Διεθνές Πανεπιστήμιο Ελλάδος
Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη εφαρμογής Ιστού για Πειράματα Αυτοματοποιημένης Μηχανικής Μάθησης

Φοιτητής:
Πρόδρομος Βεζυρίδης
Αριθμός Μητρώου: 2019018

Επιβλέπων:
Στέφανος Ουγίαρογλου

05 Ιουνίου 2026

Τίτλος Διπλωματικής Εργασίας: Ανάπτυξη εφαρμογής Ιστού για Πειράματα Αυτοματοποιημένης Μηχανικής Μάθησης
Κωδικός Διπλωματικής Εργασίας: 26208
Ονοματεπώνυμο Φοιτητή: Πρόδρομος Βεζυρίδης
Ονοματεπώνυμο Επιβλέποντος: Στέφανος Ουγίαρογλου
Ημερομηνία Ανάληψης: 23-03-2026
Ημερομηνία Ολοκλήρωσης: 05-06-2026

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

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

«Στους αγαπημένους μου»

Πρόλογος

Η παρούσα διπλωματική εργασία έχει ως αντικείμενο τη σχεδίαση και την υλοποίηση του `AutoML App`, μιας εύχρηστης διαδικτυακής εφαρμογής που αυτοματοποιεί τη διαδικασία της Μηχανικής Μάθησης (Automated Machine Learning - AutoML). Έχοντας παρακολουθήσει κατά τη διάρκεια των σπουδών μου στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων μαθήματα που άπτονται της Τεχνητής Νοημοσύνης και της Ανάλυσης Δεδομένων, διαπίστωσα ότι η χρήση των σύγχρονων εργαλείων AutoML απαιτεί συχνά προχωρημένες προγραμματιστικές γνώσεις σε Python και εξοικείωση με πολύπλοκα περιβάλλοντα λογισμικού. Αυτή η διαπίστωση αποτέλεσε το βασικό κίνητρο για τη δημιουργία μιας πλατφόρμας που θα γεφυρώνει αυτό το χάσμα, κάνοντας τη Μηχανική Μάθηση προσβάσιμη σε κάθε ενδιαφερόμενο.

Σκοπός της εφαρμογής είναι να προσφέρει ένα ενιαίο, γραφικό περιβάλλον όπου ο χρήστης μπορεί απλώς να εισάγει ένα αρχείο δεδομένων (CSV) και να εκτελεί παράλληλα τρία από τα σημαντικότερα open-source frameworks του χώρου: το H2O AutoML, το FLAML και το MLJAR. Το σύστημα αναλαμβάνει αυτόματα την εκπαίδευση, τη σύγκριση και την ανάδειξη του βέλτιστου μοντέλου, προσφέροντας παράλληλα τη δυνατότητα επαναχρησιμοποίησής του για νέες προβλέψεις, είτε μέσω της διεπαφής χρήστη (UI) είτε μέσω μιας ενσωματωμένης διαδικτυακής υπηρεσίας (Web Service).

Το όφελος από την εκπόνηση της παρούσας εργασίας είναι διττό. Από τη μία πλευρά, βοήθησε εμένα προσωπικά να αναπτύξω έναν σύνθετο τρόπο σκέψης και να εξελιχθώ ως μηχανικός λογισμικού, καθώς έπρεπε να συνδυάσω τη θεωρητική υποδομή της Μηχανικής Μάθησης με τεχνολογίες αιχμής για την ανάπτυξη διαδικτυακών συστημάτων (Full-Stack Web Development). Από την άλλη πλευρά, όπως φάνηκε και από την αξιολόγηση που πραγματοποιήθηκε με βάση την καθιερωμένη κλίμακα χρηστικότητας System Usability Scale (SUS), η εφαρμογή καταφέρνει να απλοποιήσει ουσιαστικά πολύπλοκες διαδικασίες, αποτελώντας ένα χρήσιμο και λειτουργικό εργαλείο για φοιτητές, ερευνητές ή αναλυτές που αναζητούν άμεση πρόσβαση σε τεχνολογίες AutoML.

