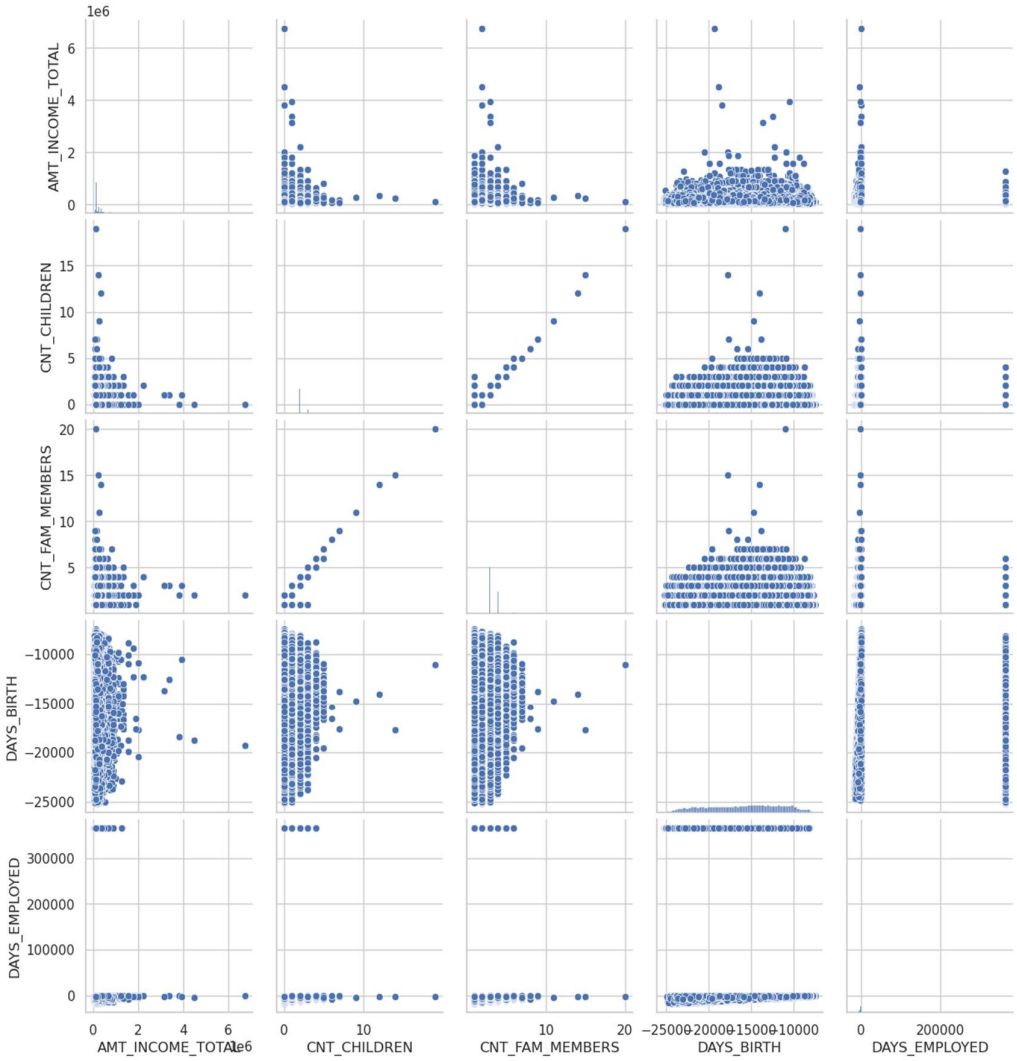
STATUS	0: 1-29 ημέρες καθυστέρηση 1: 30-59 ημέρες καθυστέρηση 2: 60-89 ημέρες καθυστέρηση 3: 90-119 ημέρες καθυστέρηση 4: 120-149 ημέρες καθυστέρηση 5:ληξιπρόθεσμες ή επισφαλείς οφειλές, διαγραφές για περισσότερο από 150 ημέρες C: εξοφλήθηκε εκείνο το μήνα X: Χωρίς δάνειο για τον μήνα

Πίνακας 8 Στοιχεία πελάτη (Ιστορικό δόσεων προηγούμενων δανείων)

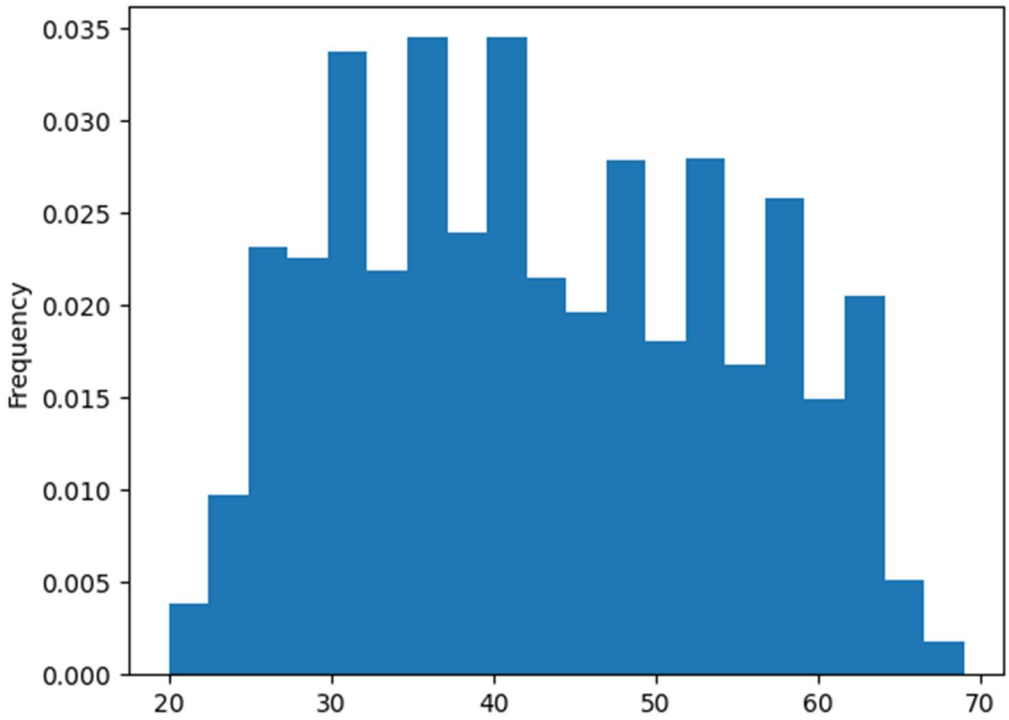
Στην επόμενη ενότητα του κεφαλαίου, επιτυγχάνεται η εξερεύνηση των κύριων μεταβλητών μέσα από οπτικοποιήσεις, που συμβάλει στην καλύτερη κατανόηση και εμπέδωση των μεταβλητών του συνόλου δεδομένων. Η διαδικασία αυτή γίνεται με τη χρήση του περιβάλλοντος Jupyter Notebook και της python, όπου αρχικά εισάγονται όλα τα δεδομένα.

4.2.3 Οπτικοποιήσεις αρχικού dataset

Στην παρακάτω εικόνα εμφανίζεται η συσχέτιση των δεδομένων – μεταβλητών σύνολο εισοδήματος, αριθμός παιδιών, αριθμός μελών οικογενείας, ημέρες από γέννηση του πελάτη και ημέρες από την πρόσληψη στην εργασία του πελάτη. κατανομή των ποσών των δανείων που έχουν δοθεί. Λόγω μεγάλου πλήθους των αιτήσεων, στη γραφική παράσταση της κατανομής οι οπτικές ενδείξεις των δεδομένων, δεν μπορούν να μας βοηθήσουν στην εξαγωγή ασφαλούς συμπεράσματος.

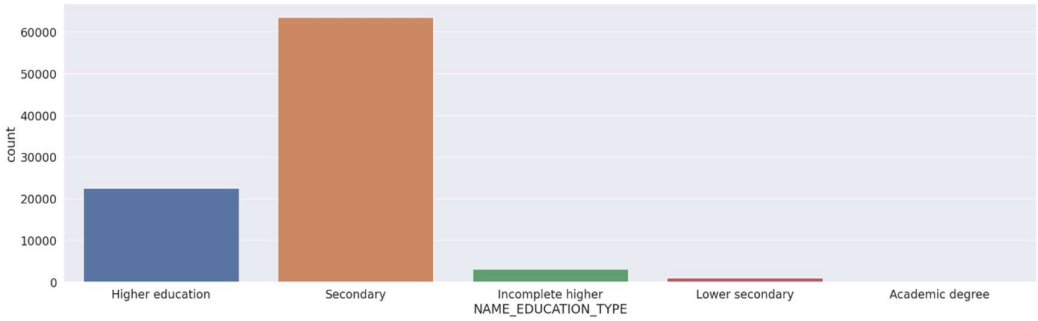


Σχήμα 10 Pairplot 5 μεταβλητών



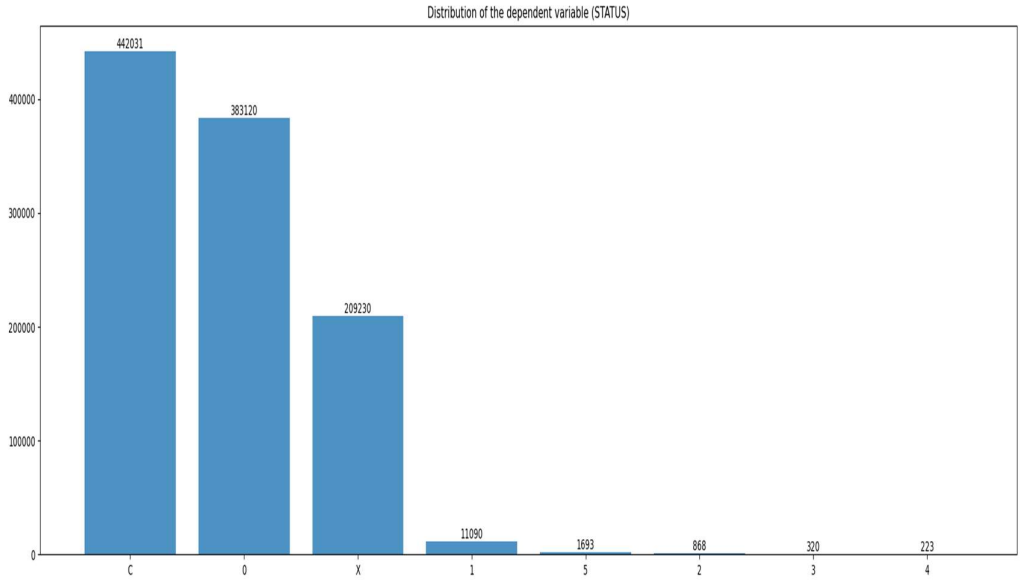
Σχήμα 11 Συχνότητα ηλικιακών ομάδων

Η παραπάνω οπτικοποίηση απεικονίζει την ηλικία των πελατών σε σχέση με τη συχνότητα εμφάνισης των ηλικιών. Διαπιστώνουμε, ότι μεγάλη συχνότητα εμφανίζουν οι ηλικίες που βρίσκονται στο διάστημα 30 – 40 και χαμηλότερη συχνότητα οι ηλικίες που βρίσκονται στο διάστημα 50-60. Παρόλο που υπάρχουν κατανομές, δεν υφίσταται ακρίβεια σε αυτήν, ώστε να μπορεί η τράπεζα να ανατρέξει σε συγκεκριμένη ηλικιακή ομάδα και να ερωτηθεί τη συχνότητα της συγκεκριμένης ηλικιακής ομάδας αλλά σε διάστημα αυτής.



Σχήμα 12 Αριθμός αιτούντων σε σχέση με το μορφωτικό επίπεδο

Σε αυτήν την οπτικοποίηση, η τράπεζα μπορεί να παρατηρήσει το πλήθος των ατόμων που αιτήθηκαν την έκδοση πιστωτικής κάρτας σε σχέση με τον τύπο της εκπαίδευσης / μόρφωσης τους. Το μεγαλύτερο αριθμό αιτούντων, αντιστοιχεί σε αυτούς που έχουν δευτεροβάθμια εκπαίδευση.



Σχήμα 13 Αριθμός αιτήσεων σε σχέση με το STATUS

Από το ανωτέρω διάγραμμα παρατηρούμε ότι το μέγιστο πλήθος αιτήσεων αφορά εκείνες που είχαν δάνειο και εξοφλήθηκε εκείνον τον μήνα ή δεν έχουν κανένα χρέος προς την τράπεζα. Ακολουθεί η κατηγορία εκείνη που αφορά τις αιτήσεις όπου η εξόφληση του χρέους τους έχει καθυστερήσει 1-29 ημέρες. X: Χωρίς δάνειο για τον μήνα

0: 1-29 ημέρες καθυστέρηση (αριθμός)

C: εξοφλήθηκε εκείνο το μήνα (αριθμός)

4.3 Επεξεργασία των δεδομένων μας

Εφόσον στην προηγούμενη ενότητα, έγινε η εισαγωγή των δυο αρχείων .csv(), application_record.csv και credit_record.csv, στο πρώτο στάδιο χρησιμοποιούμε για το 1ο αρχείο application_record η συνάρτηση .describe() που εμφανίζει την περιγραφή των δεδομένων που περιέχονται σε αυτό.

```
print(application.describe().T)
```

και μας εμφανίζει αναλυτικά τα στατιστικά στοιχεία του κυριότερου αρχείου του dataset.

	count	mean	std	min	25%	\
ID	438557.0	6.022176e+06	571637.023257	5008804.0	5609375.0	
CNT_CHILDREN	438557.0	4.273903e-01	0.724882	0.0	0.0	
AMT_INCOME_TOTAL	438557.0	1.875243e+05	110086.853066	26100.0	121500.0	
DAYS_BIRTH	438557.0	-1.599790e+04	4185.030007	-25201.0	-19483.0	
DAYS_EMPLOYED	438557.0	6.056368e+04	138767.799647	-17531.0	-3103.0	
FLAG_MOBIL	438557.0	1.000000e+00	0.000000	1.0	1.0	
FLAG_WORK_PHONE	438557.0	2.061328e-01	0.404527	0.0	0.0	
FLAG_PHONE	438557.0	2.877710e-01	0.452724	0.0	0.0	
FLAG_EMAIL	438557.0	1.082071e-01	0.310642	0.0	0.0	
CNT_FAM_MEMBERS	438557.0	2.194465e+00	0.897207	1.0	2.0	

	50%	75%	max
ID	6047745.0	6456971.0	7999952.0
CNT_CHILDREN	0.0	1.0	19.0
AMT_INCOME_TOTAL	160780.5	225000.0	6750000.0
DAYS_BIRTH	-15630.0	-12514.0	-7489.0
DAYS_EMPLOYED	-1467.0	-371.0	365243.0
FLAG_MOBIL	1.0	1.0	1.0
FLAG_WORK_PHONE	0.0	0.0	1.0
FLAG_PHONE	0.0	1.0	1.0
FLAG_EMAIL	0.0	0.0	1.0
CNT_FAM_MEMBERS	2.0	3.0	20.0

Ενώ, για το 2ο αρχείο χρησιμοποιούμε τη συνάρτηση .head() που εμφανίζει τις πρώτες πέντε σειρές του πίνακα και δίνει μια ολοκληρωμένη εικόνα των πληροφοριών που περιέχει η κάθε στήλη.

```
credit.head()
```

	ID	MONTHS_BALANCE	STATUS
0	5001711	0	X
1	5001711	-1	0
2	5001711	-2	0
3	5001711	-3	0
4	5001712	0	C

Στο δεύτερο στάδιο, για το αρχείο application_record καλείται η συνάρτηση .info(), η οποία εμφανίζει τις πληροφορίες των δεδομένων αλλά και τον τύπο τον δεδομένων.

```
application_info=application.info()
print(application_info)
print('\n')
```

Δηλαδή μας εμφανίζει πλήρη ανάλυση του αρχείου, ως προς τον τύπο των αρχείων και αν περιέχονται null τιμές και το πλήθος αυτών

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 438557 entries, 0 to 438556
Data columns (total 18 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   ID                   438557 non-null  int64
 1   CODE_GENDER          438557 non-null  object
 2   FLAG_OWN_CAR         438557 non-null  object
 3   FLAG_OWN_REALTY      438557 non-null  object
 4   CNT_CHILDREN         438557 non-null  int64
 5   AMT_INCOME_TOTAL     438557 non-null  float64
 6   NAME_INCOME_TYPE     438557 non-null  object
 7   NAME_EDUCATION_TYPE  438557 non-null  object
 8   NAME_FAMILY_STATUS   438557 non-null  object
 9   NAME_HOUSING_TYPE    438557 non-null  object
 10  DAYS_BIRTH           438557 non-null  int64
 11  DAYS_EMPLOYED        438557 non-null  int64
 12  FLAG_MOBIL           438557 non-null  int64
 13  FLAG_WORK_PHONE      438557 non-null  int64
 14  FLAG_PHONE           438557 non-null  int64
 15  FLAG_EMAIL           438557 non-null  int64
 16  OCCUPATION_TYPE      304354 non-null  object
 17  CNT_FAM_MEMBERS      438557 non-null  float64
dtypes: float64(2), int64(8), object(8)
memory usage: 60.2+ MB
None
```

Επιπλέον, και στα δύο αρχεία, γίνεται έλεγχος για null εγγραφές, το πλήθος αυτών και σε ποια στήλη περιέχονται αυτές.

```
application.isna().sum()
```

```
ID                          0
CODE_GENDER                 0
FLAG_OWN_CAR                0
FLAG_OWN_REALTY             0
CNT_CHILDREN                0
AMT_INCOME_TOTAL            0
NAME_INCOME_TYPE            0
NAME_EDUCATION_TYPE         0
NAME_FAMILY_STATUS          0
NAME_HOUSING_TYPE           0
DAYS_BIRTH                  0
DAYS_EMPLOYED               0
FLAG_MOBIL                  0
FLAG_WORK_PHONE             0
FLAG_PHONE                  0
FLAG_EMAIL                  0
OCCUPATION_TYPE        134203
CNT_FAM_MEMBERS             0
dtype: int64
```

Παρατηρούμε ότι στο αρχείο application_record, περιέχονται 134.203 εγγραφές null στη στήλη Occupation_Type.

```
credit.isna().sum()
```

```
ID              0
MONTHS_BALANCE  0
STATUS          0
dtype: int64
```

Ενώ στο αρχείο credit_record παρατηρούμε ότι δεν υπάρχουν null τιμές.

Ενώ, ζητάμε για το αρχείο application_record, να ελέγξει για διπλότυπες εγγραφές και σε περίπτωση που διαπιστωθεί ότι υπάρχει να τις αφαιρέσει.

```
application = application.drop_duplicates(subset=application.columns[1:], keep='first')
```

4.3.1 Τροποποίηση των μεταβλητών σε κατανοητή μορφή

Υπάρχουν μεταβλητές στα αρχεία, που η μορφή τους δεν είναι κατανοητή ώστε να μπορούν να επεξεργαστούν στη μηχανική μάθηση.

Μία από αυτές είναι η μεταβλητή Age, στην οποία είναι καταχωρημένες οι ημέρες αλλά με αρνητική τιμή, καθόσον μετρούνται αντίστροφα. Σε αυτό το σημείο, θα εκτελέσουμε μία εντολή της γλώσσας, ώστε η μεταβλητή Age να μετατραπεί στην μορφή του θετικού ακέραιου.

```
application['Age']=-(application['DAYS_BIRTH'])//365
```

Δηλαδή, στην μεταβλητή καταχωρούμε την αρνητική τιμή της ίδιας μεταβλητής διαιρεμένη με τον αριθμό 365, που είναι ημέρες ενός έτους.

```
application['Age']
```

Έτσι, καλώντας τη στήλη Age μας δίνει τα επιθυμητά αποτελέσματα:

```
0         32
2         58
3         52
7         61
10        46
          ..
438541    37
438545    51
438547    30
438552    62
438553    43
Name: Age, Length: 90085, dtype: int64
```

Εικόνα 5 Περιεχόμενο διαμορφωμένης στήλης 'Age'

Επίσης, υπάρχουν μεταβλητές όπως η Code_Gender, η Flag_own_car και η Flag_own_Realty, οι οποίες περιέχουν γράμματα σαν απαντήσεις, 'F', 'M', Y' και 'N' αντίστοιχα. Όμως, οι τιμές αυτές θα πρέπει να μετατραπούν σε αριθμούς '0' και '1' ώστε να μπορούν να είναι επεξεργάσιμες από τον αλγόριθμο της μηχανικής μάθησης.

Έτσι, με την παρακάτω εντολή επιτυγχάνουμε αυτήν την μετατροπή.