Περίληψη

Οι τεχνολογίες Μηχανικής Μάθησης (Machine Learning) και Αυτοματοποιημένης Μηχανικής Μάθησης (AutoML) εφαρμόζονται ολοένα και περισσότερο για την επίλυση προβλημάτων ταξινόμησης και παλινδρόμησης σε ένα ευρύ φάσμα επιστημονικών και επιχειρηματικών τομέων. Ωστόσο, η πλειονότητα των υπαρχόντων πλαισίων εργασίας AutoML απαιτεί προγραμματιστικές δεξιότητες, βαθιά κατανόηση των εννοιών της μηχανικής μάθησης και εξοικείωση με εξειδικευμένα περιβάλλοντα λογισμικού, περιορίζοντας τη δυνατότητα εφαρμογής τους από μη ειδικούς χρήστες. Επιπλέον, διαφορετικές πλατφόρμες AutoML συχνά παράγουν διαφορετικά αποτελέσματα πάνω στα ίδια σύνολα δεδομένων, καθιστώντας την επιλογή της βέλτιστης λύσης μια σύνθετη διαδικασία.

Στην παρούσα διπλωματική εργασία, παρουσιάζεται το `AutoML App`, μια διαδικτυακή πλατφόρμα AutoML που στοχεύει στην απλοποίηση της διαδικασίας εκπαίδευσης, αξιολόγησης, σύγκρισης και επαναχρησιμοποίησης μοντέλων μηχανικής μάθησης μέσα από μια ενιαία και φιλική προς τον χρήστη διεπαφή. Το προτεινόμενο σύστημα ενσωματώνει πολλαπλά open-source πλαίσια AutoML, συμπεριλαμβανομένων των H2O AutoML, FLAML και MLJAR, υποστηρίζοντας εργασίες ταξινόμησης και παλινδρόμησης. Η πλατφόρμα επιτρέπει στους χρήστες να μεταφορτώνουν σύνολα δεδομένων σε μορφή CSV, να ορίζουν τη μεταβλητή-στόχο και τα χαρακτηριστικά εισόδου, να επιλέγουν τα επιθυμητά frameworks και τις μετρικές αξιολόγησης, και να εντοπίζουν αυτόματα το μοντέλο με την καλύτερη απόδοση μέσω άμεσης σύγκρισης.

Επιπλέον, η πλατφόρμα υποστηρίζει τη διατήρηση και επαναχρησιμοποίηση των μοντέλων, επιτρέποντας την παραγωγή προβλέψεων σε νέα, άγνωστα δεδομένα είτε μέσω της γραφικής διεπαφής ιστού είτε μέσω μιας ενσωματωμένης διαδικτυακής υπηρεσίας (Web Service). Το σύστημα αξιολογήθηκε μέσω πειραματικών μελετών με τη χρήση πολλαπλών συνόλων δεδομένων, ενώ η χρηστικότητά του αποτιμήθηκε από χρήστες με τη χρήση της κλίμακας SUS (System Usability Scale). Τα αποτελέσματα δείχνουν ότι το `AutoML App` αποτελεί μια αποτελεσματική και προσβάσιμη λύση για φοιτητές, ερευνητές, προγραμματιστές και μη εξειδικευμένους χρήστες που επιζητούν απλοποιημένη πρόσβαση στις τεχνολογίες AutoML.

Abstract

Machine Learning and Automated Machine Learning (AutoML) technologies are increasingly applied to solve classification and regression problems across a wide range of scientific and business domains. However, the majority of the existing AutoML frameworks require programming skills, understanding of machine learning concepts, and familiarity with specific software environments, limiting their applicability for non-expert users. Moreover, different AutoML frameworks often yield different results on the same datasets, making the selection of the best solution a difficult task.

In this thesis, we propose `AutoML App`, a web-based AutoML platform that aims to simplify the process of training, evaluating, comparing, and reusing machine learning models through a single, user-friendly interface. The proposed system integrates several open-source AutoML frameworks, including H2O AutoML, FLAML, and MLJAR, and supports classification and regression tasks. The platform allows users to upload datasets in CSV format, define target and feature variables, select desired frameworks and evaluation metrics, and automatically find the best-performing model by comparing them.

Furthermore, the platform supports model persistence and reuse, enabling predictions on new unseen datasets through a graphical web interface or an integrated web service. The system is evaluated through experimental studies using multiple datasets and performance metrics, while its usability is assessed using the System Usability Scale (SUS). The results show that `AutoML App` provides an effective and accessible solution for students, researchers, developers, and non-expert users seeking simplified access to AutoML technologies.