```
application['CODE_GENDER'].replace('M',0,inplace=True)
application['CODE_GENDER'].replace('F',1,inplace=True)
application['FLAG_OWN_CAR'].replace('Y',0,inplace=True)
application['FLAG_OWN_CAR'].replace('N',1,inplace=True)
application['FLAG_OWN_REALTY'].replace('Y',0,inplace=True)
application['FLAG_OWN_REALTY'].replace('N',1,inplace=True)
```

Σε αυτό το σημείο εισάγουμε δύο μεθόδους τις οποίες θα καλέσουμε παρακάτω και στη στήλη 'NAME_HOUSE_TYPE' και στην 'NAME_EDUCATION_TYPE' κάποιες τιμές θα τις κάνει split για μεγαλύτερη ευκολία κατανόησης και επεξεργασίας. Συγκεκριμένα, στην πρώτη στήλη που αφορά τον τύπο του σπιτιού όταν θα συναντά την τιμή 'House / apartment' θα τη χωρίζει αυτή την τιμή. Ενώ, στη στήλη που αφορά τον τύπο μόρφωσης όταν θα συναντά την τιμή 'Secondary / secondary special' πάλι αυτήν θα την χωρίζει.

```
def get_appartment(x):
    if x == 'House / apartment' :
       x= x.split(' /')[0]
    return x

def get_ducational_type(x):
    if x == 'Secondary / secondary special' :
       x= x.split(' /')[0]
    return x
```

4.3.2 Νέες παράμετροι

Μελετώντας τα δεδομένα, από τα δύο .csv αρχεία είναι δυνατόν να προκύψουν νέες παράμετροι. Οι παράμετροι αυτές επεξηγούνται στη συνέχεια. Οι νέες παράμετροι θα δημιουργηθούν στο περιβάλλον Jupyter Notebook .

4.3.3 Περιεχόμενο στηλών

Προκειμένου να διαμορφώσουμε μία άποψη για τα δεδομένα σε συγκεκριμένες στήλες ζητάμε να μας εμφανίσει την καταμέτρηση των δεδομένων ανάλογα με την αντίστοιχη κατηγορία. Ζητάμε, δηλαδή στο αρχείο application_record, να μας εμφανίσει μέσω της εντολής application['NAME_HOUSING_TYPE'] .value_counts() τον αριθμό των δεδομένων σε κάθε κατηγορία μέσα στη συγκεκριμένη στήλη .

```
application["NAME_HOUSING_TYPE"].value_counts()
```

και μας εμφανίζει

```
House / apartment      80335
With parents            4156
Municipal apartment     3130
Rented apartment        1373
Office apartment         790
Co-op apartment          301
Name: NAME_HOUSING_TYPE, dtype: int64
```

Εικόνα 6 Πλήθος στοιχείων ανά κατηγορία τύπου κατοικίας

ενώ ταυτόχρονα καλούμε την ίδια εντολή στο αρχείο credit_record για να μας εμφανίσει το πλήθος (σε ποσοστό επί της %) από κάθε κατηγορία δεδομένων στη συγκεκριμένη στήλη 'STATUS'.

```
print("The exact percentage of approved and not approved credit cards:")
credit['STATUS'].value_counts(normalize=True)
```

όπου μας επιστρέφει τα ακόλουθα αποτελέσματα

```
The exact percentage of approved and not approved credit cards:
C    0.421554
0    0.365372
X    0.199537
1    0.010576
5    0.001615
2    0.000828
3    0.000305
4    0.000213
Name: STATUS, dtype: float64
```

Εικόνα 7 Ποσοστό ανά κατηγορία προηγούμενων αιτήσεων δανείων

4.3.4 Η παράμετρος STATUS

Στο αρχικό αρχείο credit_record.csv, η στήλη 'STATUS' περιέχει τιμές όχι μόνο αριθμητικές. Για αυτό θα πρέπει να ορίσουμε τις ήδη υπάρχουσες τιμές να έχουν τις νέες '0' και '1', ώστε να μπορεί η μηχανική μάθηση να επεξεργαστεί το αρχείο, για πιο έγκυρο αποτέλεσμα.

Οπότε, θα μετατρέψουμε τρεις τιμές σε τιμή μονάδος '1' και τις υπόλοιπες σε '0'. Όπου '1' ορίζουμε όσες αιτήσεις είναι εγκεκριμένες, δηλαδή δεν χρωστάνε οι αιτούντες, έχουν εξοφλήσει τον τρέχον μήνα ή έχουν καθυστερήσει έως 1 μήνα την πληρωμή, ενώ με την τιμή '0' ορίζουμε όσες αιτήσεις μάλλον θα απορριφθούν.

```
new_status = {'C' : 1,
              'X' : 1,
              '0' : 1,
              '1' : 0,
              '2' : 0,
              '3' : 0,
              '4' : 0,
              '5' : 0}
credit['STATUS'] = credit['STATUS'].map(new_status)
credit['STATUS']=credit['STATUS'].astype(int)
```

Όλες αυτές τις νέες τιμές τις καταχωρούμε σε μία νέα μεταβλητή new_status.

Μετά με την συνάρτηση :

```
credit['STATUS'].value_counts()
```

θα μας εμφανίσει το πλήθος της στήλης ‘STATUS’ ανά τις τιμές ‘0’ και ‘1’.

```
1    1034381
0      14194
Name: STATUS, dtype: int64
```

Εικόνα 8 Πλήθος ανά κατηγορία (διαμορφωμένη) προηγούμενων αιτήσεων δανείων

Δηλαδή, η τιμή ‘1’ εμφανίζεται σε πολύ μεγαλύτερο βαθμό από την τιμή ‘0’.

Και με την μορφή ποσοστών.

```
credit['STATUS'].value_counts(normalize = True)
```

```
1    0.986464
0    0.013536
Name: STATUS, dtype: float64
```

Εικόνα 9 Πλήθος ανά κατηγορία (διαμορφωμένη) προηγούμενων αιτήσεων δανείων (σε ποσοστά)

Σχεδόν όλα τα στοιχεία εντός της στήλης περιέχουν την τιμή ‘1’.

4.3.5 Συγχώνευση των δύο αρχείων (merge)

Σε αυτό το σημείο, θα χρειαστεί να συγχωνέψουμε τα δύο αρχεία application_record και credit_record σε μία μεταβλητή που την ονομάζουμε final.

Αυτό το επιτυγχάνουμε με την εντολή merge

```
final = application.merge(credit, on=['ID'])
```

Δηλαδή ζητάμε να συγχωνέψει τον πίνακα application με τον πίνακα credit στη στήλη ‘ID’.

```
219173 rows × 21 columns
```

Είναι οι διαστάσεις του πίνακα final που προέκυψε από τη συγχώνευση. Ενώ στην αρχή ο κάθε πίνακας είχε διαστάσεις: application: 438557 x 18 columns και credit: 1048575 x 3 columns αντίστοιχα. Τελικά, καταφέραμε με τη συγχώνευση να καταλήξουμε σε 219173 x 21 columns, ώστε να μπορούμε να εισάγουμε ένα αρχείο κατάλληλο για τη διαδικασία της μηχανικής μάθησης

4.3.6 Έλεγχος για null τιμές στο final

Θα κάνουμε έναν έλεγχο στον τελικό πίνακα final για το αν περιέχονται null τιμές.

```
final.isnull().sum()
```

και τελικά διαπιστώνουμε ότι στη στήλη ‘OCCUPATION_TYPE’ περιέχονται 67954 null τιμές.

```
ID                          0
CODE_GENDER                 0
FLAG_OWN_CAR                0
FLAG_OWN_REALTY             0
CNT_CHILDREN                0
AMT_INCOME_TOTAL            0
NAME_INCOME_TYPE            0
NAME_EDUCATION_TYPE         0
NAME_FAMILY_STATUS          0
NAME_HOUSING_TYPE           0
DAYS_BIRTH                  0
DAYS_EMPLOYED               0
FLAG_MOBIL                  0
FLAG_WORK_PHONE             0
FLAG_PHONE                  0
FLAG_EMAIL                  0
OCCUPATION_TYPE         67954
CNT_FAM_MEMBERS             0
Age                         0
MONTHS_BALANCE              0
STATUS                      0
dtype: int64
```

Εικόνα 10 Πλήθος null τιμών στο πίνακα final

4.3.7 Ξεκαθάρισμα στηλών από null τιμές

Προκειμένου να εκκαθαρίσουμε τον πίνακα final από null τιμές, οι οποίες δεν βοηθούν στη διαδικασία της μηχανικής μάθησης, θα χρησιμοποιήσουμε την εντολή .dropna().

```
final = final.dropna(axis='columns')
```

Σε νέο έλεγχο του περιεχομένου (στηλών) του πίνακα final διαπιστώνουμε ότι δεν περιέχονται πλέον τιμές null .

```
final.isna().sum()
```

```
ID                      0
CODE_GENDER             0
FLAG_OWN_CAR            0
FLAG_OWN_REALTY         0
CNT_CHILDREN            0
AMT_INCOME_TOTAL        0
NAME_INCOME_TYPE        0
NAME_EDUCATION_TYPE     0
NAME_FAMILY_STATUS      0
NAME_HOUSING_TYPE       0
DAYS_BIRTH              0
DAYS_EMPLOYED           0
FLAG_MOBIL              0
FLAG_WORK_PHONE         0
FLAG_PHONE              0
FLAG_EMAIL              0
CNT_FAM_MEMBERS         0
Age                     0
MONTHS_BALANCE          0
STATUS                  0
dtype: int64
```

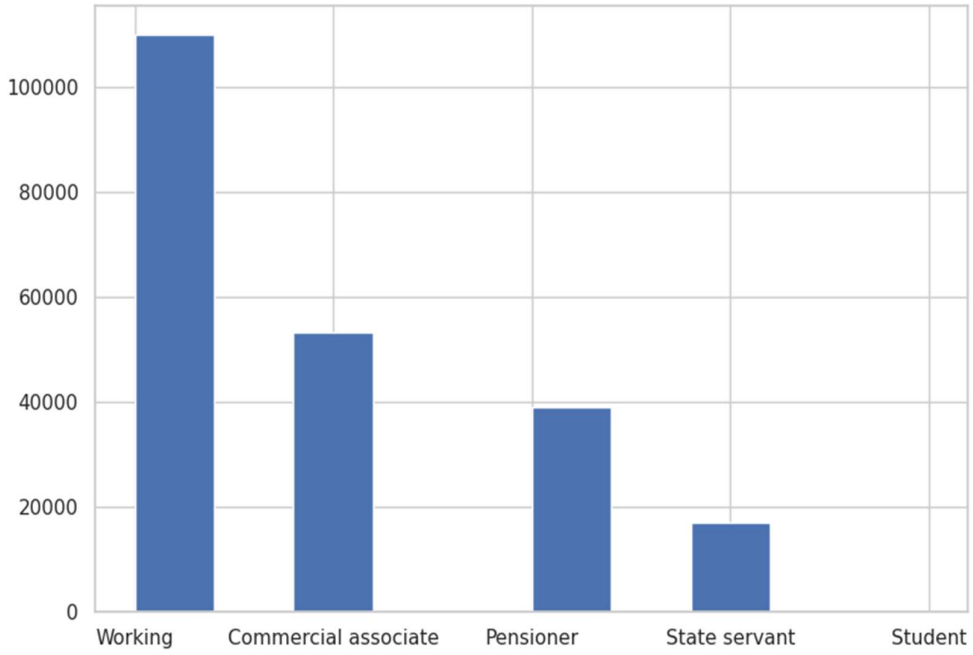
Εικόνα 11 Πλήθος null τιμών στο πίνακα final μετά την εκκαθάριση (dropna)

4.3.8 Οπτικοποιήσεις νέου dataset

4.3.8.1 Οπτικοποιήσεις (Python)

Παρακάτω θα παρουσιάσουμε σε ιστόγραμμα τα δεδομένα των δύο μεταβλητών

```
final['NAME_INCOME_TYPE'].hist(figsize=(9,6))
```



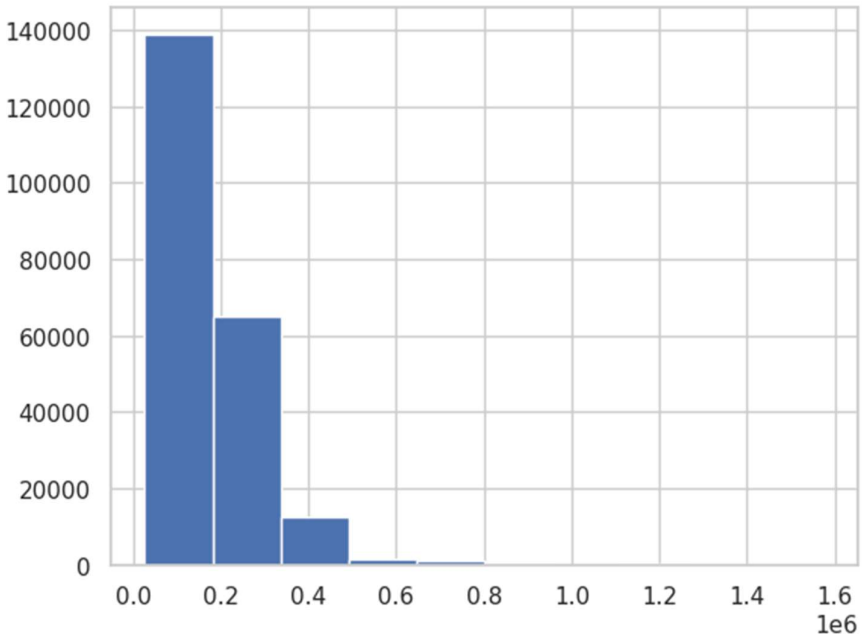
Σχήμα 14 Ιστόγραμμα πλήθους του τύπου εισοδήματος

Παρατηρούμε ότι το μεγαλύτερο μέρος των αιτούντων έχουν ως πηγή εισοδήματος την εργασία.

Ενώ, στο ιστόγραμμα που αφορά το σύνολο εισοδήματος του αιτούντα

```
final['AMT_INCOME_TOTAL'].hist()
```

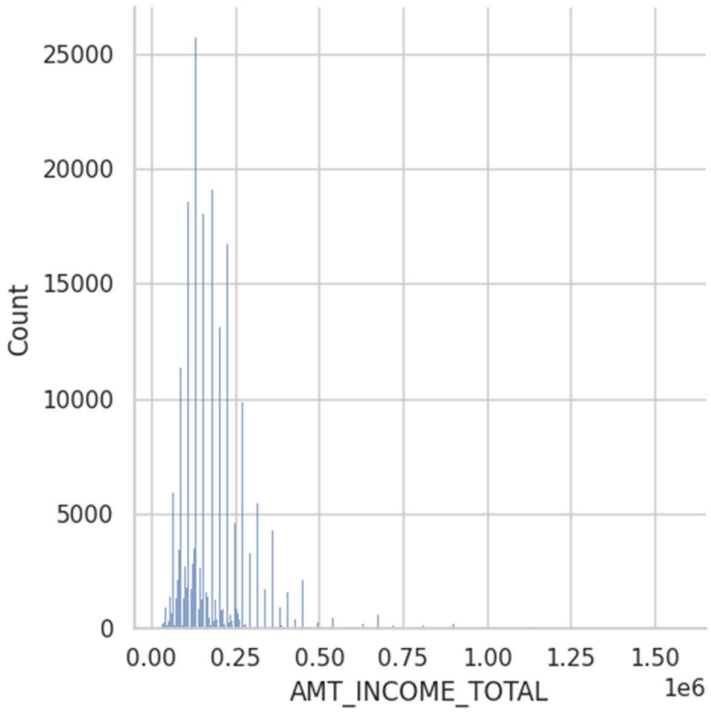
Παρατηρούμε, ότι ένα πολύ μικρό ποσοστό έχει υψηλό εισόδημα και η πλειοψηφία των αιτούντων το συνολικό εισόδημά τους δεν ξεπερνά τις 20.000 .



Σχήμα 15 Ιστόγραμμα πλήθους του συνολικού εισοδήματος

Επίσης, μέσω τη βιβλιοθήκη seaborn μπορούμε να απεικονίσουμε το συνολικό εισόδημα και το πλήθος αυτών, μέσω της εντολής

```
sns.displot(final, x="AMT_INCOME_TOTAL")
```



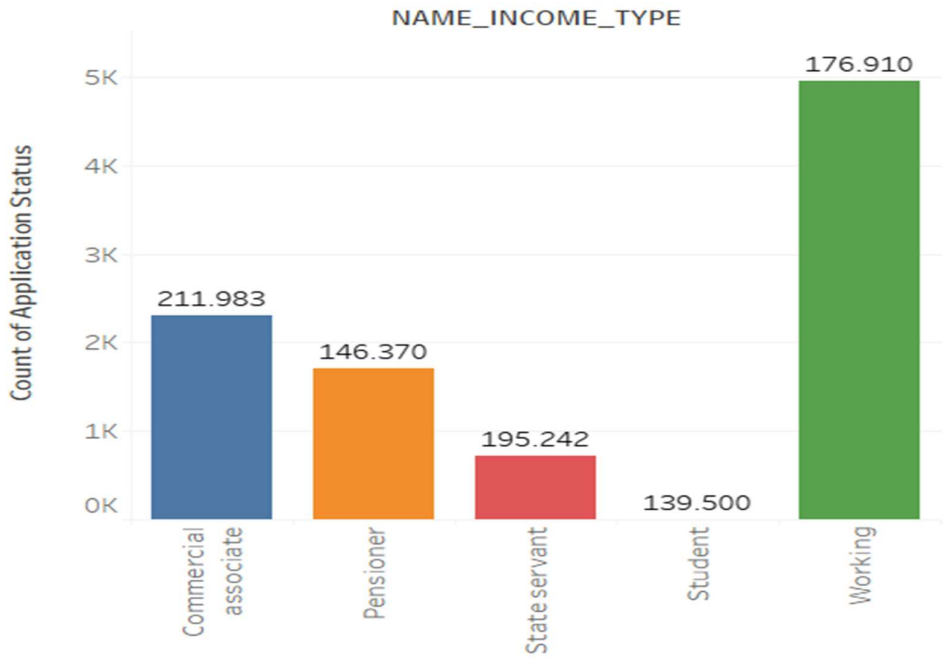
Σχήμα 16 Displot πλήθους του συνολικού εισοδήματος

Το μεγαλύτερο πλήθος συγκεντρώνεται μεταξύ 0,1 και 0,5.

4.3.8.2 Οπτικοποιήσεις (Tableau)

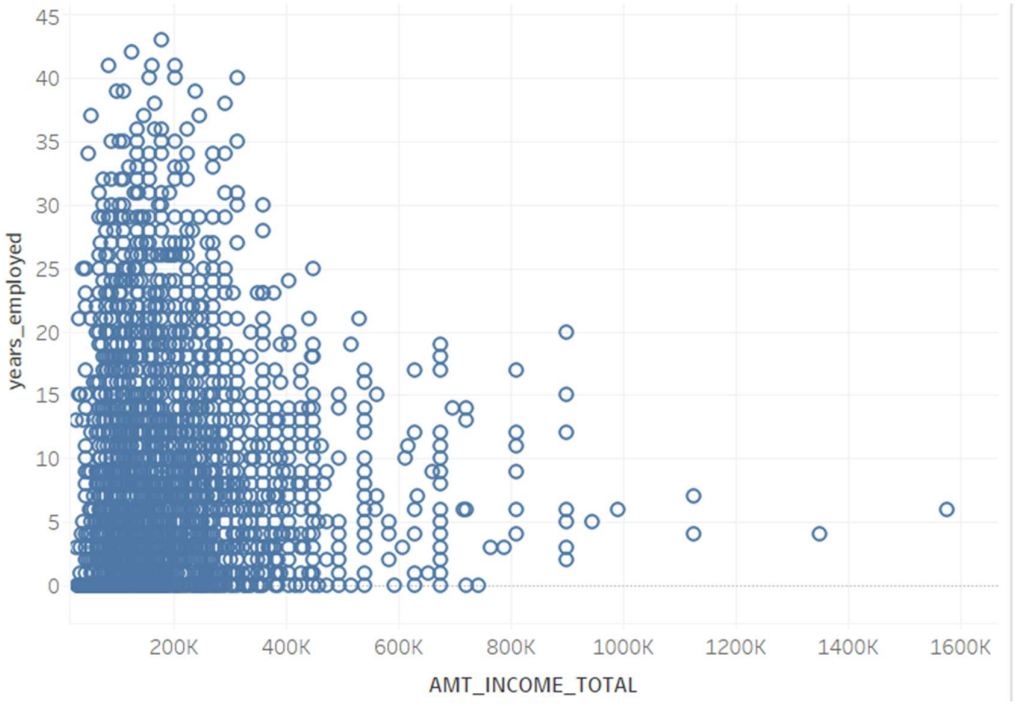
Σε αυτό το σημείο στο Jupyter Notebook, μετά την επεξεργασία των δεδομένων, τα εξάγουμε με την εντολή .to_csv() προκειμένου να το αποθηκεύσουμε στην επιφάνεια εργασίας έτσι, να είναι εύκολα προσβάσιμο από την εφαρμογή Tableau για περαιτέρω επεξεργασία για την ακριβέστερη οπτικοποίηση των δεδομένων μας.