Περιεχόμενα

Πρόλογος **iii**

Περίληψη **iv**

Abstract **v**

1 Εισαγωγή **1**
1.1 Εποπτευμένη Μηχανική Μάθηση 1
1.1.1 Μετρικές Αξιολόγησης 1
1.1.2 Διαχωρισμός Δεδομένων και Γενίκευση 2
1.1.3 Βελτιστοποίηση Υπερπαραμέτρων 2
1.2 Αυτοματοποιημένη Μηχανική Μάθηση (AutoML) 2
1.3 Κίνητρο 3
1.4 Συνεισφορά 4
1.5 Οργάνωση της εργασίας 4

2 Πλατφόρμες Αυτοματοποιημένης Μηχανικής Μάθησης **6**
2.1 MLJAR 6
2.2 H2O AutoML 7
2.3 FLAML 7
2.4 Auto-sklearn 8
2.5 Auto-WEKA 9
2.6 Συγκριτική Αξιολόγηση Πλατφορμών AutoML 9

3 Εφαρμογές Ιστού Αυτοματοποιημένης Μηχανικής Μάθησης **11**
3.1 Εμπορικές Πλατφόρμες AutoML 11
3.2 AutoKNN 12
3.3 AutoDTrees 12
3.4 kClusterHub 12
3.5 WebApriori 12
3.6 Συζήτηση και Συσχέτιση με την Παρούσα Εργασία 13

4 Γλώσσες και Τεχνολογίες Ανάπτυξης **14**
4.1 Τεχνολογίες Διεπαφής Χρήστη (Frontend) 14
4.2 Τεχνολογίες Backend και Διαχείρισης Δεδομένων 15
4.3 Γλώσσα Προγραμματισμού Python και Πλαίσια AutoML 15

4.4 Εργαλεία Φιλοξενίας και Ελέγχου Εκδόσεων 16

5 Πειραματική Σύγκριση των Πλατφόρμων Αυτοματοποιημένης Μηχανικής Μάθησης 17
5.1 Σύνολα Δεδομένων (Datasets) 17
5.2 Τρόπος Διεξαγωγής των Πειραμάτων (Experimental Setup) 18
5.3 Μετρικές Αξιολόγησης και Αποδοτικότητα 18
5.4 Αποτελέσματα και Ανάλυση 19
5.5 Συζήτηση και Συμπεράσματα Πειράματων 19

6 Σχεδίαση και Υλοποίηση του AutoML App 21
6.1 Ανάλυση Απαιτήσεων 21
6.1.1 Λειτουργικές Απαιτήσεις 21
6.1.2 Μη Λειτουργικές Απαιτήσεις 22
6.1.3 User Stories 22
6.2 Αρχιτεκτονική Συστήματος 23
6.2.1 Διάγραμμα Αρχιτεκτονικής Τεσσάρων Επιπέδων 23
6.2.2 Διάγραμμα Ροής Λειτουργίας (Flowchart) 24
6.3 Σχεδίαση Σχεσιακής Βάσης Δεδομένων 25
6.3.1 Οντότητες και Μοντέλο Οντοτήτων - Συσχετίσεων 25
6.3.2 Σχεσιακό Σχήμα και Ανάλυση Δομής Πινάκων 28
6.4 Υλοποίηση Backend Web API 28
6.4.1 Διαχείριση Χρηστών και Μηχανισμός Αυθεντικοποίησης 28
6.4.2 Δρομολόγηση Διεργασιών Εκπαίδευσης (Job Ingestion) 29
6.4.3 Διασύνδεση με το Επίπεδο Μηχανικής Μάθησης (Inference Integration) 30
6.4.4 Αρχιτεκτονική Endpoints 31
6.5 Υλοποίηση Επιπέδου Εκτέλεσης Python (Python Execution Layer) 32
6.5.1 Ασύγχρονος Συντονιστής Διεργασιών (Background Worker) 32
6.6 Υλοποίηση Επιπέδου AutoML (Python Backends) 33
6.6.1 Μηχανή FLAML (Fast Lightweight AutoML) 33
6.6.2 Μηχανή H2O AutoML 34
6.6.3 Μηχανή MLJAR AutoML 35
6.6.4 Μηχανισμός Παραγωγής Προβλέψεων (Inference Engine) 35
6.7 Υλοποίηση του Frontend 39
6.7.1 Αρχιτεκτονική Δομή και Οργάνωση Αρχείων 39
6.7.2 Δυναμική Διαχείριση Διεπαφής (dashboard.js) 40
6.7.3 Τοπική Ανάλυση Δεδομένων μέσω FileReader API 40
6.7.4 Ασύγχρονη Επικοινωνία και Μηχανισμός Polling 41
6.7.5 Ασφάλεια και Διαχείριση Client-Side Κατάστασης 41

7 Παρουσίαση της Εφαρμογής AutoML App 42
7.1 Αρχική Σελίδα και Πλοήγηση 42
7.2 Αυθεντικοποίηση Χρήστη 44
7.3 Ροή Εργασίας από την Οπτική του Χρήστη 45
7.4 Διαχείριση Πόρων και Λειτουργικές Ενότητες (Modules) 46