```
final.to_csv(r'C:\Users\dppel\Desktop\final4.csv', index=False, header=True)
```



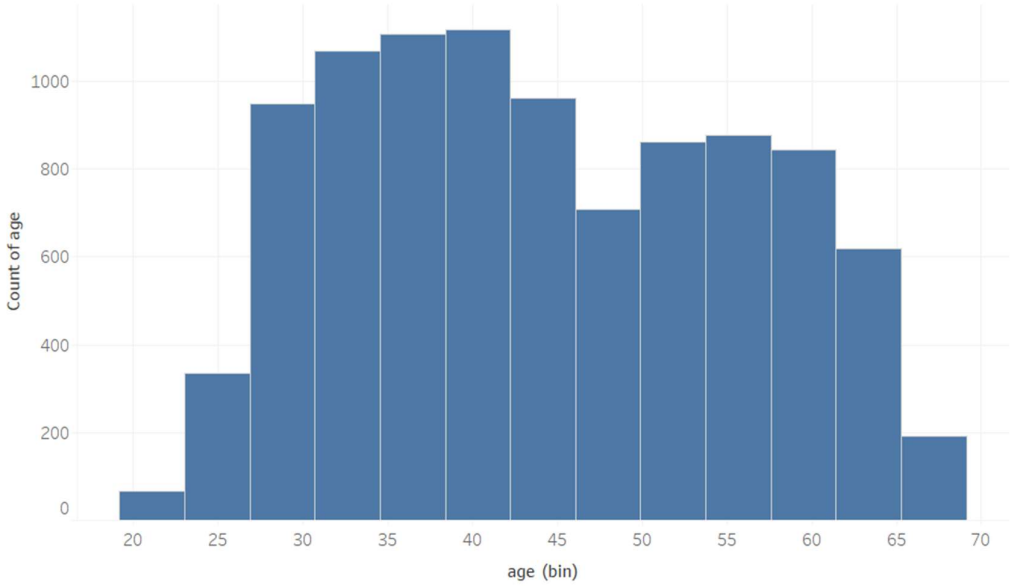
Σχήμα 17 Αριθμός αιτήσεων σε σχέση με τον τύπο εργασίας

Στην παραπάνω εικόνα, απεικονίζεται ο αριθμός των αιτήσεων σε σχέση με τον τύπο του εισοδήματος του αιτούντα. Τα νούμερα που φαίνονται απεικονίζουν το συνολικό εισόδημα του ατόμου. Παρατηρούμε, ότι οι εργαζόμενοι με μέσο όρο εισοδήματος 176.910 έχουν το μεγαλύτερο πλήθος αιτήσεων σε σχέση με άλλους με μεγαλύτερο ή μικρότερο μέσο όρο εισοδήματος.



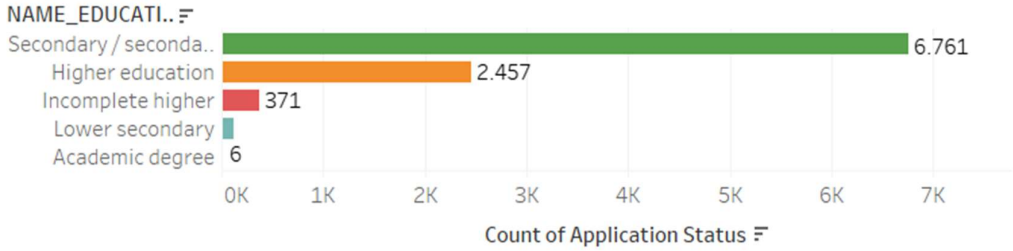
Σχήμα 18 Εισόδημα σε σχέση με τα χρόνια εργασίας

Σε αυτό το διάγραμμα, έχουμε απεικονίσει το εισόδημα των αιτούντων σε σχέση με τα χρόνια εργασίας του. Διαπιστώνουμε, ότι μεγάλο πλήθος εργαζομένων, έχουν μικρό εισόδημα γιατί προφανέστατα έχουν και λίγα χρόνια εργασίας.



Σχήμα 19 Πλήθος ηλικιακών ομάδων (bin)

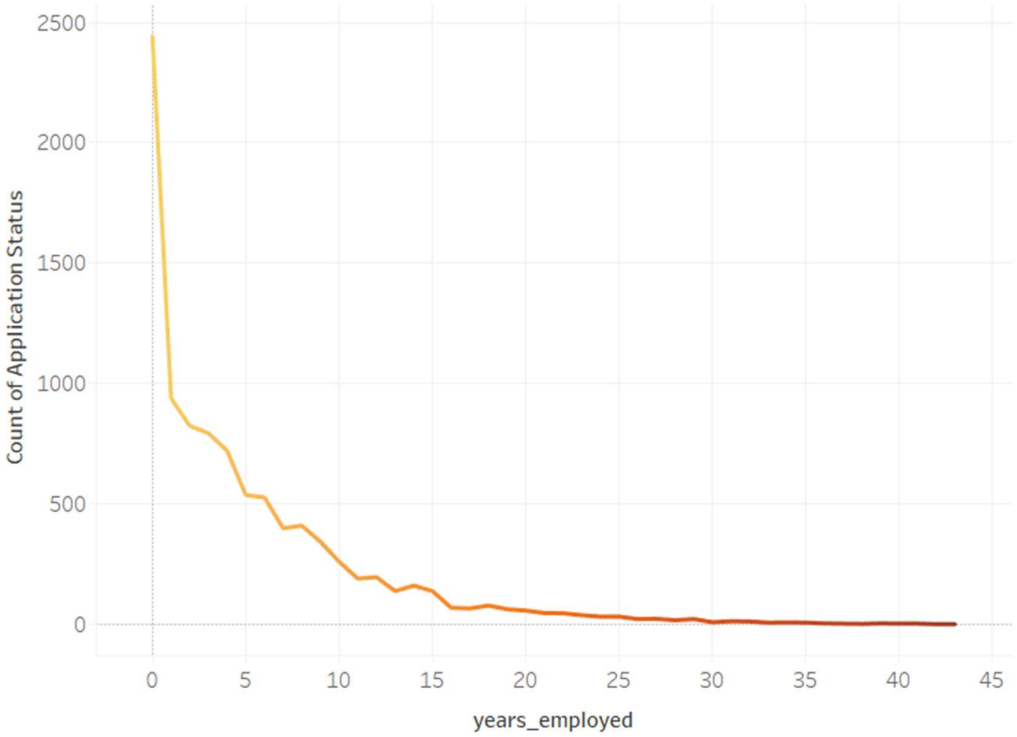
Δημιουργήσαμε bins της μεταβλητής Age όπου μετά από συσχέτιση τους με τον count(age) μας απεικόνισε το διάγραμμα αυτό. Το μεγαλύτερο εύρος της ηλικίας των ατόμων, κυμαίνεται μεταξύ 30-45 χρονών.



Σχήμα 20 Αριθμός αιτήσεων σε σχέση με τον τύπο μόρφωσης

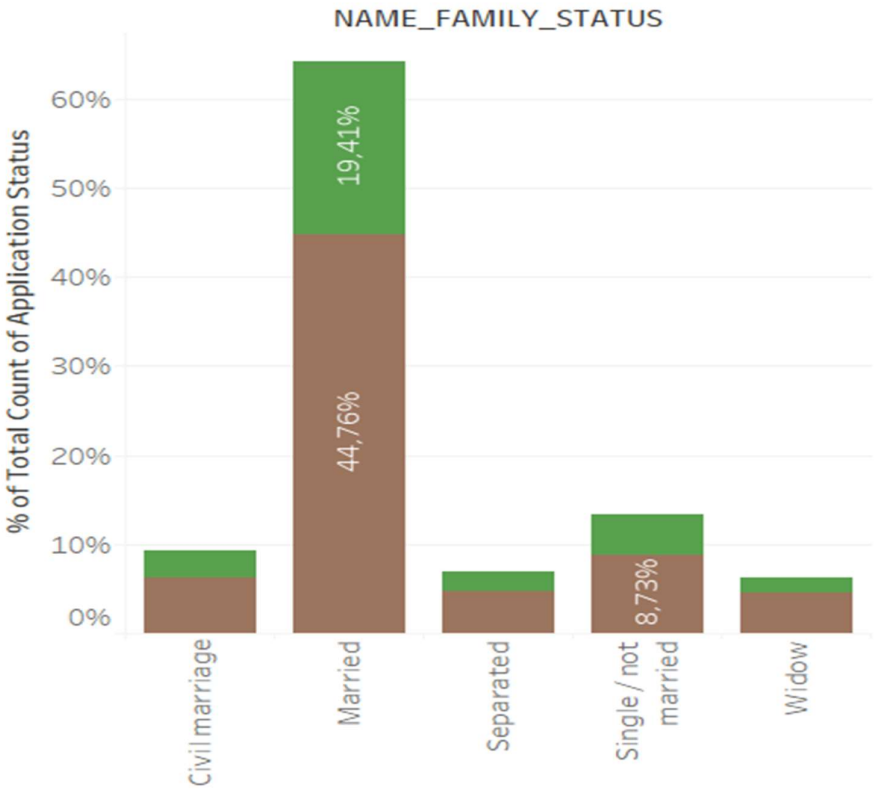
Με την κατάλληλη διαδικασία οδηγηθήκαμε στην ανωτέρω οπτικοποίηση, η οποία μας δείχνει τον αριθμό των αιτήσεων σε σχέση με το μορφωτικό επίπεδο του αιτούντα.

Το μεγαλύτερο πλήθος αιτήσεων, ήτοι 6.761, το συγκεντρώνουν τα άτομα που έχουν δευτεροβάθμια εκπαίδευση.



Σχήμα 21 Χρόνια εργασίας σε σχέση με αριθμό αιτήσεων

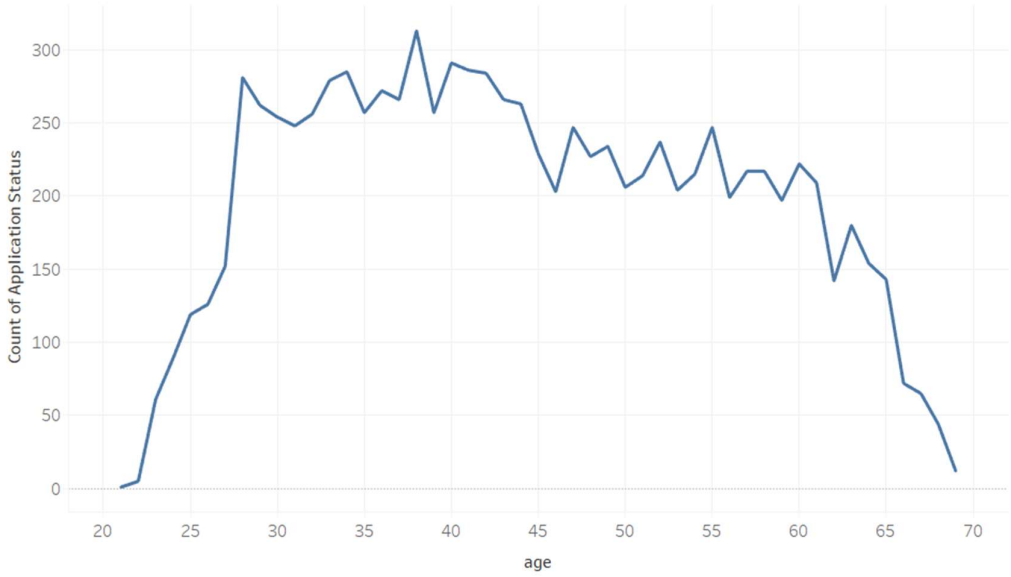
Σε αυτό το διάγραμμα συνδυάσαμε τα χρόνια εργασίας του αιτούντα σε σχέση με τον αριθμό των αιτήσεων και βγάλαμε το συμπέρασμα ότι των περισσότερο αριθμό αιτήσεων τον συγκεντρώνουν αυτοί που δεν έχουν χρόνια εργασίας.



Σχήμα 22 Ποσοστό αριθμού αιτήσεων σε σχέση με την οικογενειακή κατάσταση και την κατοχή ιδιοκτησίας

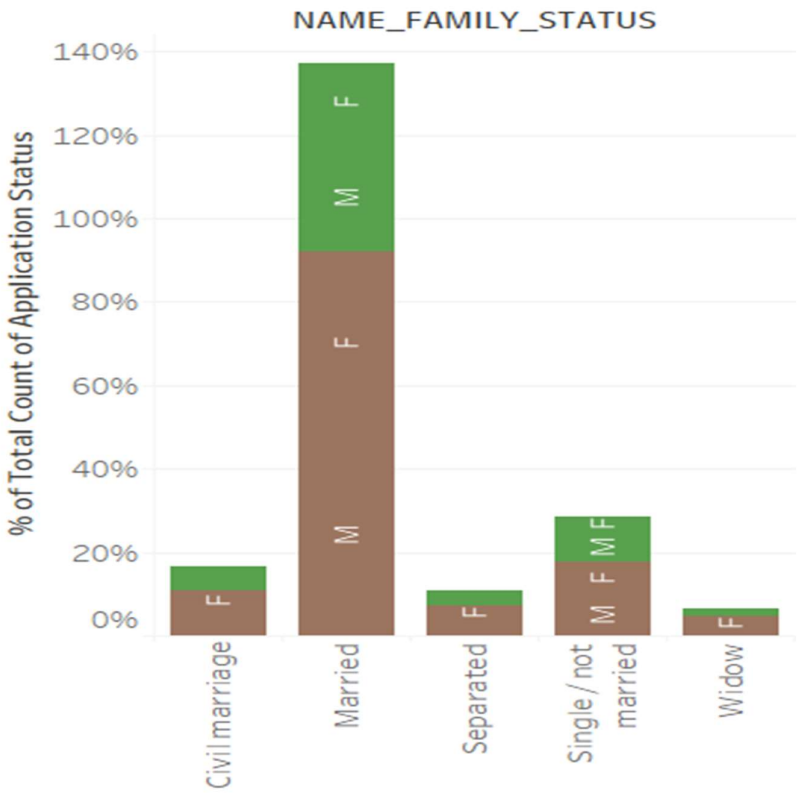
Στο πάνω διάγραμμα, με πράσινο χρώμα είναι τα άτομα που δεν έχουν κάτι στην ιδιοκτησία τους ενώ με το καφέ χρώμα είναι αυτοί που έχουν.

Προκύπτει ότι το μεγαλύτερο ποσοστό αιτήσεων ανήκει σε παντρεμένα άτομα. Εκ των οποίων το 44,76% βρίσκονται σε ιδιοκτησιακό καθεστώς ενώ το 19,41% όχι.



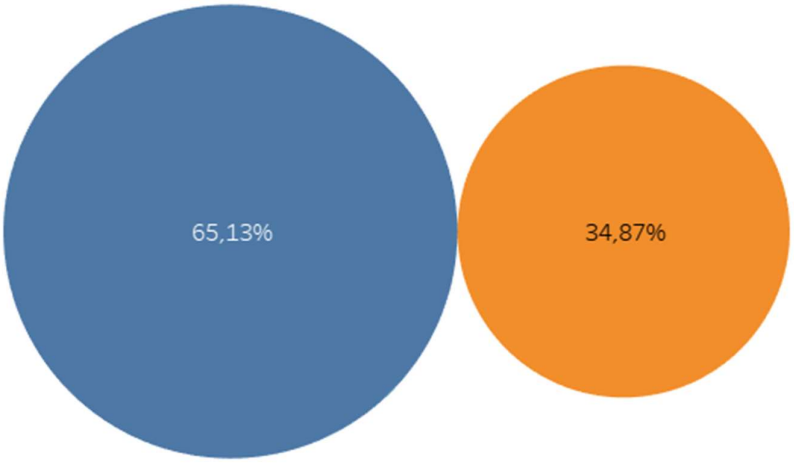
Σχήμα 23 Αριθμός αιτήσεων σε σχέση με την ηλικία

Συμπεραίνουμε ότι τις περισσότερες αιτήσεις έχουν γίνει από τις ηλικίες 28 και 38 ετών αντίστοιχα.



Σχήμα 24 Ποσοστό αριθμού αιτήσεων σε σχέση με την οικογενειακή κατάσταση, την κατοχή ιδιοκτησίας και το φύλο

Απεικονίζουμε με καφέ χρώμα τις αιτήσεις που κατέχουν ιδιοκτησία και με πράσινο χρώμα αυτές που δεν έχουν. Επίσης, έχοντας ως κριτήριο την οικογενειακή κατάσταση αλλά και το φύλο του αιτούντα, παρατηρούμε ότι το μεγαλύτερο ποσοστό των αιτήσεων το έχουν οι παντρεμένοι άντρες σε ποσοστό 47,16% έναντι των γυναικών 44,76% .



Σχήμα 25 Ποσοστό αριθμού αιτήσεων σε σχέση με το φύλο

Αναπαριστούμε με μπλε χρώμα το ποσοστό των αιτήσεων που είναι άνδρες και με πορτοκαλί τις γυναίκες. Το ποσοστό των ανδρών 65,13% είναι σχεδόν διπλάσιο από των γυναικών που είναι 34,87% .

4.3.9 Διερεύνηση για ακραίες τιμές

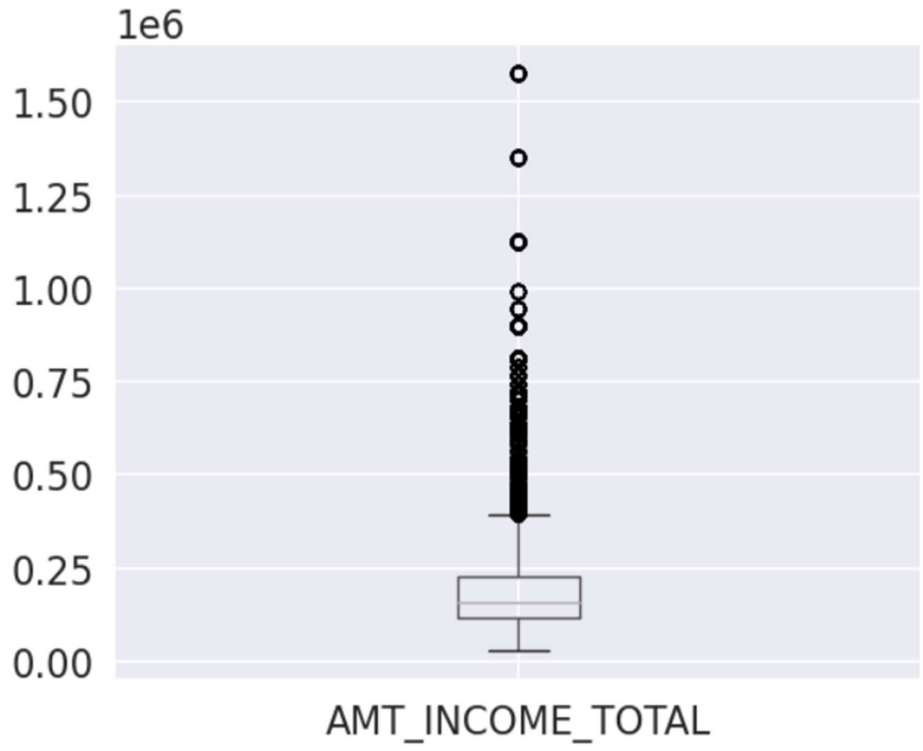
Θα διερευνήσουμε αν στο dataset μας περιέχονται ακραίες τιμές.

Στην αρχή θα ορίσουμε μία διαδικασία, προκειμένου να καθορίσουμε τα τεταρτημόρια και ορίζουμε το min, max των δεδομένων.