7.4.1 Modules Διαχείρισης (Datasets, Models, Predictions) 46
7.4.2 Εκπαίδευση Μοντέλου (Train Model) 46
7.4.3 Παραγωγή Προβλέψεων (Predict) 48
7.4.4 Τα Σύνολα Δεδομένων μου (My Datasets) 49
7.4.5 Τα Μοντέλα μου (My Models) 50
7.4.6 Οι Προβλέψεις μου (My Predictions) 51

8 Εμπειρία Χρήσης και Αξιολόγηση **52**
8.1 Η Κλίμακα Χρηστικότητας Συστημάτων (System Usability Scale - SUS) 52
8.2 Αποτελέσματα Αξιολόγησης και Συζήτηση 53

9 Συμπεράσματα και Μελλοντικές Επεκτάσεις **55**
9.1 Συμπεράσματα 55
9.2 Ουσιαστική Συνεισφορά της Εργασίας 56
9.3 Μελλοντικές Επεκτάσεις 56
9.3.1 Ενσωμάτωση Επιπλέον Frameworks 56
9.3.2 Ερμηνευσιμότητα Μοντέλων (Explainable AI) 56
9.3.3 Βελτιώσεις Κλιμακωσιμότητας (Scalability) 56
9.3.4 Εμπειρία Χρήστη και Experiment Tracking 57

Βιβλιογραφία **58**

Κατάλογος σχημάτων

6.1 Αρχιτεκτονική τεσσάρων επιπέδων του AutoML App. 24
6.2 Διάγραμμα ροής δεδομένων και ασύγχρονης εκτέλεσης στο AutoML App. . . . 24
6.3 Διάγραμμα Οντοτήτων - Συσχετίσεων (ER) της βάσης δεδομένων automl_platform. 27

7.1 Κεντρικό section (Hero) της αρχικής σελίδας. 42
7.2 Ενότητα Features (Χαρακτηριστικά). 43
7.3 Ενότητα How it Works (Πώς λειτουργεί). 43
7.4 Ενότητα About (Σχετικά με την εφαρμογή). 44
7.5 Διεπαφή Σύνδεσης (Login). 45
7.6 Διεπαφή Εγγραφής (Register). 45
7.7 Η διαδικασία προετοιμασίας της εκπαίδευσης: (a) Μεταφόρτωση και επισκόπηση του συνόλου δεδομένων και (b) Ρύθμιση παραμέτρων και αυτόματη επιλογή τύπου προβλήματος. 47
7.8 Αναδυόμενο παράθυρο (Modal) τελικών αποτελεσμάτων αξιολόγησης και κατάταξης των frameworks βάσει της επιλεγμένης μετρικής. 48
7.9 Η διαδικασία παραγωγής προβλέψεων (Inference): (a) Επιλογή εκπαιδευμένου μοντέλου και μεταφόρτωση νέου συνόλου δεδομένων και (b) Προεπισκόπηση αποτελεσμάτων πρόβλεψης και αυτόματος υπολογισμός μετρικών αξιολόγησης. 49
7.10 Πίνακας διαχείρισης και επισκόπησης ιδιωτικών και δημόσιων συνόλων δεδομένων. 50
7.11 Διεπαφή διαχείρισης, λήψης και επισκόπησης των βέλτιστων εκπαιδευμένων μοντέλων. 50
7.12 Ιστορικό αρχείο αποθηκευμένων αποτελεσμάτων πρόβλεψης και λήψης εξαγόμενων αρχείων. 51

Κατάλογος πινάκων

2.1 Σύγκριση των κυριότερων AutoML frameworks 10

5.1 Χαρακτηριστικά των συνόλων δεδομένων της πειραματικής αξιολόγησης. . . . 18
5.2 Συνολικά αποτελέσματα πειραματικής σύγκρισης υπό σταθερούς χρονικούς περιορισμούς. 19

6.1 Σύνοψη Πινάκων, Κλειδιών και Συσχετίσεων της Βάσης Δεδομένων. 28

8.1 Αποτελέσματα της Κλίμακας Χρηστικότητας Συστημάτων (SUS) για την εφαρμογή AutoML App 53

Κεφάλαιο 1

Εισαγωγή

1.1 Εποπτευμένη Μηχανική Μάθηση

Η Εποπτευμένη Μηχανική Μάθηση (Supervised Machine Learning) αποτελεί τον πιο διαδεδομένο κλάδο της Τεχνητής Νοημοσύνης σε πραγματικές εφαρμογές [1, 2]. Ο κύριος στόχος της είναι η ανάπτυξη υπολογιστικών μοντέλων τα οποία εκπαιδεύονται πάνω σε ένα σύνολο δεδομένων που περιέχει ήδη τις «σωστές απαντήσεις» (labels). Μέσω αυτής της διαδικασίας, ο αλγόριθμος επιχειρεί να «μάθει» τα πρότυπα που συνδέουν τα δεδομένα εισόδου με τα αποτελέσματα εξόδου, εξασφαλίζοντας υψηλή ακρίβεια στις προβλέψεις για νέα, άγνωστα δεδομένα.

Η παραδοσιακή προσέγγιση της εποπτευμένης μάθησης απαιτεί τον πλήρη έλεγχο της ροής εργασίας από τον αναλυτή δεδομένων. Η διαδικασία ξεκινά με τη συλλογή των δεδομένων και συνεχίζεται με την προεπεξεργασία τους, όπου τα ακατέργαστα δεδομένα μετατρέπονται σε μορφή κατάλληλη για τον αλγόριθμο. Αυτό περιλαμβάνει τον καθαρισμό από θόρυβο, τη διαχείριση ελλιπών τιμών (missing values) και την κανονικοποίηση των χαρακτηριστικών.

Στη συνέχεια, ο μηχανικός καλείται να επιλέξει τον καταλληλότερο αλγόριθμο ανάμεσα σε μια πληθώρα επιλογών, όπως τα Δέντρα Απόφασης (Decision Trees), οι Μηχανές Διανυσμάτων Υποστήριξης (SVM) ή τα Τυχαία Δάση (Random Forests). Η επιλογή αυτή, σε συνδυασμό με τη ρύθμιση των υπερπαραμέτρων (hyperparameter tuning), αποτελεί μια επίπονη διαδικασία δοκιμής και σφάλματος (trial and error). Μια λανθασμένη ρύθμιση μπορεί να οδηγήσει σε χαμηλή απόδοση ή στο φαινόμενο της υπερπροσαρμογής (overfitting) [1].

Ανάλογα με τη φύση της μεταβλητής-στόχου (target variable), τα προβλήματα της εποπτευμένης μάθησης κατηγοριοποιούνται σε δύο κύριες κατευθύνσεις:

- **Ταξινόμηση (Classification):** Χρησιμοποιείται όταν η έξοδος είναι κατηγορική.
- **Παλινδρόμηση (Regression):** Χρησιμοποιείται όταν η έξοδος είναι συνεχής αριθμητική τιμή.

1.1.1 Μετρικές Αξιολόγησης

Για να διαπιστωθεί η ικανότητα του μοντέλου να γενικεύει σε νέα δεδομένα, είναι απαραίτητη η χρήση συγκεκριμένων μετρικών αξιολόγησης. Στα προβλήματα ταξινόμησης, το βασικότερο

εργαλείο ανάλυσης είναι ο Πίνακας Σύγχυσης (Confusion Matrix).
Με βάση αυτόν, ορίζονται μετρικές όπως η Ακρίβεια (Accuracy), η Πιστότητα (Precision), η Ανάκληση (Recall) και το F1-Score [3].

Αντίστοιχα, στα προβλήματα παλινδρόμησης (Regression), χρησιμοποιούνται διαφορετικές μετρικές αξιολόγησης, όπως το Μέσο Απόλυτο Σφάλμα (Mean Absolute Error - MAE), το Μέσο Τετραγωνικό Σφάλμα (Mean Squared Error - MSE), η Ρίζα του Μέσου Τετραγωνικού Σφάλματος (Root Mean Squared Error - RMSE), καθώς και ο Συντελεστής Προσδιορισμού R^2, οι οποίες αποτυπώνουν την απόκλιση των προβλέψεων από τις πραγματικές τιμές και την ικανότητα του μοντέλου να εξηγεί τη διακύμανση των δεδομένων [2, 1].

1.1.2 Διαχωρισμός Δεδομένων και Γενίκευση

Η ικανότητα ενός μοντέλου να αποδίδει σωστά σε νέα δεδομένα ονομάζεται **γενίκευση (generalization)** [1]. Για να διασφαλιστεί αυτό, το σύνολο δεδομένων χωρίζεται σε σύνολο εκπαίδευσης και σύνολο ελέγχου [4].

Κατά τη διαδικασία αυτή, πρέπει να αποφευχθεί η **υποπροσαρμογή (underfitting)** και η **υπερπροσαρμογή (overfitting)** [1].