```
def dropOL(ftr):
    q75,q25 = np.percentile(final[ftr],[75,25])
    intr_qr = q75-q25
    mx = q75+(1.5*intr_qr)
    mn = q25-(1.5*intr_qr)
    return mx,mn
```

Έπειτα, μέσω του boxplot θα διαπιστώσουμε αν υπάρχουν ακραίες τιμές στην στήλη AMT_INCOME_TOTAL

```
final.boxplot('AMT_INCOME_TOTAL')
```



Σχήμα 26 Boxplot συνολικού εισοδήματος

Από το παραπάνω διάγραμμα διαπιστώνουμε ότι υπάρχει μεγάλη απόσταση μεταξύ του άνω φράχτη και του μέγιστου σημείου που είναι στο 0,4. Έτσι τα δεδομένα μας είναι πιο απλωμένα και υπάρχουν περισσότερες ακραίες τιμές.

Κάνουμε εκκαθάριση των ακραίων τιμών

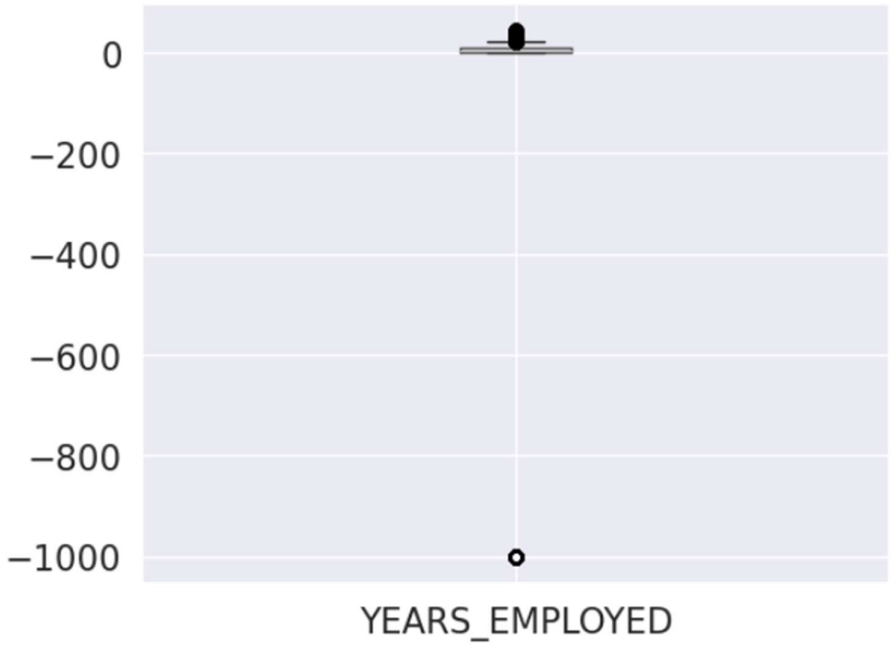
```
mx,mn = dropOL('AMT_INCOME_TOTAL')
```

```
final.drop(final[final.AMT_INCOME_TOTAL > mx].index,inplace=True)
```

Όπου θα αφαιρέσουμε τις τιμές που βρίσκονται πάνω από το mx του AMT_INCOME_TOTAL.

Αντίστοιχα, θα κάνουμε και για το YEARS_EMPLOYED

```
final.boxplot('YEARS_EMPLOYED')
```



Σχήμα 27 Boxplot χρόνια εργασίας

Σε αυτό το διάγραμμα διαπιστώνουμε ότι δεν υπάρχουν ακραίες τιμές παρά μόνο μία εκτός του άνω φράχτη. Το μεγαλύτερο πλήθος των τιμών βρίσκεται εντός των δύο φραχτών, αφού η YEARS_EMPLOYED παίρνει θετικές τιμές.

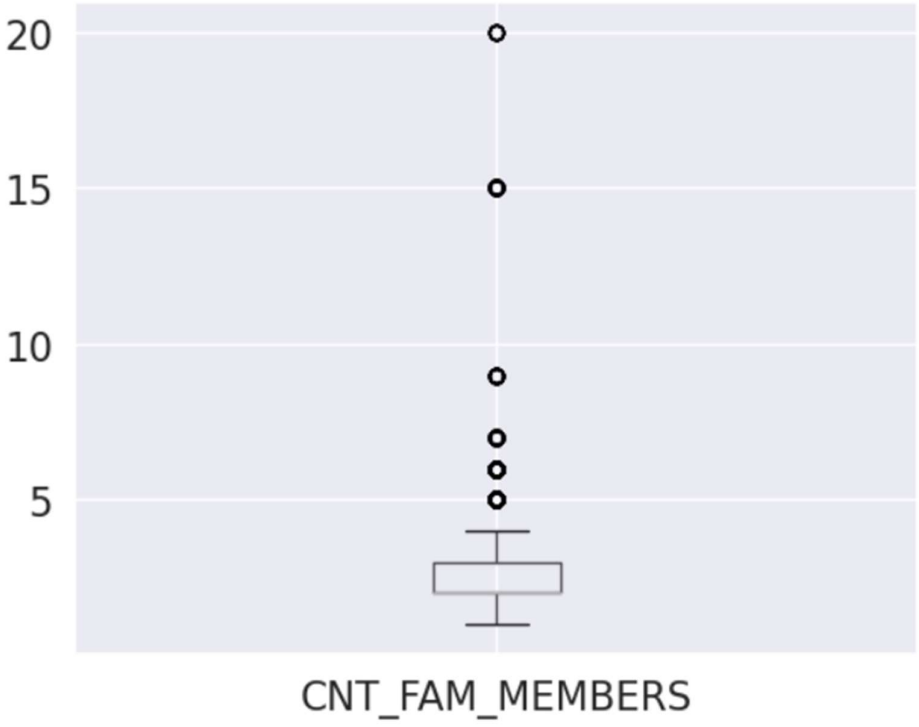
Οπότε θα ακολουθήσουμε τη διαδικασία αφαίρεσης της ακραίας τιμής .

```
mx,mn = dropOL('YEARS_EMPLOYED')
```

```
final.drop(final[final.DAYS_EMPLOYED > mx].index,inplace=True)
```

Ακολούθως, θα δούμε τις ακραίες τιμές μέσω boxplot στο CNT_FAM_MEMBERS

```
final.boxplot('CNT_FAM_MEMBERS')
```



Σχήμα 28 Boxplot αριθμός μελών οικογένειας

Παρατηρούμε ότι, δεν είναι πολλές οι ακραίες τιμές έξω από τον άνω φράχτη.

```
mx,mn = dropOL('CNT_FAM_MEMBERS')
```

Και θα αφαιρέσουμε τις τιμές που είναι πάνω από 6 παιδιά. Ώστε σε περίπτωση έγκρισης της αίτησης για έκδοση πιστωτικής κάρτας, να μπορεί ο αιτούμενος να αποπληρώσει το χρέος του προς την τράπεζα, λόγω χρήσης της πιστωτικής κάρτας.

```
final.drop(final[final.CNT_FAM_MEMBERS > 6].index,inplace=True)
```

Με την εντολή info ζητάμε να μας δοθούν πληροφορίες για το τύπο των δεδομένων που περιέχονται στις στήλες και ποιες είναι αυτές.

```
final.info()
```

και μας εμφανίζονται

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 173539 entries, 16 to 219172
Data columns (total 14 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   CODE_GENDER          173539 non-null  int64  
 1   FLAG_OWN_CAR         173539 non-null  int64  
 2   FLAG_OWN_REALTY      173539 non-null  int64  
 3   AMT_INCOME_TOTAL     173539 non-null  float64
 4   NAME_INCOME_TYPE     173539 non-null  object 
 5   NAME_EDUCATION_TYPE  173539 non-null  object 
 6   NAME_FAMILY_STATUS   173539 non-null  object 
 7   NAME_HOUSING_TYPE    173539 non-null  object 
 8   DAYS_EMPLOYED        173539 non-null  int64  
 9   FLAG_PHONE           173539 non-null  int64  
 10  CNT_FAM_MEMBERS      173539 non-null  float64
 11  Age                  173539 non-null  int64  
 12  MONTHS_BALANCE       173539 non-null  int64  
 13  STATUS               173539 non-null  int64  
dtypes: float64(2), int64(8), object(4)
memory usage: 19.9+ MB
```

Εικόνα 12 Πληροφορίες για τον πίνακα final

Σχεδόν όλα τα δεδομένα είναι μορφή αριθμών ακεραίων (int64) και δεκαδικών (float64), πλην τεσσάρων στηλών (NAME_INCOME_TYPE, NAME_EDUCATION_TYPE, NAME_FAMILY_STATUS και NAME_HOUSING_TYPE) όπου το περιεχόμενο των στηλών είναι γράμματα (object).

Μπορούμε να το διαπιστώσουμε και με την εντολή .head()

```
final.head()
```

	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE	NAME_FAMILY_STATUS	NAME_HOUSING_TYPE
16	0	0	0	112500.0	Working	Secondary	Married	House
17	0	0	0	112500.0	Working	Secondary	Married	House
18	0	0	0	112500.0	Working	Secondary	Married	House
19	0	0	0	112500.0	Working	Secondary	Married	House
20	0	0	0	112500.0	Working	Secondary	Married	House

Εικόνα 13 Δειγματοληπτικές πέντε εγγραφές του πίνακα final (μέσω της εντολής .head())

Πλέον των αριθμών, υπάρχουν και γράμματα (objects) .

Οπότε, για να μπορέσουμε να τα επεξεργαστούμε όσο το δυνατόν περισσότερο, θα πρέπει να τα μετατρέψουμε σε αριθμούς.

4.3.10 Μετατροπή δεδομένων σε αριθμητικές τιμές

Για τη μετατροπή στηλών από μη αριθμητικά δεδομένα σε αριθμητικά, μία δεδομένη συνάρτηση που χρησιμοποιείται είναι η LabelEncoder της βιβλιοθήκης scikitlearn.

Για τις στήλες αυτές, δημιουργείται μία μεταβλητή στην οποία θα χρησιμοποιούμε την συνάρτηση LabelEncoded.

Μετά με μία επαναλαμβανόμενη διαδικασία στις συγκεκριμένες στήλες θα εφαρμόζεται για τη μεταβλητή που δημιουργήσαμε το fit_transform ώστε να γίνεται η μετατροπή τους σε αριθμητικά δεδομένα.

```
features = ['AMT_INCOME_TOTAL','NAME_EDUCATION_TYPE', 'NAME_INCOME_TYPE','NAME_FAMILY_STATUS','NAME_HOUSING_TYPE']

le = preprocessing.LabelEncoder()

for col in features:
    final[col] = le.fit_transform(final[col])
```

και καλώντας ξανά την εντολή .info() εμφανίζεται

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 173539 entries, 16 to 219172
Data columns (total 14 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   CODE_GENDER          173539 non-null  int64
 1   FLAG_OWN_CAR         173539 non-null  int64
 2   FLAG_OWN_REALTY      173539 non-null  int64
 3   AMT_INCOME_TOTAL     173539 non-null  int64
 4   NAME_INCOME_TYPE     173539 non-null  int64
 5   NAME_EDUCATION_TYPE  173539 non-null  int64
 6   NAME_FAMILY_STATUS   173539 non-null  int64
 7   NAME_HOUSING_TYPE    173539 non-null  int64
 8   DAYS_EMPLOYED        173539 non-null  int64
 9   FLAG_PHONE           173539 non-null  int64
 10  CNT_FAM_MEMBERS      173539 non-null  float64
 11  Age                  173539 non-null  int64
 12  MONTHS_BALANCE       173539 non-null  int64
 13  STATUS               173539 non-null  int64
dtypes: float64(1), int64(13)
memory usage: 19.9 MB
```

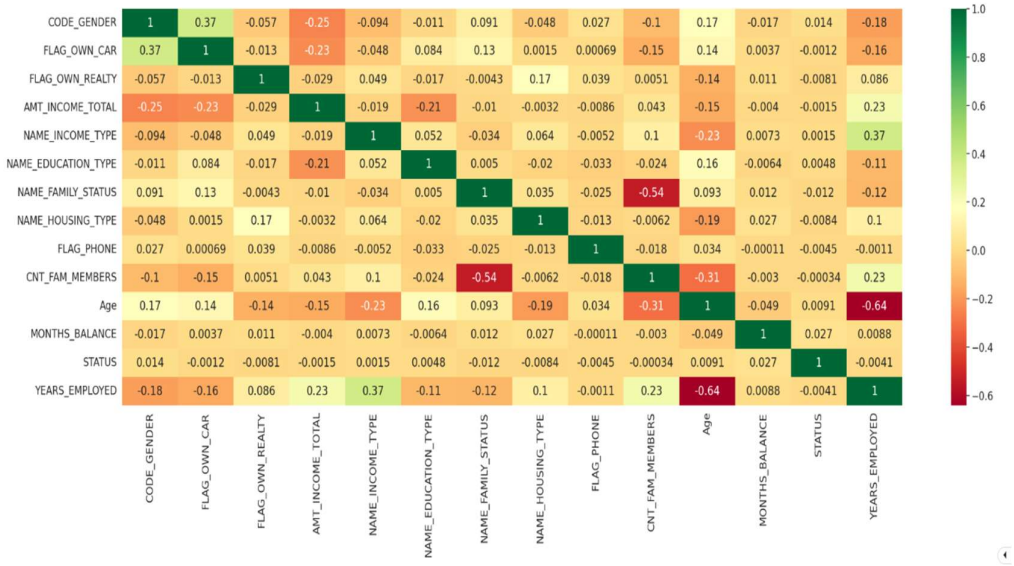
Εικόνα 14 Πληροφορίες του πίνακα final μετά την μετατροπή σε αριθμητικές τιμές

όπου διαπιστώνουμε ότι όλα τα δεδομένα μας πλέον είναι αριθμητικά.

4.3.11 Μελέτη συσχέτισης δεδομένων

Χρησιμοποιώντας από τη βιβλιοθήκη Seaborn, το διάγραμμα heatmap ή διαφορετικά πίνακας συσχετίσεων, θα μπορέσουμε να παρατηρήσουμε ποιες στήλες έχουν μεταξύ τους τη μεγαλύτερη συσχέτιση. Όσο πιο κοντά στη τιμή 1 μία στήλη με μία άλλη τόσο μεγαλύτερη συσχέτιση έχουν μεταξύ τους.

```
plt.figure(figsize=(32,10))
sns.heatmap(final.corr(),annot=True,cmap='RdYlGn')
```



Εικόνα 15 Πίνακας συσχετίσεων (heatmap) 14 μεταβλητών

Από το παραπάνω διάγραμμα διαπιστώνουμε ότι, 4 στήλες μεταξύ τους έχουν υψηλό βαθμό συσχέτισης. Η FLAG_OWN_CAR με το CODE_GENDER έχει βαθμό συσχέτισης 0,37. Με τον ίδιο βαθμό συσχετίζονται και οι στήλες YEARS_EMPLOYED και NAME_INCOME_TYPE.

Τα παραπάνω σημαίνουν, ότι ο παράγοντας που συμβάλει περισσότερο στην αν κάποιος έχει αμάξι είναι το φύλο του αιτούμενου. Ενώ, ο τύπος του εισοδήματος εξαρτάται από τα χρόνια απασχόλησης.

4.4. Δημιουργία μοντέλων – Machine Learning

4.4.1 Δημιουργία ισορροπίας στα δεδομένα - SMOTE

Από τα παραπάνω διαγράμματα διαπιστώνουμε ότι υπάρχει ανισορροπία στα δεδομένα μας. Η αντιμετώπιση της ανισορροπίας επιτυγχάνεται μέσω της βιβλιοθήκης SMOTE και συγκεκριμένα με την υπερβολική δειγματοληψία (oversampling).

Ειδικότερα, η βιβλιοθήκη SMOTE, Λειτουργεί δημιουργώντας συνθετικά δείγματα από την δευτερεύουσα τάξη (χωρίς κάρτα) αντί να δημιουργεί αντίγραφα.

Επιλέγει τυχαία έναν από τους k-πλησιέστερους γείτονες και τον χρησιμοποιεί για να δημιουργήσει παρόμοιες, αλλά τυχαία τροποποιημένες, νέες παρατηρήσεις. (Polanitzer, 2022)

Έτσι, αρχικοποιούμε τις τιμές των μεταβλητών X και Y ως ακολούθως

```
Y = final['STATUS']
X = final.drop(['STATUS'], axis=1)
```

Έπειτα θα χρησιμοποιήσουμε τη βιβλιοθήκη SMOTE

```
from imblearn.over_sampling import SMOTE
Y = Y.astype('int')
X_balance,Y_balance = SMOTE().fit_resample(X,Y)
X_balance = pd.DataFrame(X_balance, columns = X.columns)
```

4.4.2 Διαχωρισμός δεδομένων (split)

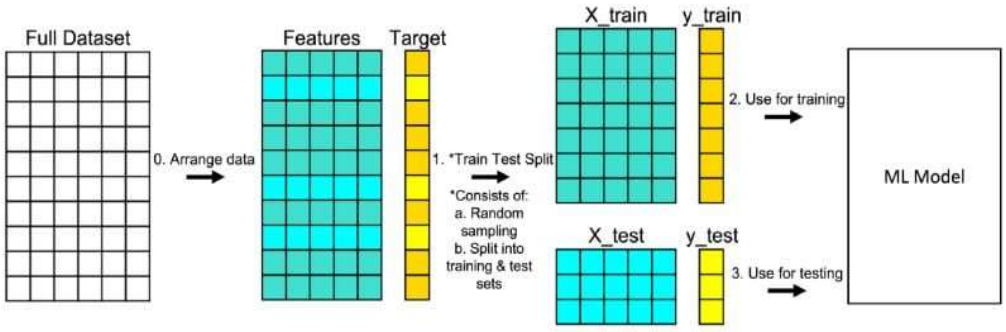
4.4.2.1 Τί είναι το train, test, split των δεδομένων

Το Train test split είναι μια διαδικασία επικύρωσης μοντέλου που μας επιτρέπει να προσομοιώσουμε την απόδοση του μοντέλου μας με νέα δεδομένα. (Galarnyk, 2022)

4.4.2.2 Διαδικασία του train, test, split

Το train, test, split είναι μια διαδικασία επικύρωσης μοντέλου που μας επιτρέπει να προσομοιώσουμε την απόδοση ενός μοντέλου σε νέα/αόρατα δεδομένα. (Galarnyk, 2022)

Ας δούμε στο παρακάτω σχήμα, τη διαδικασία που ακολουθείται:



Σχήμα 29 Διαδικασία split σε train και test

Τα βήματα που ακολουθούνται όπως αυτά φαίνονται στο σχήμα, είναι:

1. Τακτοποίηση των δεδομένων μας
2. Διαχωρισμός των δεδομένων μας
3. Εκπαίδευση του μοντέλου μας
4. Δοκιμή του μοντέλου μας

Μετά την τακτοποίηση των δεδομένων που έχουμε προβεί προγενέστερα, πηγαίνουμε στο επόμενο βήμα που είναι ο διαχωρισμός των δεδομένων μας σε train και test.