1.1.3 Βελτιστοποίηση Υπερπαραμέτρων

Η εύρεση των ιδανικών υπερπαραμέτρων είναι κρίσιμη και συχνά πραγματοποιείται μέσω τεχνικών όπως η αναζήτηση πλέγματος ή η τυχαία αναζήτηση [5].

1.2 Αυτοματοποιημένη Μηχανική Μάθηση (AutoML)

Η Αυτοματοποιημένη Μηχανική Μάθηση (Automated Machine Learning - AutoML) αποτελεί ένα σύγχρονο ερευνητικό πεδίο της Τεχνητής Νοημοσύνης, το οποίο στοχεύει στην αυτοματοποίηση της διαδικασίας ανάπτυξης μοντέλων μηχανικής μάθησης, μειώνοντας την ανάγκη για εκτενή ανθρώπινη παρέμβαση [6, 7, 8].

Στην κλασική προσέγγιση της εποπτευμένης μάθησης, η επιλογή αλγορίθμων, η προεπεξεργασία δεδομένων και η ρύθμιση υπερπαραμέτρων απαιτούν σημαντική εμπειρία και χρόνο από τον μηχανικό δεδομένων. Το AutoML επιχειρεί να αυτοματοποιήσει αυτά τα στάδια, καθιστώντας τη μηχανική μάθηση πιο προσβάσιμη και αποδοτική [6, 8].

Ένα τυπικό σύστημα AutoML περιλαμβάνει επιμέρους διαδικασίες όπως [8]:

- Επιλογή αλγορίθμου (model selection)
- Προεπεξεργασία δεδομένων (data preprocessing)
- Βελτιστοποίηση υπερπαραμέτρων (hyperparameter optimization)
- Αξιολόγηση και επιλογή τελικού μοντέλου

Σύγχρονες προσεγγίσεις AutoML χρησιμοποιούν τεχνικές όπως Bayesian Optimization, evolutionary algorithms και reinforcement learning για την αναζήτηση του βέλτιστου μοντέλου στον χώρο των πιθανών λύσεων [5, 9, 7].

Στην πράξη, η υλοποίηση AutoML συστημάτων πραγματοποιείται μέσω εξειδικευμένων λογισμικών πλαισίων (AutoML frameworks), τα οποία διαφέρουν ως προς τις τεχνικές και τις στρατηγικές αναζήτησης που χρησιμοποιούν [6, 8]. Ενδεικτικά παραδείγματα περιλαμβάνουν:

- **Auto-sklearn** [7] και **FLAML** [10, 11]: βασίζονται κυρίως σε Bayesian optimization [9] και efficient search strategies.
- **Auto-WEKA** [12]: συνδυάζει την επιλογή αλγορίθμου με τη βελτιστοποίηση υπερπαραμέτρων στο περιβάλλον της πλατφόρμας WEKA, χρησιμοποιώντας Bayesian optimization.
- **TPOT** [13] και **GAMA** [14]: χρησιμοποιούν evolutionary algorithms για εξερεύνηση pipelines.
- **H2O AutoML** [15], **MLJAR** [16] και **AutoGluon** [17]: παρέχουν αυτοματοποιημένες end-to-end λύσεις για classification και regression.
- **Auto-Keras** [18] και **Auto-PyTorch** [19]: βασίζονται σε neural architecture search (NAS).
- **NASLib** [20]: ερευνητικό framework για neural architecture search.
- **LightAutoML** [21]: optimized framework για γρήγορη παραγωγή μοντέλων σε tabular data.
- **Auto-TS** [22]: εξειδικευμένο σε time series forecasting.
- **FEDOT** [23]: pipeline-based AutoML framework με εξελικτική βελτιστοποίηση.
- **TransmogrifAI** [24]: AutoML πλατφόρμα της Salesforce για enterprise εφαρμογές.

Το AutoML έχει ιδιαίτερη σημασία σε εφαρμογές όπου απαιτείται γρήγορη ανάπτυξη μοντέλων ή όταν η διαθεσιμότητα εξειδικευμένων μηχανικών μηχανικής μάθησης είναι περιορισμένη. Παράλληλα, συμβάλλει στη μείωση του χρόνου ανάπτυξης και στη βελτίωση της απόδοσης μέσω συστηματικής αναζήτησης βέλτιστων παραμέτρων [6, 8].

Ωστόσο, παρά τα πλεονεκτήματα του, το AutoML δεν αντικαθιστά πλήρως τον ανθρώπινο παράγοντα, καθώς ζητήματα όπως η ερμηνευσιμότητα των μοντέλων και η επιλογή κατάλληλων μετρικών αξιολόγησης εξακολουθούν να απαιτούν ανθρώπινη καθοδήγηση [6].

1.3 Κίνητρο

Παρά τη σημαντική πρόοδο της μηχανικής μάθησης, η διαδικασία ανάπτυξης και αξιοποίησης μοντέλων παραμένει ιδιαίτερα απαιτητική για τον μέσο χρήστη. Η ανάγκη για προγραμματιστικές γνώσεις, η κατανόηση σύνθετων εννοιών, καθώς και η χειροκίνητη διαχείριση των επιμέρους σταδίων (προεπεξεργασία δεδομένων, επιλογή αλγορίθμων, βελτιστοποίηση υπερπαραμέτρων) προϋποθέτουν εξειδικευμένες γνώσεις και σημαντική εμπειρία, περιορίζοντας

την πρόσβαση σε έναν στενό κύκλο ειδικών. Πέρα από το τεχνικό εμπόδιο, ο παράγοντας του χρόνου αποτελεί κρίσιμο αποτρεπτικό παράγοντα. Η παραδοσιακή διαδικασία δοκιμής και σφάλματος (trial and error) για την εύρεση της βέλτιστης αρχιτεκτονικής μοντέλου είναι εξαιρετικά χρονοβόρα, καθώς απαιτεί επαναληπτικές δοκιμές σε τεράστια υπολογιστικά διαστήματα παραμέτρων. Η έλλειψη εργαλείων που να αυτοματοποιούν αυτά τα στάδια επιβραδύνει τη λήψη αποφάσεων σε κρίσιμα επιχειρησιακά σενάρια και καθιστά την αξιοποίηση της μηχανικής μάθησης μια επίπονη και μη αποδοτική διαδικασία.

Αν και οι σύγχρονες προσεγγίσεις AutoML στοχεύουν στην αυτοματοποίηση αυτών των διαδικασιών, η πρόσβαση σε τέτοια εργαλεία συχνά παραμένει περιορισμένη ή απαιτεί εξειδικευμένες γνώσεις για την αποτελεσματική αξιοποίησή τους. Επιπλέον, η απουσία φιλικών προς τον χρήστη διεπαφών περιορίζει τη χρήση τους από μη ειδικούς, όπως αναλυτές δεδομένων χωρίς προγραμματιστικό υπόβαθρο ή επαγγελματίες άλλων πεδίων.

Με βάση τα παραπάνω, προκύπτει η ανάγκη για την ανάπτυξη συστημάτων που καθιστούν τη μηχανική μάθηση πιο προσβάσιμη και εύχρηστη, επιτρέποντας σε χρήστες χωρίς προγραμματιστικές γνώσεις να αξιοποιούν τις δυνατότητές της. Η ανάπτυξη τέτοιων συστημάτων διευκολύνει την πρόσβαση και την αξιοποίηση της μηχανικής μάθησης από ευρύτερες κατηγορίες χρηστών.

1.4 Συνεισφορά

Η παρούσα διπλωματική εργασία επικεντρώνεται στην ανάπτυξη ενός διαδικτυακού συστήματος Αυτοματοποιημένης Μηχανικής Μάθησης (AutoML App), το οποίο επιτρέπει σε χρήστες χωρίς προγραμματιστικές γνώσεις να εκπαιδεύουν, να αξιολογούν και να αξιοποιούν μοντέλα μηχανικής μάθησης μέσω ενός φιλικού γραφικού περιβάλλοντος.

Η προτεινόμενη εφαρμογή παρέχει τη δυνατότητα φόρτωσης δεδομένων σε μορφή CSV, αυτόματης προεπεξεργασίας των δεδομένων, εκπαίδευσης πολλαπλών μοντέλων μηχανικής μάθησης και επιλογής του βέλτιστου μοντέλου με βάση συγκεκριμένες μετρικές αξιολόγησης. Επιπλέον, υποστηρίζει την άμεση χρήση των εκπαιδευμένων μοντέλων για προβλέψεις πάνω σε νέα δεδομένα.

Κύρια συνεισφορά της εργασίας αποτελεί η σχεδίαση και υλοποίηση μιας ολοκληρωμένης web εφαρμογής AutoML, η οποία ενοποιεί βασικά στάδια της μηχανικής μάθησης σε ένα ενιαίο και αυτοματοποιημένο περιβάλλον, μειώνοντας την ανάγκη για εξειδικευμένες τεχνικές γνώσεις.

Η εφαρμογή που αναπτύχθηκε στο πλαίσιο της εργασίας διατίθεται διαδικτυακά μέσω της πλατφόρμας KClusterHub στη διεύθυνση `https://kclusterhub.iee.ihu.gr/automl/`, ενισχύοντας την προσβασιμότητα και τη χρηστικότητα της μηχανικής μάθησης σε πρακτικά σενάρια.