```
X_train, X_test, y_train, y_test = train_test_split(X_balance,Y_balance,
                                                    test_size=0.3,
                                                       random_state = 10086)
```

Με το test_size = 0.3 διαχωρίζουμε τα δεδομένα μας σε 70% σε train set και σε 30% σε test set.

Το random_state, μια παράμετρος ψευδοτυχαίων αριθμών που μας επιτρέπει να αναπαράγουμε τον ίδιο διαχωρισμό train test, κάθε φορά που εκτελείτε ο κώδικας. (Galarnyk, 2022)

Υπάρχουν περιπτώσεις όπου το random_state, υποδηλώνει το όνομα και χρησιμοποιείται για την προετοιμασία της εσωτερικής γεννήτριας τυχαίων αριθμών, η οποία θα αποφασίσει το διαχωρισμό των δεδομένων σε δείκτες train και test στην περίπτωσή μας.

Εάν το random_state είναι ένας ακέραιος, τότε χρησιμοποιείται για την εμφάνιση ενός νέου αντικειμένου RandomState. (O., 2022)

4.4.3 Decision Trees (Δέντρα Αποφάσεων)

Στο παρακάτω μοντέλο έχουν τοποθετηθεί ορίσματα στην συνάρτηση Grid Search αλλά φυσικά με υπερπαραμέτρους διαφορετικές για κάθε αλγόριθμο. Παρακάτω αναφέρονται τα ορίσματα αυτά πιο αναλυτικά:

- **params_grid:** είναι η τιμή του dictionary που έχει οριστεί στο κώδικα με το σύνολο υπερπαραμέτρων για το κάθε μοντέλο.

- **n_jobs**=2: ο αριθμός διεργασιών που εκτελούνται παράλληλα για αυτή την εργασία. Η τιμή -1 δηλώνει ότι χρησιμοποιούνται όλοι οι διαθέσιμοι επεξεργαστές.

- **verbose**=1: δείχνει λεπτομερώς την εκτύπωση ενώ προσαρμόζονται τα δεδομένα στο GridSearchCV

- **scoring**= “ roc_auc “ : μέτρηση αξιολόγησης (developers c.-l. , 2023)

```
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
decision_param_grid={"max_depth":[2,40,2],
"min_samples_split":[2,20,2]}
decision = DecisionTreeClassifier(random_state=123)
rand_search = GridSearchCV(estimator=decision, param_grid=decision_param_grid,
n_jobs=2, verbose=1, scoring='roc_auc')
rand_search.fit(X_train,y_train)
preds=rand_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", rand_search.best_params_)
print("Best score found:", rand_search.best_score_)
```

```
Fitting 5 folds for each of 9 candidates, totalling 45 fits
Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)
              precision    recall  f1-score   support

           0       0.98      0.99      0.98     59677
           1       0.99      0.98      0.98     59728

    accuracy                           0.98    119405
   macro avg       0.98      0.98      0.98    119405
weighted avg       0.98      0.98      0.98    119405

Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)
[[58830   847]
 [ 1405 58323]]
Best parameters found: {'max_depth': 40, 'min_samples_split': 20}
Best score found: 0.9907522855315394
```

Εικόνα 16 Έκθεση ταξινόμησης (Classification report) – Decision Tree Classifier

4.4.3.1 Ορισμός παραμέτρων (Decision Tree Classifier)

Στη συγκεκριμένη περίπτωση (Decision Tree Classifier), έχουμε ορίσει τις παραμέτρους :

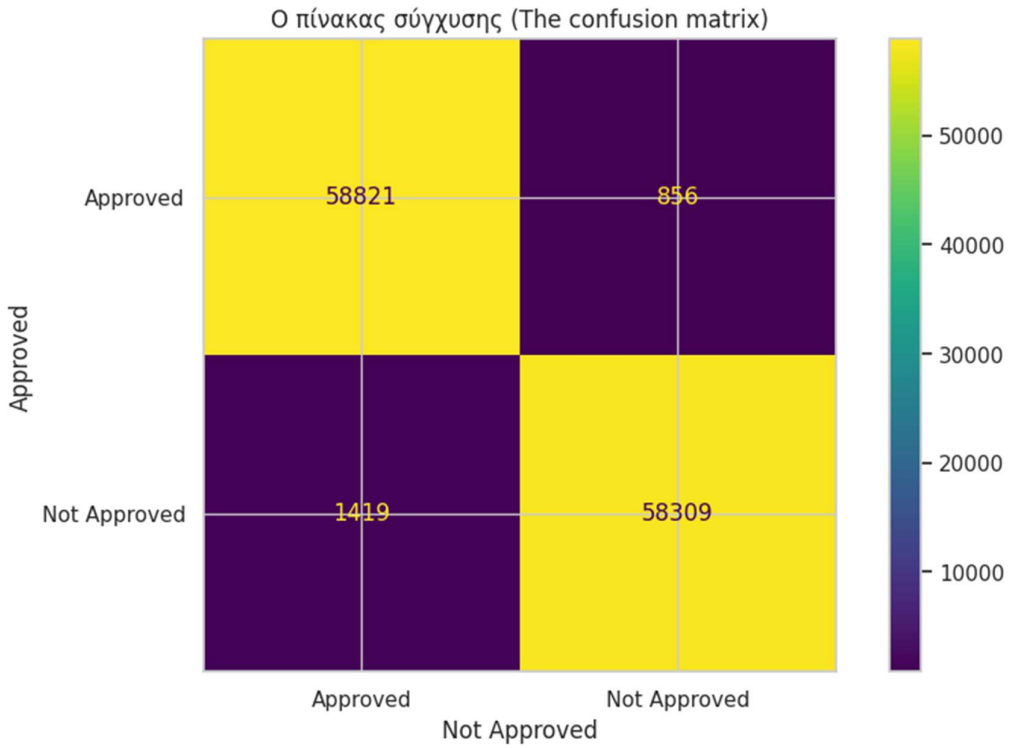
- max_depth=40,
- min_samples_split=20,
- random_state=1024

προκειμένου να εκπαιδεύσουμε το μοντέλο μας, επιλέξαμε τις παραπάνω τιμές. Ο υπολογισμός του αλγορίθμου των Δέντρων Αποφάσεων, θα γίνει μέσω της πλατφόρμας της Kaggle, λόγω ότι διαθέτει μεγάλη ισχύ επεξεργαστών, τόσο του κλασικού CPU όσο και του GPU (κάρτα γραφικών).

```python
from sklearn.metrics import accuracy_score, confusion_matrix
model1 = DecisionTreeClassifier(max_depth=40,
                                min_samples_split=20,
                                random_state=1024)
model1.fit(X_train, y_train)
y_predict = model1.predict(X_test)

print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()
```

```
Accuracy Score is 0.98095
```



Σχήμα 30 Πίνακας σύγχυσης (Confusion matrix) – Decision Trees Classifier

Το σύνολο δεδομένων που χρησιμοποιείται για την εκπαίδευση (train dataset) έχει 119.405 τιμές. Οι 856 από αυτές ταξινομούνται λανθασμένα ως θετικές ενώ είναι αρνητικές (FP) και οι 1.419 από αυτές ταξινομούνται ως αρνητικές ενώ είναι θετικές (FN).

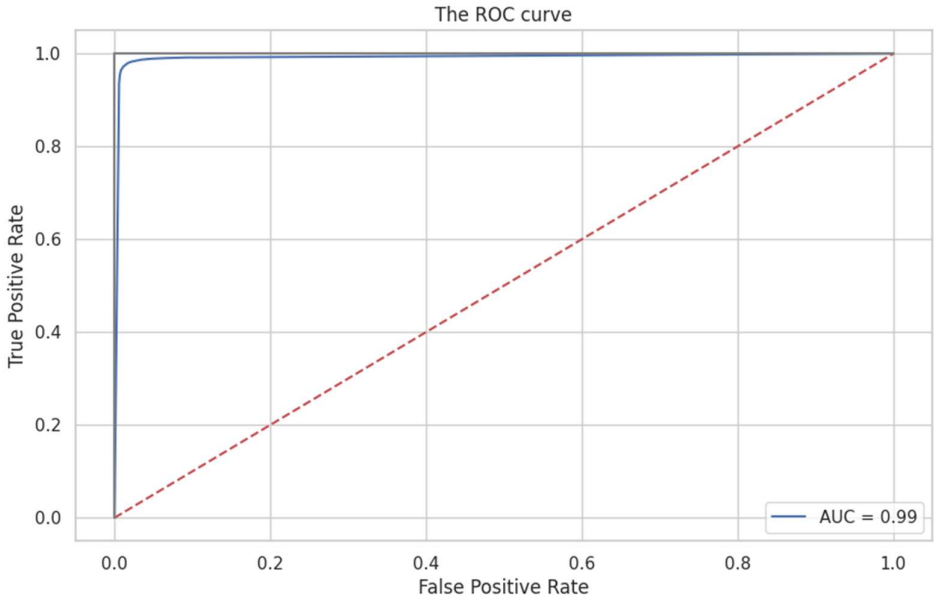
4.4.3.2 The ROC Curve (Decision Trees)

Θα ακολουθήσουμε στη συνέχεια κάποια βήματα – εντολές στην Python προκειμένου να σχεδιαστεί η καμπύλη ROC και εμφάνιση της τιμής AUC.

```
from sklearn import metrics
preds2 = model1.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds2)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)

plt.figure(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()
```

```
Roc_Auc: 0.99
```



Σχήμα 31 Η καμπύλη ROC (Receiver Operating Characteristic) – Decision Trees Classifier

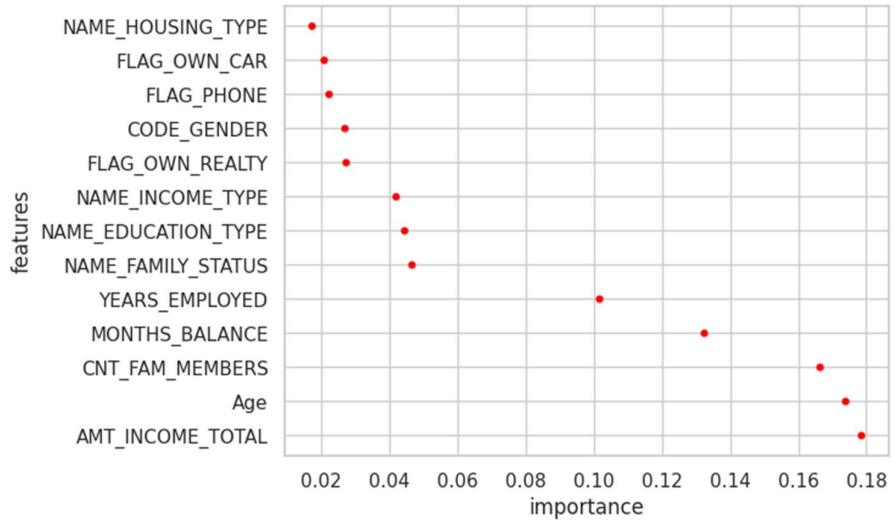
Από το διάγραμμα διαπιστώνουμε ότι η τιμή της ROC είναι η ιδανική αφού κινείται στην μέγιστη τιμή 1.0, για τον άξονα των ορθώς θετικών (true positive) και σταθεροποιείται στην τιμή 1. Ενώ, η τιμή AUC είναι 0.99 .

4.4.3.3 Διάγραμμα σπουδαιότητας (Decision Trees)

Παρακάτω θα εμφανίσουμε το διάγραμμα σπουδαιότητας, χρησιμοποιώντας τη διαδικασία :

```
def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model1, X_train,20)
```



Σχήμα 32 Διάγραμμα σπουδαιότητας – Decision Trees Classifier

Το διάγραμμα σπουδαιότητας μας δείχνει ότι το πιο σημαντικό γνώρισμα είναι *το συνολικό εισόδημά του,* με τιμή κοντά στο 0,18 .

Αμέσως, επόμενα σημαντικά είναι **η ηλικία του αιτούντα** για τη χορήγηση της πιστωτικής κάρτας και **ο αριθμός των μελών της οικογένειας.**

4.4.4 Random Forest (Τυχαίο δάσος)

Στο παρακάτω μοντέλο έχουν τοποθετηθεί τα ίδια ορίσματα στην συνάρτηση Grid Search αλλά φυσικά με υπερπαραμέτρους διαφορετικές για κάθε αλγόριθμο. Παρακάτω αναφέρονται τα ορίσματα αυτά πιο αναλυτικά:

• **params_grid:** είναι η τιμή του dictionary που έχει οριστεί στο κώδικα με το σύνολο υπερπαραμέτρων για το κάθε μοντέλο.

• **n_jobs**=2: ο αριθμός διεργασιών που εκτελούνται παράλληλα για αυτή την εργασία. Η τιμή -1 δηλώνει ότι χρησιμοποιούνται όλοι οι διαθέσιμοι επεξεργαστές.

• **verbose**=1: δείχνει λεπτομερώς την εκτύπωση ενώ προσαρμόζονται τα δεδομένα στο GridSearchCV

• **scoring**= “ roc_auc “ : μέτρηση αξιολόγησης (developers c.-l. , 2023)

```
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
decision_param_grid={"max_depth":[2,40,2],
"min_samples_split":[2,20,2]}
decision = DecisionTreeClassifier(random_state=123)
rand_search = GridSearchCV(estimator=decision, param_grid=decision_param_grid,
n_jobs=2, verbose=1, scoring='roc_auc')
rand_search.fit(X_train,y_train)
preds=rand_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", rand_search.best_params_)
print("Best score found:", rand_search.best_score_)
```

```
Fitting 5 folds for each of 27 candidates, totalling 135 fits
Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)
              precision    recall  f1-score   support

           0       0.98      0.99      0.99     59677
           1       0.99      0.98      0.99     59728

    accuracy                           0.99    119405
   macro avg       0.99      0.99      0.99    119405
weighted avg       0.99      0.99      0.99    119405

Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)
[[58994   683]
 [ 1059 58669]]
Best parameters found: {'max_depth': 20, 'min_samples_leaf': 2, 'n_estimators': 300}
Best score found: 0.9967707679726627
```

Εικόνα 17 Έκθεση ταξινόμησης (Classification report)- Random Forest Classifier

4.4.4.1 Υπερπαράμετροι ενός Τυχαίου Δάσους

Παρακάτω είναι η λίστα με τις πιο σημαντικές παραμέτρους και παρακάτω μια πιο εκλεπτυσμένη ενότητα σχετικά με τον τρόπο βελτίωσης της ισχύος πρόβλεψης και την ευκολότερη φάση εκπαίδευσης του μοντέλου σας. (Arya, 2022)

i. **max_depth:** Το μέγιστο βάθος του δέντρου - σημαίνει τη μεγαλύτερη διαδρομή μεταξύ του κόμβου ρίζας και του κόμβου του φύλλου.
ii. **min_samples_leaf:** Αυτός είναι ο ελάχιστος αριθμός δειγμάτων που απαιτείται να βρίσκονται σε έναν κόμβο φύλλου όπου η προεπιλογή = 1
iii. **n_estimators:** Αυτός είναι ο αριθμός των δέντρων στο δάσος.
iv. **κριτήριο:** Η συνάρτηση για τη μέτρηση της ποιότητας ενός διαχωρισμού

4.4.4.2 Συντονισμός των υπερπαραμέτρων για τη βελτίωση των προβλέψεων

- **n_estimators :** Ο αριθμός των δέντρων στο μοντέλο. Η τιμή που πήρε είναι {300}
- **max_depth :** Το μέγιστο βάθος του δέντρου. Η τιμή που πήρε είναι {20}
- **min_samples_leaf :** ο ελάχιστος αριθμός δειγμάτων που απαιτείται να βρίσκονται σε έναν κόμβο φύλλου. Η τιμή που πήρε είναι {2}.

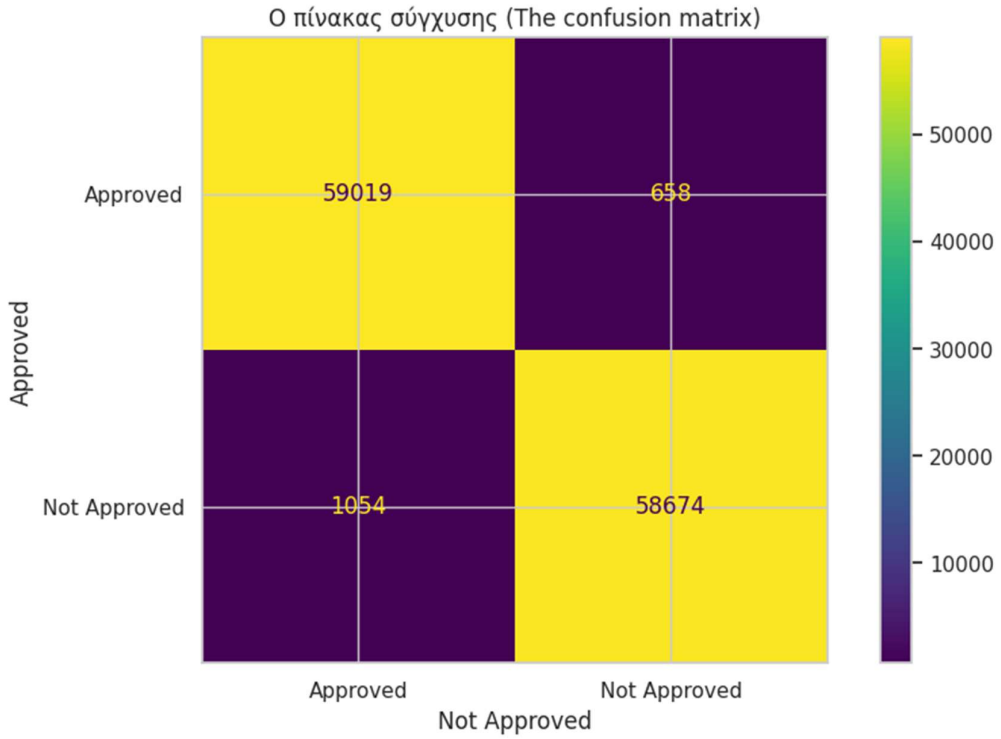
Ο υπολογισμός του αλγορίθμου των Τυχαίου Δάσους (Random Forest), θα γίνει μέσω της πλατφόρμας της Kaggle, λόγω ότι διαθέτει μεγάλη ισχύ επεξεργαστών.

```
from sklearn.ensemble import RandomForestClassifier
model2 = RandomForestClassifier(n_estimators=300,
                                max_depth=20,
                                min_samples_leaf=2
                                )
model2.fit(X_train, y_train)
y_predict = model2.predict(X_test)

print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
confusion_matrix(y_test,y_predict)

cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, y_predict))
```

```
Accuracy Score is 0.98566
```



Σχήμα 33 Πίνακας σύγχυσης Τυχαίου Δάσους (Confusion matrix Random Forest)

Το σύνολο δεδομένων που χρησιμοποιείται για την εκπαίδευση (train dataset) έχει 119.405 τιμές. Οι 658 από αυτές ταξινομούνται λανθασμένα ως θετικές ενώ είναι αρνητικές (FP) και 1.054 από αυτές που ταξινομούνται ως αρνητικές ενώ είναι θετικές (FN).

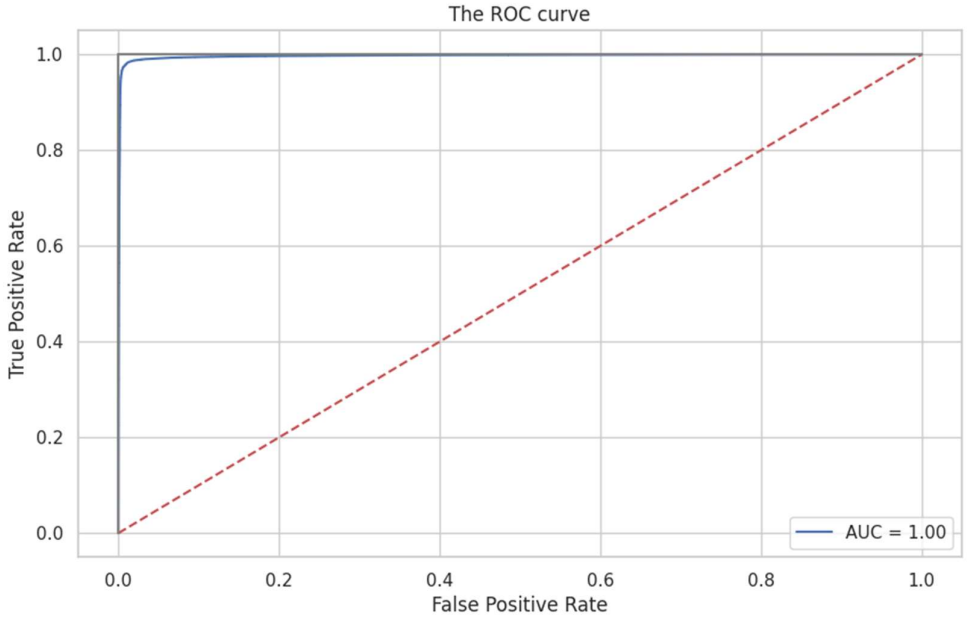
4.4.4.3 The ROC Curve (Random Forest)

Όπως προηγουμένως, θα ακολουθήσουμε κάποια βήματα – εντολές στην Python προκειμένου να σχεδιαστεί η καμπύλη ROC.

```
from sklearn import metrics
preds3 = model2.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds3)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)

plt.subplots(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()
```

```
Roc Auc: 1.00
```



Σχήμα 34 Η καμπύλη ROC (Receiver Operating Characteristic – Random Forest)

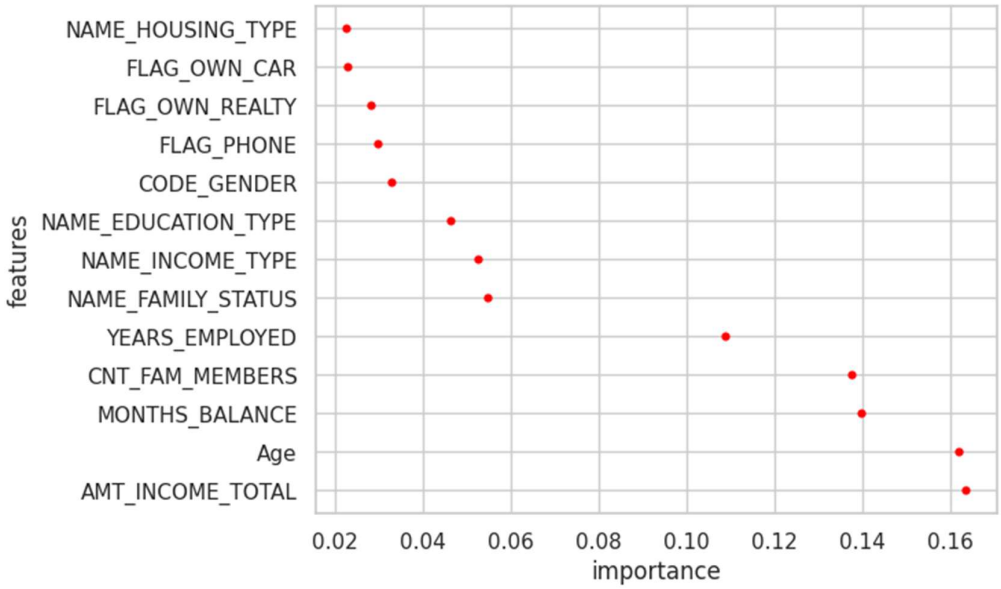
Από το διάγραμμα διαπιστώνουμε ότι η τιμή της ROC είναι η ιδανική αφού κινείται στην μέγιστη τιμή 1.0, για τον άξονα των ορθώς θετικών (true positive) και σταθεροποιείται στην τιμή 1. Ενώ, η τιμή AUC είναι 1.00 .

4.4.4.4 Διάγραμμα σπουδαιότητας (Random Forest)

Παρακάτω θα εμφανίσουμε το διάγραμμα σπουδαιότητας, χρησιμοποιώντας τη διαδικασία :

```
def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model2, X_train,20)
```



Σχήμα 35 Διάγραμμα σπουδαιότητας – Random Forest Classifier

Το διάγραμμα σπουδαιότητας μας δείχνει ότι πιο σημαντικό γνώρισμα είναι *το συνολικό εισόδημα του αιτούντα,* με τιμή πάνω από το 0,16 .
Αμέσως, επόμενα σημαντικά είναι **η ηλικία** και **ο μήνας υπολοίπου σε σχέση με την ημερομηνία αίτησης (-1 δηλώνει την πιο πρόσφατη ημερομηνία υπολοίπου),** για τη χορήγηση της πιστωτικής κάρτας .

4.4.5 XGBoost

Οι υπερπαράμετροι είναι ορισμένες τιμές, που καθορίζουν τη διαδικασία εκμάθησης ενός αλγορίθμου.
Μια μεθοδολογία, που θα ακολουθήσουμε όπως και προηγουμένω για την επιλογή των σωστών υπερπαραμέτρων είναι η Αναζήτηση Πλέγματος (Grid Search). Στο παρακάτω μοντέλο έχουν τοποθετηθεί τα ίδια ορίσματα στην συνάρτηση Grid Search αλλά φυσικά με υπερπαραμέτρους διαφορετικές για κάθε αλγόριθμο. Παρακάτω αναφέρονται τα ορίσματα αυτά πιο αναλυτικά:

- **params_grid:** είναι η τιμή του dictionary που έχει οριστεί στο κώδικα με το σύνολο υπερπαραμέτρων για το κάθε μοντέλο.

- **n_jobs**=2: ο αριθμός διεργασιών που εκτελούνται παράλληλα για αυτή την εργασία. Η τιμή -1 δηλώνει ότι χρησιμοποιούνται όλοι οι διαθέσιμοι επεξεργαστές.

- **verbose**=3: δείχνει λεπτομερώς την εκτύπωση ενώ προσαρμόζονται τα δεδομένα στο GridSearchCV

• **scoring**= “ roc_auc “ : μέτρηση αξιολόγησης (developers c.-l. , 2023)

```
import xgboost as xgb
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
xgb_param_grid={"n_estimators":[200,400, 1000],"max_depth":[5,10],"learning_rate":[0.05,0.1],"min_child_weight":[6,8]}
xgb = XGBClassifier(eval_metric="auc", random_state=123)
xgb_search = GridSearchCV(estimator=xgb, param_grid=xgb_param_grid,n_jobs=2, refit=True, verbose=3, scoring='roc_auc')
xgb_search.fit(X_train,y_train)
preds=xgb_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", xgb_search.best_params_)
print("Best score found:", xgb_search.best_score_)
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)
              precision    recall  f1-score   support

           0       0.99      0.99      0.99     59677
           1       0.99      0.98      0.99     59728

    accuracy                           0.99    119405
   macro avg       0.99      0.99      0.99    119405
weighted avg       0.99      0.99      0.99    119405

Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)
[[59252   425]
 [  897 58831]]
Best parameters found: {'learning_rate': 0.1, 'max_depth': 10, 'min_child_weight': 6, 'n_estimators': 1000}
Best score found: 0.9978476941445779
```

Εικόνα 18 Έκθεση ταξινόμησης (Classification report XGB

- Ειδικότερα, παρακάτω αναφέρονται οι παράμετροι που επιλέξαμε να χρησιμοποιήσουμε στο μοντέλο μας:

1. **max_depth**

✓ Προεπιλογή = 6

✓ Είναι το μέγιστο βάθος ενός δέντρου.

✓ Χρησιμοποιείται για τον έλεγχο της υπερβολικής προσαρμογής .

2. **min_child_weight**

✓ Προεπιλογή = 0

✓ Καθορίζει το ελάχιστο άθροισμα βαρών όλων των παρατηρήσεων που απαιτούνται σε ένα παιδί.

✓ Χρησιμοποιείται για τον έλεγχο της υπερβολικής προσαρμογής.

✓ Όσο μεγαλύτερο είναι το min_child_weight, τόσο πιο συντηρητικός θα είναι ο αλγόριθμος.

✓ Κυμαίνεται από 0 έως άπειρο.

3. **n_estimators**
   ✓ Ο αριθμός των δέντρων στο δάσος.

4. **learning_rate**
   ✓ Ο ρυθμός μάθησης συρρικνώνει τη συμβολή κάθε δέντρου. (developers s.-l. , 2023)

4.4.5.1 Ορισμός παραμέτρων (XGBooster Classifier)

Στη συγκεκριμένη περίπτωση (XGBooster Classifier), έχουμε ορίσει τις παραμέτρους :

- max_depth=10,
- n_estimators=1000
- min_child_weight=8
- learning_rate=0.1

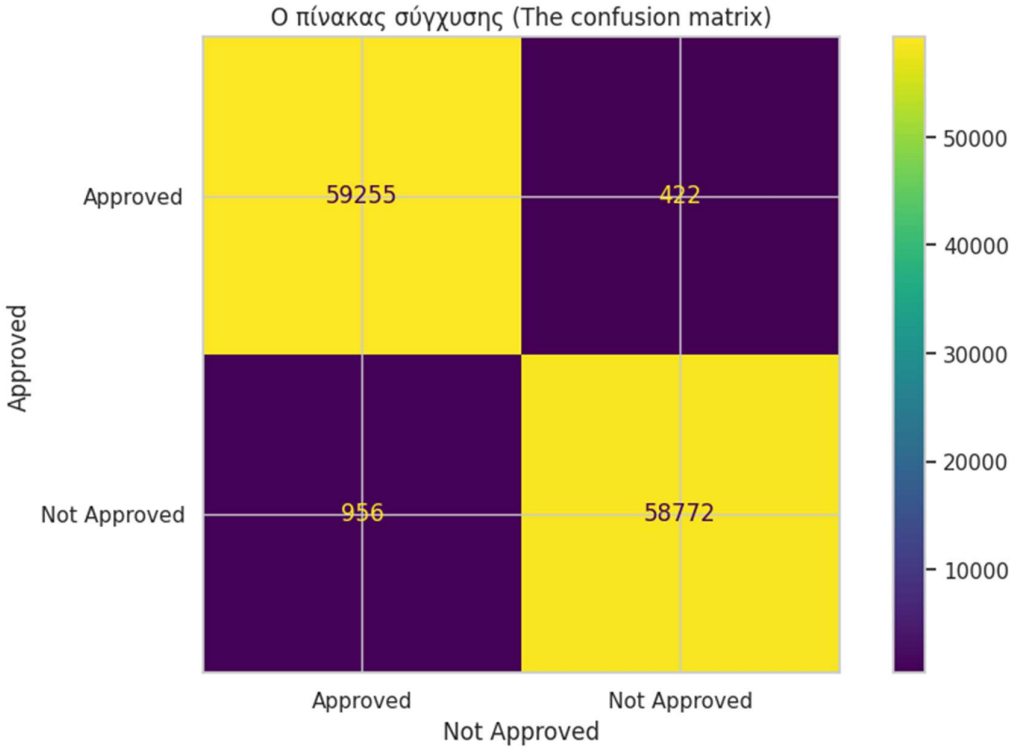
προκειμένου να εκπαιδεύσουμε το μοντέλο μας, επιλέξαμε για δοκιμή τις παραπάνω τιμές.

Ο υπολογισμός του αλγορίθμου XGBoost, θα γίνει μέσω της πλατφόρμας της Kaggle, λόγω ότι διαθέτει μεγάλη ισχύ επεξεργαστών.

```
from xgboost import XGBClassifier
model3 = XGBClassifier(max_depth=10,
                       n_estimators=1000,
                       min_child_weight=8,
                       learning_rate =0.1,
                       )

model3.fit(X_train, y_train)
y_predict = model3.predict(X_test)
print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, y_predict))
```

```
Accuracy Score is 0.98846
```



Σχήμα 36 Πίνακας σύγχυσης XGBoost (Confusion matrix XGBoost)

Το σύνολο δεδομένων που χρησιμοποιείται για την εκπαίδευση (train dataset) έχει 119.405 τιμές. Οι 422 από αυτές ταξινομούνται λανθασμένα ως θετικές ενώ είναι αρνητικές (FP) και οι 956 από αυτές ταξινομούνται ως αρνητικές ενώ είναι θετικές (FN).

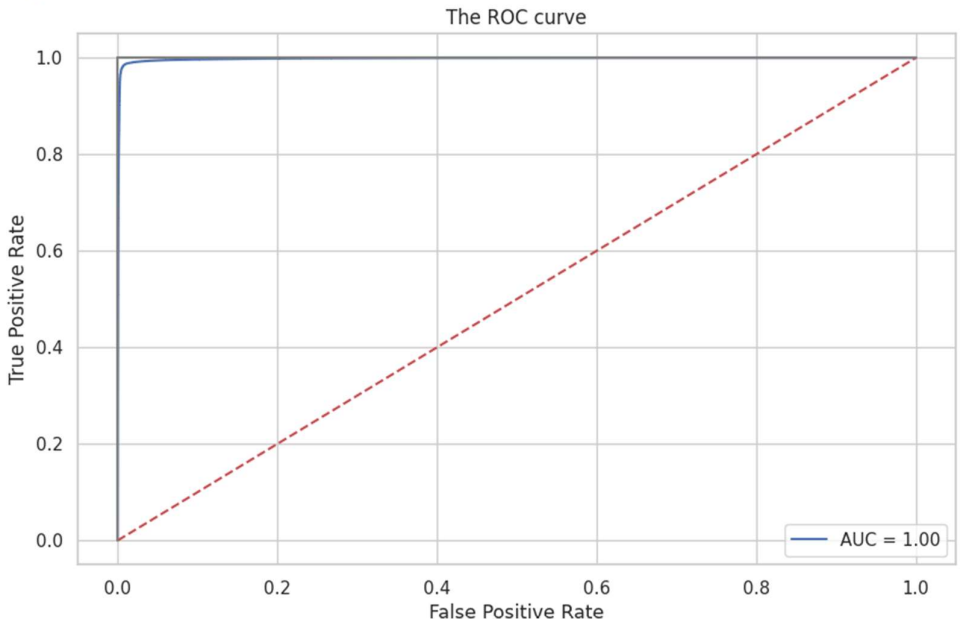
4.4.5.2 The ROC Curve (XGBoost)

Όπως προηγουμένως, θα ακολουθήσουμε κάποια βήματα – εντολές στην Python προκειμένου να σχεδιαστεί η καμπύλη ROC.

```
from sklearn import metrics
preds4 = model3.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds4)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)

plt.subplots(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()
```

```
Roc_Auc: 1.00
```



Σχήμα 37 Η καμπύλη ROC (Receiver Operating Characteristic - XGBoost)

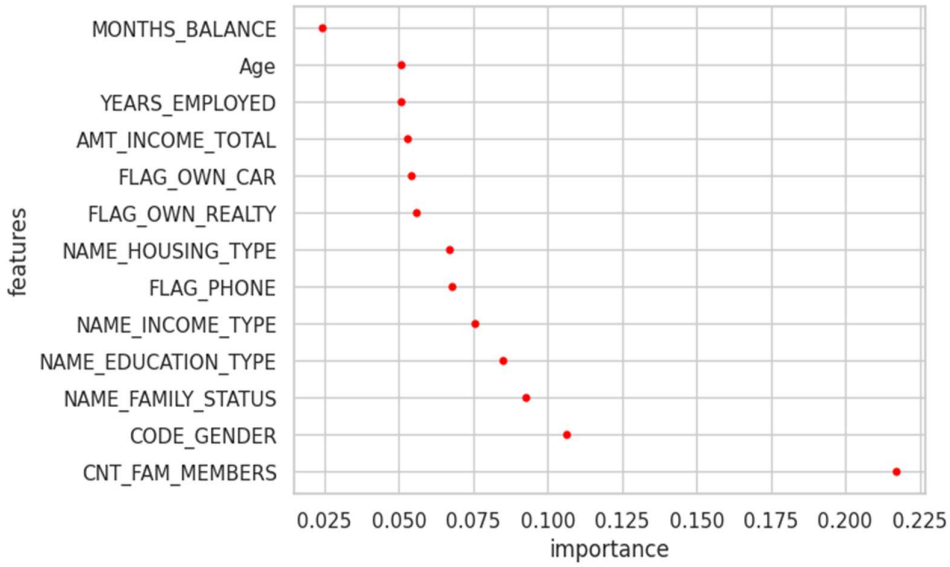
Από το διάγραμμα διαπιστώνουμε ότι η τιμή της ROC είναι η ιδανική αφού κινείται στην μέγιστη τιμή 1.0, για τον άξονα των ορθώς θετικών (true positive) και σταθεροποιείται στην τιμή 1. Ενώ, η τιμή AUC είναι 1.00 .

4.4.5.3 Διάγραμμα σπουδαιότητας (XGBoost)

Παρακάτω θα εμφανίσουμε το διάγραμμα σπουδαιότητας, χρησιμοποιώντας τη διαδικασία :

```
def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model3, X_train,20)
```



Σχήμα 38 Διάγραμμα σπουδαιότητας - XGBoost

Το διάγραμμα σπουδαιότητας μας δείχνει ότι πιο σημαντικό γνώρισμα είναι *ο αριθμός των μελών της οικογένειας του αιτούντα,* με τιμή κοντά στο 0,225 περίπου .

Αμέσως, επόμενα σημαντικά είναι **το φύλο** και **η οικογενειακή κατάσταση του αιτούντα,** για τη χορήγηση της πιστωτικής κάρτας .

4.4.6 Συνοπτικοί πίνακες των τιμών των μετρικών αξιολόγησης με ή χωρίς υπερδειγματοληψία (με ή χωρίς oversampling)

Ταυτόχρονα, με την ανωτέρω διαδικασία, ακολουθήθηκαν και βήματα δημιουργίας μοντέλων μηχανικής μάθησης με δεδομένα χωρίς υπερδειγματοληψία (Παράρτημα Α').

Παρακάτω παρουσιάζουμε πίνακα περιεχόμενο από τις τιμές των μετρικών αξιολόγησης και για τα τρία μοντέλα με υπερδειγματοληψία:

Με OverSampling (SMOTE)									
Models	Μετρικές Αξιολόγησης					Confusion Matrix (CM)			
	Precision	Recall	F1 - score	Accuracy	AUC	TN	TP	FN	FP
Decision Trees	0.99	0.98	0.98	0.98	0.99	58.821	58.309	1.419	856
Random Forest	0.99	0.98	0.99	0.99	1.00	59.019	58.674	1.054	658
XG Boost	0.99	0.98	0.99	0.99	1.00	59.255	58.772	956	422

Πίνακας 9 Τελικά αποτελέσματα μοντέλων με Oversampling

Ενώ, ακολούθως παρουσιάζουμε πίνακα περιεχόμενο από τις τιμές των μετρικών αξιολόγησης και για τα τρία μοντέλα χωρίς υπερδειγματοληψία:

Χωρίς OverSampling									
Models	Μετρικές Αξιολόγησης					Confusion Matrix (CM)			
	Precision	Recall	F1 - score	Accuracy	AUC	TN	TP	FN	FP
Decision Trees	0.99	0.99	0.99	0.98	0.80	278	59.365	320	783
Random Forest	0.99	0.99	0.99	0.99	0.87	293	60.004	73	768
XG Boost	0.99	1.00	0.99	0.99	0.86	312	60.004	103	749

Πίνακας 10 Τελικά αποτελέσματα μοντέλων χωρίς Oversampling

Διαπιστώνουμε ότι οι μετρικές αξιολόγησης έχουν παρόμοιες τιμές και στις δύο περιπτώσεις, διαφέρουν μόνο στις τιμές AUC. Επίσης, οι τιμές στις μήτρες σύγχυσης (Confusion Matrix) διαφέρουν κατά πολύ στις κατηγορίες True Negative (TN), False Negative (FN) και False Positive (FP).

4.4.7 Εφαρμογή SHapley Additive exPlanations (SHAP) στο μοντέλο XGBoost

Από τον παραπάνω πίνακα (μετρικές αξιολόγησης) διαπιστώνουμε ότι το καλύτερο μοντέλο μας είναι το XGboost.

Έτσι, με την εφαρμογή των τιμών SHAP θα μπορέσουμε να δούμε πώς κάθε χαρακτηριστικό γνώρισμα επηρεάζει κάθε τελική πρόβλεψη, δηλαδή τη σημασία κάθε χαρακτηριστικού σε σύγκριση με άλλα και την εξάρτηση του μοντέλου στην αλληλεπίδραση μεταξύ των χαρακτηριστικών.

Αρχικά, δημιουργούμε ένα αντικείμενο επεξήγησης παρέχοντας ένα μοντέλο XGBoost και στη συνέχεια, θα υπολογίσουμε την τιμή SHAP χρησιμοποιώντας ένα σύνολο

δοκιμών.

Στη συνέχεια εφαρμόζουμε ένα ειδικό αλγόριθμο τον TreeExplainer στον XGBoost, ο οποίος μειώνει την πολυπλοκότητα από $O(T\ L\ 2^n)$ σε $O(T\ L\ D^2)$ σημείο που nείναι ο αριθμός των χαρακτηριστικών, T ο αριθμός των δέντρων, Lο μέγιστος αριθμός φύλλων και D το μέγιστο βάθος δέντρου. Αυτός μετατρέπει τη διαδικασία TreeExplainer σε μια διαδικασία με δυνατότητα επεξεργασίας που μπορεί στη συνέχεια να επιταχυνθεί με GPU. (Mark J. Bennett, 2020)

```
best=model3

explainer = shap.TreeExplainer(best)

X=final.iloc[:,3:].copy()
X
```

Και παίρνουμε ως αποτέλεσμα

	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE	NAME_FAMILY_STATUS	NAME_HOUSING_TYPE	FLAG_PHONE	CNT_FAM_MEMBERS	Age	MONTHS_BALANCE	STATUS	YEARS_EMPLOYED
16	76	4	4	1	1	0	2.0	58	0	1	
17	76	4	4	1	1	0	2.0	58	-1	1	
18	76	4	4	1	1	0	2.0	58	-2	1	
19	76	4	4	1	1	0	2.0	58	-3	1	
20	76	4	4	1	1	0	2.0	58	-4	1	
...	...	...	...	...	...	...	...	...	...	...	...
219168	76	4	4	3	4	0	1.0	25	-9	1	
219169	76	4	4	3	4	0	1.0	25	-10	0	
219170	76	4	4	3	4	0	1.0	25	-11	0	
219171	76	4	4	3	4	0	1.0	25	-12	1	
219172	76	4	4	3	4	0	1.0	25	-13	1	

202485 rows × 11 columns

Πίνακας 11 Αντικείμενο επεξήγησης (SHAP)

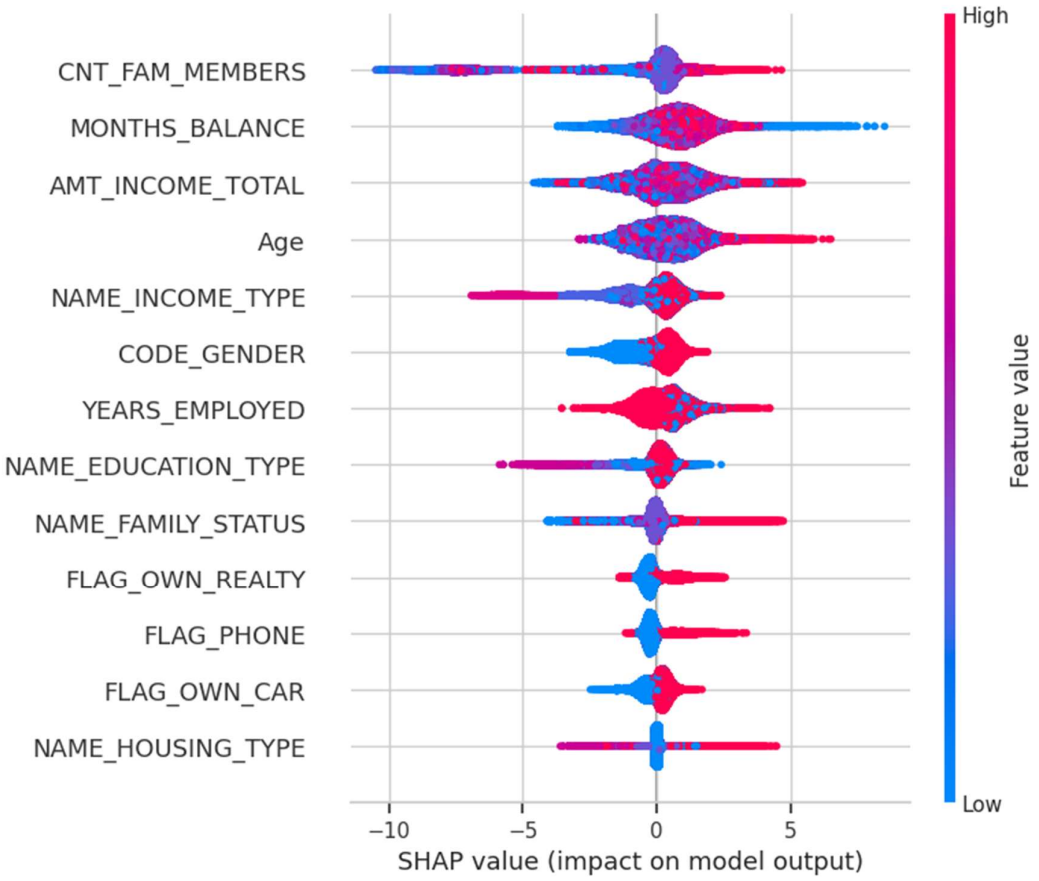
Ενώ με τη συνάρτηση θα υπολογίσουμε τις τιμές SHAP

```
shap_values = explainer.shap_values(X_test)
```

Και έπειτα θα εμφανίσουμε μέσω του διαγράμματος summary plot τις τιμές SHAP με χρήση και του σύνολο δοκιμών.

```
%matplotlib inline
fig=plt.figure(figsize=(21,8))
shap.summary_plot(shap_values, X_test, show=False)

plt.savefig('shap_summary.png', bbox_inches='tight')
```



Σχήμα 39 Σχέση τιμών γνωρισμάτων με τιμές SHAP

Η συνοπτική γραφική παράσταση (summary plot), αποκαλύπτει με ποιον τρόπο κάθε χαρακτηριστικό επηρεάζει την προβλεπόμενη έξοδο. Η χρωματική απόχρωση αναφέρεται στην τιμή του χαρακτηριστικού (κόκκινο υψηλό, μπλε χαμηλό). Αυτό δηλώνει για παράδειγμα ότι η ύπαρξη υψηλού βαθμού μειώνει την προβλεπόμενη πιθανότητα προεπιλογής (in'Analytics, 2021).

Από το παραπάνω διάγραμμα, βγάζουμε τα εξής συμπεράσματα:

- Ο άξονας Υ υποδεικνύει τα ονόματα των χαρακτηριστικών με σειρά σπουδαιότητας από πάνω προς τα κάτω.
- Ο άξονας Χ αντιπροσωπεύει την τιμή SHAP, η οποία υποδεικνύει το βαθμό αλλαγής στις πιθανότητες καταγραφής.
- Το χρώμα κάθε σημείου στο γράφημα αντιπροσωπεύει την τιμή του αντίστοιχου χαρακτηριστικού, με το κόκκινο να υποδεικνύει υψηλές τιμές και το μπλε να υποδηλώνει χαμηλές τιμές.
- Κάθε σημείο αντιπροσωπεύει μια σειρά δεδομένων από το αρχικό σύνολο δεδομένων.

Έτσι, στο συγκεκριμένο διάγραμμα το σπουδαιότερο γνώρισμα είναι ο αριθμός των

μελών της οικογένειας, όταν παίρνει χαμηλές τιμές δηλαδή η οικογένεια αποτελείται από πολύ λίγα μέλη (πχ εργένης ή μόνο το ζευγάρι ή ζευγάρι με ένα παιδί) οι τιμές SHAP value παίρνουν τιμές αρνητικές που σημαίνει ότι είναι δεν συμβάλουν θετικά στην πρόβλεψη του μοντέλου μας για την έγκριση της αίτησης.
Το αμέσως επόμενο σπουδαίο γνώρισμα είναι η μηνιαία ισορροπία. Όπως έχουμε ήδη προαναφέρει το γνώρισμα αυτό λαμβάνει τιμές από 0 όπου είναι ο μήνας των εξαγόμενων δεδομένων ως το σημείο εκκίνησης κινούμενο προς τα πίσω, δηλαδή είναι 0 είναι ο τρέχων μήνας της υποβολής της αίτησης, -1 είναι ο προηγούμενος μήνας και ούτω καθεξής. Οπότε, όταν αυτό το γνώρισμα παίρνει χαμηλές τιμές οι οποίες είναι αρνητικές, σημαίνει έχει περάσει μεγάλο χρονικό διάστημα από την ημερομηνία υποβολής της αίτησης με αποτέλεσμα το μοντέλο μας να θεωρεί ότι υπάρχει καθυστέρηση στην διαδικασία της έγκρισης της αίτησης, άρα συμβάλει θετικά στην πρόβλεψη του μοντέλου μας για την έγκριση της αίτησης, καθόσον η τιμή SHAP value παίρνει μεγάλες θετικές τιμές (>5).
Το γνώρισμα του συνολικού εισοδήματος του αιτούντα είναι το τρίτο σπουδαιότερο γνώρισμα. Από το διάγραμμα διαπιστώνουμε ότι είναι σημαντικό, καθόσον οι SHAP values πλησιάζουν πολύ κοντά στο 0. Στο σημείο αυτό, υπάρχουν υψηλές τιμές του γνωρίσματος αλλά και χαμηλές δηλαδή υπάρχουν άτομα που έκαναν αίτηση με χαμηλό αλλά και πολύ υψηλό εισόδημα. Ειδικά για την τελευταία κατηγορία, οι SHAP values πλησιάζουν πολύ κοντά στην τιμή 5.

5. Συμπέρασμα

Στόχος ήταν να προβλεφθεί αν θα εγκριθεί η αίτηση για χορήγηση πιστωτικής κάρτας.

Τα δεδομένα ήταν αρκετά μη ισορροπημένα (imbalanced data), Μέσω της βιβλιοθήκης SMOTE καταφέραμε να μετατρέψουμε τα δεδομένα σε ισορροπημένα. Εφαρμόσαμε στο μοντέλο μας αλγορίθμους όπως το Decision Tree Classifier, τον Random Forest Classifier και στο τέλος επιλέχθηκε να εφαρμοστεί ο αλγόριθμος XGBoost ένας από τους πιο ευρέως χρησιμοποιούμενους αλγόριθμους μηχανικής μάθησης ειδικά σε προβλήματα δυαδικής ταξινόμησης και παλινδρόμησης.

Η ίδια διαδικασία ακολουθήθηκε και χωρίς τη βιβλιοθήκη SMOTE, χωρίς δηλαδή υπερδειγματοληψία.

Στον αλγόριθμο Decision Tree Classifier, χρησιμοποιήσαμε τις υπερπαραμέτρους max_depth και min_samples_spli. Στον Random Forest Classifier, χρησιμοποιήσαμε τις υπερπαραμέτρους n_estimators, max_depth και min_samples_leaf. Ενώ στον αλγόριθμο XGBoost χρησιμοποιήσαμε τις υπερπαραμέτρους, max_depth, n_estimators, min_child_weight, subsample, learning_rate και seed.

Παρατηρώντας τα διαγράμματα σπουδαιότητας χαρακτηριστικών σε κάθε αλγόριθμο διαπιστώνουμε ότι :

-Στον Decision Tree Classifier, το χαρακτηριστικό το AMT_INCOME_TOTAL είναι αυτό που εμφανίζεται πολλές φορές στα δέντρα αποφάσεων. Από το ίδιο διάγραμμα, διακρίνεται και το χαρακτηριστικό Age, που δεν υπήρχε στο αρχικό σύνολο και δημιουργήθηκε κατά την προ-επεξεργασία των δεδομένων και CNT_FAM_MEMBERS.

- Στον Random Forest Classifier, το χαρακτηριστικό AMT_INCOME_TOTAL είναι αυτό που εμφανίζεται πολλές φορές στα δέντρα αποφάσεων. Από το ίδιο διάγραμμα, ακολουθούν τα χαρακτηριστικά το Age και MONTHS_BALANCE .

-Στον XGBoost, τα χαρακτηριστικά CNT_FAM_MEMBERS, CODE_GENDER, και NAME_FAMILY_STATUS.

Στο πρόβλημα αυτό, επιλέχθηκε να χρησιμοποιηθεί το f1-score και το accuracy για την αξιολόγηση των μοντέλων μηχανικής μάθησης που δημιουργήθηκαν:

- Στον Decision Tree Classifier, έδωσε **f1-score 0,98 (support 59728) και accuracy 0,98.**
- Στον Random Forest Classifier, έδωσε **f1-score 0,99 (support 59728) και accuracy 0,99.**
-Στον XGBoost, έδωσε **f1-score 0,99 (support 59728) και accuracy 0,99.**

Το Precision-Recall, είναι ένα χρήσιμο μέτρο για την επιτυχία της πρόβλεψης όταν τα δεδομένα είναι πολύ μη ισορροπημένα. Στην ανάκτηση πληροφοριών, η ακρίβεια (Precision) είναι ένα μέτρο της συνάφειας των αποτελεσμάτων, ενώ η ανάκληση (Recall) είναι ένα μέτρο για το πόσα πραγματικά σχετικά αποτελέσματα επιστρέφονται.

Και στους τρεις αλγορίθμους ζητήσαμε να μας δώσει τις τιμές precision και recall:

- Στον Decision Tree Classifier, έδωσε **precision με score 0.99 και recall με score 0.98.**
- Στον Random Forest Classifier έδωσε **precision με score 0.99 και recall με score 0.98.**
- Στον XGBoost, έδωσε **precision με score 0.99 και recall με score 0.98.**

Οι παραπάνω τιμές, συμβάλλουν να εξάγουμε το συμπέρασμα ότι και στις τρεις περιπτώσεις η ακρίβεια των αποτελεσμάτων είναι πολύ υψηλή αλλά και υπάρχει πλειονότητα θετικών αποτελεσμάτων.

Από τη θεωρία προκύπτει ότι σε ένα ιδανικό σύστημα όπου το precision είναι υψηλό και το recall έχει επίσης υψηλή τιμή, θα επιστρέψει πολλά αποτελέσματα, όπου το μεγαλύτερο πλήθος τους επισημαίνεται σωστά.

Παραπάνω, είδαμε ότι οι δύο τιμές precision και recall σχετίζονται επίσης με το f1-score .

Έπειτα, εφαρμόσαμε τις τιμές SHAP στο καλύτερο μοντέλο μας XGBoost, όπου διαπιστώσαμε ότι στο διάγραμμα τα γνωρίσματα ο αριθμός των μελών της οικογένειας, μηνιαία ισορροπία και το σύνολο εισοδήματος παίζουν σημαντικό ρόλο στον καθορισμό των αποτελεσμάτων.

Συγκεκριμένα, ο αριθμός των μελών της οικογένειας είναι κυρίως υψηλή με θετική τιμή SHAP που σημαίνει ότι όταν παίρνει χαμηλές τιμές δηλαδή η οικογένεια αποτελείται από πολύ λίγα μέλη (πχ εργένης ή μόνο το ζευγάρι ή ζευγάρι με ένα παιδί) οι τιμές SHAP value παίρνουν τιμές αρνητικές που σημαίνει ότι δεν συμβάλουν θετικά στην πρόβλεψη του μοντέλου μας .

Ενώ, η μηναία ισορροπία όταν παίρνει χαμηλές τιμές οι οποίες είναι αρνητικές, σημαίνει έχει περάσει μεγάλο χρονικό διάστημα από την ημερομηνία υποβολής της αίτησης με αποτέλεσμα το μοντέλο μας να θεωρεί ότι υπάρχει καθυστέρηση στην διαδικασία της έγκρισης της αίτησης, άρα συμβάλει θετικά στην πρόβλεψη του μοντέλου μας για την έγκριση της αίτησης, καθόσον η τιμή SHAP value παίρνει μεγάλες θετικές τιμές (>5).

Τέλος, το συνολικό εισόδημα του αιτούντα είναι επίσης σημαντικό, καθόσον οι SHAP values πλησιάζουν πολύ κοντά στο 0. Στο σημείο αυτό, υπάρχουν υψηλές τιμές του

γνωρίσματος αλλά και χαμηλές δηλαδή υπάρχουν άτομα που έκαναν αίτηση με χαμηλό αλλά και πολύ υψηλό εισόδημα. Ειδικά για την τελευταία κατηγορία, οι SHAP values πλησιάζουν πολύ κοντά στην τιμή 5.

Συμπερασματικά, στη μηχανική μάθηση χρειάζεται μεγάλο πλήθος δεδομένων, ώστε οι αλγόριθμοι μηχανικής μάθησης να μπορούν να μάθουν με μεγαλύτερη ακρίβεια να προβλέψουν αυτό που τους ζητείται. Πλέον στις μέρες μας, είναι υπερβολικά μεγάλος ο όγκος δεδομένων που μεταφέρεται, επεξεργάζεται και αποθηκεύεται από π.χ. διαφημιστικές, φαρμακευτικές εταιρείες. Και το φαινόμενο αυτό συνέχεια μεγεθύνεται μέρα με τη μέρα, χρόνια με τα χρόνια.

Στο αρχικό στάδιο, σημαντικό είναι να γνωρίσουμε τα δεδομένα μας όσο πιο αναλυτικά γίνεται έτσι ώστε να είμαστε σε θέση να επεξεργαστούμε με τον πιο ιδανικό τρόπο. Έπειτα θα ακολουθήσει μία προ-επεξεργασία τους όπου θα δούμε την καταλληλόλητά τους για την μηχανική μάθηση. Σίγουρο είναι ότι στο τέλος θα επεξεργαστούν με τέτοιο ώστε να μετατραπούν, όπου καταστεί δυνατό, για να είναι εύκολα αναγνώσιμα από τους αλγόριθμους της μηχανικής μάθησης.

Σε κάθε περίπτωση, απαραίτητη είναι τόσο η συγχώνευση δεδομένων όσο και η δημιουργία νέων παραμέτρων., οι οποίοι αρχικά δεν υφίστατο στα δεδομένα που είχαν εισαχθεί.

Στη μηχανική μάθηση, μία προϋπόθεση για την εύρεση καλύτερου αλγορίθμου είναι η επανάληψη. Γίνονται δοκιμές στις υπερπαραμέτρους των αλγορίθμων, ώστε να βρεθεί η καλύτερη δυνατή τιμή τους.

Έχοντας ως γνώμονα τις τιμές αυτές, ο κάθε αλγόριθμος θα μπορέσει με ακρίβεια να προβλέψει αυτό το οποίο θα του ζητηθεί.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import itertools
import warnings
warnings.filterwarnings('ignore')
import xgboost as xgb # XGBoost typically uses the alias "xgb"
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, roc_auc_score
from sklearn.model_selection import cross_val_score
from sklearn.metrics import f1_score, precision_score, recall_score
from imblearn.over_sampling import SMOTE
from sklearn.metrics import confusion_matrix
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder,OneHotEncoder
# Pre Processing
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn import preprocessing, svm

application=pd.read_csv('/kaggle/input/credit-card-approval-prediction/application_record.csv')
credit=pd.read_csv('/kaggle/input/credit-card-approval-prediction/credit_record.csv')

print(application.describe().T)

print(application.shape)

print(credit.shape)

credit.head()

application_info=application.info()
print(application_info)
print('\n')

application.isna().sum()


credit.isna().sum()


credit['STATUS'].value_counts()


application = application.drop_duplicates(subset=application.columns[1:], keep='first')

credit_info=credit.info()
```

```
print(credit_info)
print('\n')

application['Age']=-(application['DAYS_BIRTH'])//365
application['Age'].plot(kind='hist',bins=20,density=True)

application['Age']


application['CODE_GENDER'].replace('M',0,inplace=True)
application['CODE_GENDER'].replace('F',1,inplace=True)
application['FLAG_OWN_CAR'].replace('Y',0,inplace=True)
application['FLAG_OWN_CAR'].replace('N',1,inplace=True)
application['FLAG_OWN_REALTY'].replace('Y',0,inplace=True)
application['FLAG_OWN_REALTY'].replace('N',1,inplace=True)

def calc_day_of_birth (day_num):
    today = date.today()
    birthDay = (today + timedelta(days=day_num)).strftime('%Y-%m-%d')
    return birthDay


def get_appartment(x):
    if x == 'House / apartment' :
       x= x.split(' /')[0]
    return x

def get_ducational_type(x):
    if x == 'Secondary / secondary special' :
       x= x.split(' /')[0]
    return x

application["NAME_HOUSING_TYPE"].value_counts()


application['NAME_HOUSING_TYPE'] = application['NAME_HOUSING_TYPE'].apply(get_appartment)


application['NAME_EDUCATION_TYPE'] =application['NAME_EDUCATION_TYPE'].apply(get_ducational_type)


credit.info()

#plot counts of the target variable
class_sep=credit['STATUS'].value_counts().reset_index()
#barplot with matplotlib
fig, ax = plt.subplots()
fig.set_size_inches([25,7])
p1=ax.bar(class_sep['index'], class_sep['STATUS'], alpha=0.8)
ax.set_xticks(class_sep['index'])
ax.bar_label(p1)
plt.title("Distribution of the dependent variable (STATUS)")
plt.show()

print("The exact percentage of approved and not approved credit cards:")
credit['STATUS'].value_counts(normalize=True)
```

```
sns.set(style='whitegrid', context='notebook')
cols = ['AMT_INCOME_TOTAL', 'CNT_CHILDREN', 'CNT_FAM_MEMBERS', 'DAYS_BIRTH', 'DAYS_EMPLOYED']
sns.pairplot(application[cols], size=2.5);
plt.show()

plt.figure(figsize=(25,7))
sns.countplot(x=application.NAME_EDUCATION_TYPE)

new_status = {'C' : 1,
              'X' : 1,
              '0' : 1,
              '1' : 0,
              '2' : 0,
              '3' : 0,
              '4' : 0,
              '5' : 0}
credit['STATUS'] = credit['STATUS'].map(new_status)
credit['STATUS']=credit['STATUS'].astype(int)

credit['STATUS'].value_counts()


credit['STATUS'].value_counts(normalize = True)


application.info()
print('\n')
credit.info()

final = application.merge(credit, on=['ID'])


final

final.to_csv(r'C:\Users\dppel\Desktop\final4.csv', index=False, header=True)


final['YEARS_EMPLOYED']=-(final['DAYS_EMPLOYED'])//365

final.isnull().sum()

final = final.dropna(axis='columns')

final.info()

final.isna().sum()


final['NAME_INCOME_TYPE'].hist(figsize=(9,6))

final['AMT_INCOME_TOTAL'].hist()


final["NAME_HOUSING_TYPE"].value_counts()


final = final.drop(['ID', 'DAYS_BIRTH', 'FLAG_EMAIL', 'FLAG_WORK_PHONE', 'FLAG_MOBIL', 'CNT_CHILDREN','DAYS_EMPLOYED'],axis=1)
final
```

```
def dropOL(ftr):
    q75,q25 = np.percentile(final[ftr],[75,25])
    intr_qr = q75-q25
    mx = q75+(1.5*intr_qr)
    mn = q25-(1.5*intr_qr)
    return mx,mn

sns.displot(final, x="AMT_INCOME_TOTAL")


final.boxplot('AMT_INCOME_TOTAL')


mx,mn = dropOL('AMT_INCOME_TOTAL')


final.drop(final[final.AMT_INCOME_TOTAL > mx].index,inplace=True)


final.boxplot('YEARS_EMPLOYED')


mx,mn = dropOL('YEARS_EMPLOYED')


final.drop(final[final.YEARS_EMPLOYED > mx].index,inplace=True)


final.boxplot('CNT_FAM_MEMBERS')


mx,mn = dropOL('CNT_FAM_MEMBERS')


final.drop(final[final.CNT_FAM_MEMBERS > 6].index,inplace=True)


final.info()


features = ['AMT_INCOME_TOTAL','NAME_EDUCATION_TYPE', 'NAME_INCOME_TYPE','NAME_FAMILY_STATUS','NAME_HOUSING_TYPE']

le = preprocessing.LabelEncoder()

for col in features:
    final[col] = le.fit_transform(final[col])

final.info()

plt.figure(figsize=(32,10))
sns.heatmap(final.corr(),annot=True,cmap='RdYlGn')

Y = final['STATUS']
X = final.drop(['STATUS'], axis=1)

from imblearn.over_sampling import SMOTE
Y = Y.astype('int')
X_balance,Y_balance = SMOTE().fit_resample(X,Y)
X_balance = pd.DataFrame(X_balance, columns = X.columns)
```

```
X_train, X_test, y_train, y_test = train_test_split(X_balance,Y_balance,
                                                    test_size=0.3,
                                                       random_state = 10086)

X.info()

Y.info()


from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
decision_param_grid={"max_depth":[2,40,2],
"min_samples_split":[2,20,2]}
decision = DecisionTreeClassifier(random_state=123)
rand_search = GridSearchCV(estimator=decision, param_grid=decision_param_grid,
n_jobs=2, verbose=1, scoring='roc_auc')
rand_search.fit(X_train,y_train)
preds=rand_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", rand_search.best_params_)
print("Best score found:", rand_search.best_score_)

from sklearn.metrics import accuracy_score, confusion_matrix
model1 = DecisionTreeClassifier(max_depth=40,
                                min_samples_split=20,
                                random_state=1024)
model1.fit(X_train, y_train)
y_predict = model1.predict(X_test)

print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()


from sklearn import metrics
preds2 = model1.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds2)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)

plt.figure(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()
```

```
def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model1, X_train,20)


from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
random_param_grid={"n_estimators":[100,300,50],
                   "max_depth" : [2,20,2],
                   "min_samples_leaf":[2,20,2]}
random = RandomForestClassifier(random_state=123)
random_search = GridSearchCV(estimator=random, param_grid=random_param_grid,
n_jobs=2, verbose=1, scoring='roc_auc')
random_search.fit(X_train,y_train)
preds=rand_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", random_search.best_params_)
print("Best score found:", random_search.best_score_)


from sklearn.ensemble import RandomForestClassifier
model2 = RandomForestClassifier(n_estimators=300,
                                max_depth=20,
                                min_samples_leaf=2
                                )
model2.fit(X_train, y_train)
y_predict = model2.predict(X_test)

print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
confusion_matrix(y_test,y_predict)

cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, y_predict))
```

```
from sklearn import metrics
preds3 = model2.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds3)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)

plt.subplots(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()


def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model2, X_train,20)

import xgboost as xgb
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
xgb_param_grid={"n_estimators":[200,400, 1000],"max_depth":[5,10],"learning_rate":[0.05,0.1],"min_child_weight":[6,8]}
xgb = XGBClassifier(eval_metric="auc", random_state=123)
xgb_search = GridSearchCV(estimator=xgb, param_grid=xgb_param_grid,n_jobs=2, refit=True, verbose=3, scoring='roc_auc')
xgb_search.fit(X_train,y_train)
preds=xgb_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", xgb_search.best_params_)
print("Best score found:", xgb_search.best_score_)

from xgboost import XGBClassifier
model3 = XGBClassifier(max_depth=10,
                       n_estimators=1000,
                       min_child_weight=8,
                       learning_rate =0.1,
                       )
```

```
model3.fit(X_train, y_train)
y_predict = model3.predict(X_test)
print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, y_predict))


from sklearn import metrics
preds4 = model3.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds4)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)

plt.subplots(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()

auc = metrics.roc_auc_score(y_test, preds4)
plt.subplots(figsize=(10,6))
plt.plot(fpr,tpr,label="AUC="+str(auc))
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.legend(loc=4)
plt.title("The AUC curve")
plt.show()

def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model3, X_train,20)
```

```
import shap
from statsmodels.graphics import tsaplots

best=model3

explainer = shap.TreeExplainer(best)

X=final.iloc[:,3:].copy()
X

shap_values = explainer.shap_values(X_test)

%matplotlib inline
fig=plt.figure(figsize=(21,8))
shap.summary_plot(shap_values, X_test, show=False)

plt.savefig('shap_summary.png', bbox_inches='tight')
```

```
X_train, X_test, y_train, y_test = train_test_split(X,Y,
                                                    test_size=0.3,
                                                       random_state = 10086)

from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
decision_param_grid={"max_depth":[2,40,2],
"min_samples_split":[2,20,2]}
decision = DecisionTreeClassifier(random_state=123)
rand_search = GridSearchCV(estimator=decision, param_grid=decision_param_grid, n_jobs=2, verbose=1, scoring='roc_auc' )
rand_search.fit(X_train,y_train)
preds=rand_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", rand_search.best_params_)
print("Best score found:", rand_search.best_score_)

from sklearn.metrics import accuracy_score, confusion_matrix
model1 = DecisionTreeClassifier(max_depth=40,
                                min_samples_split=20,
                                random_state=1024)
model1.fit(X_train, y_train)
y_predict = model1.predict(X_test)

print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()

from sklearn import metrics
preds2 = model1.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds2)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)

plt.figure(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()
```

```
def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model1, X_train,20)

from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
random_param_grid={"n_estimators":[100,300,50],
                   "max_depth" : [2,20,2],
                   "min_samples_leaf":[2,20,2]}
random = RandomForestClassifier(random_state=123)
random_search = GridSearchCV(estimator=random, param_grid=random_param_grid,n_jobs=2, verbose=1, scoring='roc_auc')
random_search.fit(X_train,y_train)
preds=random_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", random_search.best_params_)
print("Best score found:", random_search.best_score_)

from sklearn.ensemble import RandomForestClassifier
model2 = RandomForestClassifier(n_estimators=300,
                                max_depth=20,
                                min_samples_leaf=2
                                )
model2.fit(X_train, y_train)
y_predict = model2.predict(X_test)

print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
confusion_matrix(y_test,y_predict)

cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, y_predict))

from sklearn import metrics
preds3 = model2.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds3)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)
```

```
plt.subplots(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()

def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model2, X_train,20)

import xgboost as xgb
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
xgb_param_grid={"n_estimators":[200,400, 1000],"max_depth":[5,10],"learning_rate":[0.05,0.1],"min_child_weight":[6,8]}
xgb = XGBClassifier(eval_metric="auc", random_state=123)
xgb_search = GridSearchCV(estimator=xgb, param_grid=xgb_param_grid,n_jobs=2, refit=True, verbose=3, scoring='roc_auc')
xgb_search.fit(X_train,y_train)
preds=xgb_search.predict(X_test)
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, preds))
print("Μια αρχική παρουσίαση του πίνακα σύγχυσης (confusion matrix)")
print(confusion_matrix(y_test, preds))
print("Best parameters found:", xgb_search.best_params_)
print("Best score found:", xgb_search.best_score_)
```

```
from xgboost import XGBClassifier
model3 = XGBClassifier(max_depth=10,
                       n_estimators=1000,
                       min_child_weight=8,
                       learning_rate =0.1,
                       )

model3.fit(X_train, y_train)
y_predict = model3.predict(X_test)
print('Accuracy Score is {:.5}'.format(accuracy_score(y_test, y_predict)))
cm=confusion_matrix(y_test,y_predict)
disp=ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=['Approved','Not Approved'])
fig, ax = plt.subplots(figsize=(10,6))
disp.plot(ax=ax)
plt.title("Ο πίνακας σύγχυσης (The confusion matrix)")
plt.ylabel("Approved")
plt.xlabel("Not Approved")
plt.show()
from sklearn.metrics import accuracy_score, f1_score,classification_report, confusion_matrix
print("Παρακάτω παρουσιάζεται η έκθεση ταξινόμησης (classification report)")
print(classification_report(y_test, y_predict))

from sklearn import metrics
preds4 = model3.predict_proba(X_test)[::,1]
fpr, tpr,thresold = metrics.roc_curve(y_test, preds4)
roc_auc=metrics.auc(fpr,tpr)
print('Roc_Auc: %0.2f' % roc_auc)

plt.subplots(figsize=(10,6))
plt.plot(fpr,tpr, 'b', label='AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0,1], [0,1], 'r--')
plt.plot([0,0], [1,0], c='.5')
plt.plot([1,1],  c='.5')
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.title("The ROC curve")
plt.show()

auc = metrics.roc_auc_score(y_test, preds4)
plt.subplots(figsize=(10,6))
plt.plot(fpr,tpr,label="AUC="+str(auc))
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.legend(loc=4)
plt.title("The AUC curve")
plt.show()
```

```
def plot_importance(classifer, x_train, point_size = 25):
    '''plot feature importance'''
    values = sorted(zip(x_train.columns, classifer.feature_importances_), key = lambda x: x[1] * -1)
    imp = pd.DataFrame(values,columns = ["Name", "Score"])
    imp.sort_values(by = 'Score',inplace = True)
    sns.scatterplot(x = 'Score',y='Name', linewidth = 0,
                data = imp,s = point_size, color='red').set(
    xlabel='importance',
    ylabel='features')

plot_importance(model3, X_train,20)

import shap
from statsmodels.graphics import tsaplots

best=model3

explainer = shap.TreeExplainer(best)

X=final.iloc[:,3:].copy()
X

shap_values = explainer.shap_values(X_test)

%matplotlib inline
fig=plt.figure(figsize=(21,8))
shap.summary_plot(shap_values, X_test, show=False)

plt.savefig('shap_summary.png', bbox_inches='tight')
```

Βιβλιογραφία

Activate State. (2020, 10 9). Ανάκτηση από https://www.activestate.com/resources/quick-reads/what-is-pandas-in-python-everything-you-need-to-know/.

Aji Gautama Putrada, E. K. (2022). *TPOT on Increasing The Performance of Credit.* 2nd International Conference on Intelligent Cybernetics Technology & Applications (ICICyTA).

Amazon Web Services, I. (2023). *AWS*. Ανάκτηση από https://docs.aws.amazon.com/sagemaker/latest/dg/xgboost-HowItWorks.html

Arya, N. (2022, 8 22). *KDnuggets*. Ανάκτηση από Tuning Random Forest Hyperparameters: https://www.kdnuggets.com/2022/08/tuning-random-forest-hyperparameters.html

Awan, A. A. (2023, 06). *Datacamp*. Ανάκτηση από Datacamp Tutorial: https://www.datacamp.com/tutorial/introduction-to-shap-values-machine-learning-interpretability

Banerjee, P. (2019). *Kaggle*. Ανάκτηση από Random Forest Classifier Tutorial with Python: https://www.kaggle.com/code/prashant111/random-forest-classifier-tutorial

Bhandari, A. (2023, 10 27). *Analytics Vidhya*. Ανάκτηση από Understanding & Interpreting Confusion Matrix in Machine Learning (Updated 2023): https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/#Why_Do_We_Need_a_Confusion_Matrix?

Card Insider. (2021, 08 31). Ανάκτηση από https://cardinsider.com/blog/anatomy-of-a-credit-card/

Cessie, S. v. (1992). Ridge estimators in logistic regression. Applied Statistics 41 (1).

Chauhan, N. S. (2022, 2 9). *KDnuggets*. Ανάκτηση από Decision Tree Algorithm, Explained: https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

contributors, A. I. (2015). What is the Jupyter Notebook?

Daniel, M. (2013). *Tableau Your Data.* Indianapolis: Wiley.

Daniel, M. (2016). *Tableau Your Data (2nd ed.).* Indianapolis: Wiley.

datacamp. (2022, 11). Ανάκτηση από https://www.datacamp.com/blog/what-is-python-used-for

David, D. (2020, 8 6). *FreeCodeCamp*. Ανάκτηση από Random Forest Classifier Tutorial: How

to Use Tree-Based Algorithms for Machine Learning: https://www.freecodecamp.org/news/how-to-use-the-tree-based-algorithm-for-machine-learning/

developers, c.-l. (2023). *scikit-learn*. Ανάκτηση από https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

developers, i.-l. (2023). *scikit-learn.org*. Ανάκτηση από Decision Trees: https://scikit-learn.org/stable/modules/tree.html

developers, s.-l. (2023). *scikit-learn*. Ανάκτηση από sklearn.ensemble.GradientBoostingClassifier: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

DocPlayer. (2015). Ανάκτηση από https://docplayer.gr/1661742-Aitisi-gia-horigisi-atomikis-pistotikis-kartas-mastercard.html

European Central Bank. (2022, 07 22). Ανάκτηση από https://www.ecb.europa.eu/press/pr/stats/paysec/html/ecb.pis2021~956efe1ee6.el.html

European Central Bank. (2023, 07 06). Ανάκτηση από https://data.ecb.europa.eu/publications/payments-statistics/3032563

European Central Bank. (2023, 09). Ανάκτηση από ECB Data Portal.

Florentin Butarua, ,. Q. (2016). *Risk and risk management in the credit card industry.* United States: Journal of Banking and Finance.

Galarnyk, M. (2022, 7 28). *builtin*. Ανάκτηση από Understanding Train Test Split: https://builtin.com/data-science/train-test-split

geeksforgeeks. (2023, 6 10). Ανάκτηση από Guide to AUC ROC Curve in Machine Learning: https://www.geeksforgeeks.org/auc-roc-curve/

II.Friedman, J. (2001). *Greedy Function Approximation: A Gradient Boosting Machine.* IMS Reitz Lecture.

in'Analytics, V. (2021, 04 20). *SHAP for Credit Risk: Interpreting Machine Learning Black Box*. Ανάκτηση από Medium: https://valooresanalyticsdept.medium.com/shap-for-credit-risk-interpreting-machine-learning-black-box-459a511e9e1e

John Hunter, D. D. (2012). *Matplotlib*. Ανάκτηση από https://matplotlib.org/

Kaggle. (2020). Ανάκτηση από https://www.kaggle.com/discussions/getting-started/44916

Kariuki, C. (2021, 12 9). *Section*. Ανάκτηση από https://www.section.io/engineering-education/imbalanced-learn-python-package-for-machine-learning/

Lemagnen, K. (2017, 9 11). *Medium*. Ανάκτηση από Getting started with XGBoost: https://blog.cambridgespark.com/getting-started-with-xgboost-3ba1488bb7d4?gi=b7c5c7e6bb5b

Ltd., B. P. (2023). *BigCommerce essentials*. Ανάκτηση από https://www.bigcommerce.com/ecommerce-answers/what-csv-file-and-what-does-it-mean-my-ecommerce-business/

Mark J. Bennett, J. A. (2020, 11 12). *Technical Blog* . Ανάκτηση από Nvidia Developer: https://developer.nvidia.com/blog/explaining-and-accelerating-machine-learning-for-loan-delinquencies/

MastersInDataScience.org. (2023). *Master's in Data Science with edX*. Ανάκτηση από How Does the Decision Tree Work?: https://www.mastersindatascience.org/learning/machine-learning-algorithms/decision-tree/

Mitchell, R. (2017, 9 11). *nvidia DEVELOPER*. Ανάκτηση από Gradient Boosting, Decision Trees and XGBoost with CUDA: https://developer.nvidia.com/blog/gradient-boosting-decision-trees-xgboost-cuda/

NumPy. (χ.χ.). Ανάκτηση από https://numpy.org/doc/stable/user/whatisnumpy.html

NVIDIA. (2023). *nvidia*. Ανάκτηση από What is XGBoost?: https://www.nvidia.com/en-us/glossary/data-science/xgboost/

O., P. (2022, 4). *Staxkeroverflow*. Ανάκτηση από scikit-learn random state in splitting dataset: https://stackoverflow.com/questions/42191717/scikit-learn-random-state-in-splitting-dataset

Olivier Chapelle, B. S. (2006). *Semi-Supervised Learning.* London, England: The MIT Press (Massachusetts Institute of Technology).

P. Tan, M. S. (2005). *Introduction to Data.* Addison Wesley.

Plotly. (2020). Ανάκτηση από https://plotly.com/python/

Polanitzer, R. (2022, 1 27). *Medium*. Ανάκτηση από Predict Credit Card Application Approval

with Python (Using Econometric Analysis Book by William H. Greene): https://medium.com/@polanitzer/predict-credit-card-application-approval-with-python-using-econometric-analysis-book-by-william-h-9999351955b9

Quinlan, J. (1993). *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufman.

S. Hemkiran, G. S. (2021). *Design of Automatic Credit Card Approval System Using Machine Learning.* Singapore: Springer.

SAP. (2023). Ανάκτηση από Artificial Intelligence: https://www.sap.com/greece/products/artificial-intelligence/what-is-machine-learning.html

SAP. (2023). Ανάκτηση από https://www.sap.com/greece/products/artificial-intelligence/what-is-machine-learning.html

scikit-learn, d. (2023). *scikit-learn*. Ανάκτηση από sklearn.tree.DecisionTreeClassifier: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

Simpliearn. (2022, 12 16). *Simpl;learn*. Ανάκτηση από https://www.simplilearn.com/tutorials/scikit-learn-tutorial/what-is-scikit-learn-and-how-to-install-it#what_is_scikit_learn

Stack Abuse. (2023, 8 7). Ανάκτηση από Understanding ROC Curves with Python: https://stackabuse.com/understanding-roc-curves-with-python/

To vima. (2022, 05 29). Ανάκτηση από https://www.tovima.gr/2022/05/29/finance/trapeza-tis-elladas-synexis-anodos-tis-xrisis-ton-karton-pliromis/

VISA. (2003). *Marketing. History of card products and acquiring, 2.4-2.6.* Visa International Service Association.

Waskom, M. (2023). *Seaborn*. Ανάκτηση από https://seaborn.pydata.org/tutorial/introduction.html

Weka (μηχανική μάθηση). (2021, 01 18). Ανάκτηση από Wikipedia: https://el.wikipedia.org/wiki/Weka_(%CE%BC%CE%B7%CF%87%CE%B1%CE%BD%CE%B9%CE%BA%CE%AE_%CE%BC%CE%AC%CE%B8%CE%B7%CF%83%CE%B7)

Wikipedia. (2021, 06). Ανάκτηση από https://en.wikipedia.org/wiki/Tableau_Software

Wikipedia. (2023, 9 26). Ανάκτηση από Gradient boosting: https://en.wikipedia.org/wiki/Gradient_boosting

Wikipedia. (2023, 8 25). Ανάκτηση από Ensemble learning: https://en.wikipedia.org/wiki/Ensemble_learning

Yengejeh, A. A. (2023). *Analysis of Cr Analysis of Credit Appr edit Approval by Decision T y Decision Tree* . Florida: University of Central Florida.

Ελληνική Ένωση Τραπεζών. (2022, 07 25). Ανάκτηση από https://www.hba.gr/ActivityAreas/Details/2292