1.5 Οργάνωση της εργασίας

Η παρούσα διπλωματική εργασία οργανώνεται σε εννέα κεφάλαια, τα οποία καλύπτουν τόσο το θεωρητικό υπόβαθρο της Αυτοματοποιημένης Μηχανικής Μάθησης όσο και τη σχεδίαση, υλοποίηση και αξιολόγηση της προτεινόμενης εφαρμογής.

Στο Κεφάλαιο 1 παρουσιάζονται οι βασικές έννοιες της Εποπτευμένης Μηχανικής Μάθησης και της Αυτοματοποιημένης Μηχανικής Μάθησης (AutoML), καθώς και το κίνητρο και η συνεισφορά της παρούσας εργασίας.

Στο Κεφάλαιο 2 πραγματοποιείται αναλυτική παρουσίαση δημοφιλών πλατφορμών AutoML ανοικτού κώδικα, όπως τα MLJAR, H2O AutoML, FLAML και Auto-sklearn, με έμφαση στα χαρακτηριστικά και τις τεχνικές βελτιστοποίησης που χρησιμοποιούν.

Στο Κεφάλαιο 3 παρουσιάζονται υπάρχουσες διαδικτυακές εφαρμογές και συστήματα που σχετίζονται με την Αυτοματοποιημένη Μηχανική Μάθηση, καθώς και εφαρμογές που έχουν αναπτυχθεί στο πλαίσιο ακαδημαϊκών και ερευνητικών δραστηριοτήτων.

Στο Κεφάλαιο 4 αναλύονται οι γλώσσες προγραμματισμού, οι τεχνολογίες και τα λογισμικά εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής AutoML App.

Στο Κεφάλαιο 5 παρουσιάζεται η πειραματική σύγκριση των AutoML πλατφορμών μέσω διαφορετικών συνόλων δεδομένων και μετρικών αξιολόγησης, ενώ αναλύονται τα αποτελέσματα που προέκυψαν.

Στο Κεφάλαιο 6 περιγράφεται η σχεδίαση και η υλοποίηση της εφαρμογής AutoML App, συμπεριλαμβανομένης της αρχιτεκτονικής του συστήματος, της βάσης δεδομένων, του backend, του frontend και των WEB API υπηρεσιών.

Στο Κεφάλαιο 7 παρουσιάζονται αναλυτικά οι βασικές λειτουργίες της εφαρμογής μέσω στιγμιότυπων οθόνης και παραδειγμάτων χρήσης.

Στο Κεφάλαιο 8 παρουσιάζεται η αξιολόγηση της εμπειρίας χρήσης της εφαρμογής μέσω του ερωτηματολογίου SUS (System Usability Scale), καθώς και η ανάλυση των αποτελεσμάτων που προέκυψαν.

Τέλος, στο Κεφάλαιο 9 παρουσιάζονται τα συμπεράσματα της εργασίας και προτείνονται πιθανές μελλοντικές επεκτάσεις και βελτιώσεις της εφαρμογής.

Κεφάλαιο 2

Πλατφόρμες Αυτοματοποιημένης Μηχανικής Μάθησης

Η ραγδαία ανάπτυξη της Αυτοματοποιημένης Μηχανικής Μάθησης (AutoML) οδήγησε στη δημιουργία πληθώρας λογισμικών πλαισίων και εργαλείων, τα οποία στοχεύουν στην αυτοματοποίηση των βασικών σταδίων της διαδικασίας ανάπτυξης μοντέλων μηχανικής μάθησης. Οι πλατφόρμες αυτές επιτρέπουν την αυτόματη επιλογή αλγορίθμων, τη βελτιστοποίηση υπερπαραμέτρων και την αξιολόγηση μοντέλων, μειώνοντας σημαντικά την ανάγκη ανθρώπινης παρέμβασης [6].

Κάθε πλατφόρμα AutoML ακολουθεί διαφορετική στρατηγική αναζήτησης και βελτιστοποίησης, αξιοποιώντας τεχνικές όπως Bayesian Optimization, evolutionary algorithms, ensemble learning και meta-learning. Επιπλέον, τα διαθέσιμα frameworks διαφέρουν ως προς την υπολογιστική πολυπλοκότητα, την ταχύτητα εκπαίδευσης, την ερμηνευσιμότητα των μοντέλων και την ευκολία χρήσης.

Στο παρόν κεφάλαιο παρουσιάζονται αναλυτικά πέντε από τις σημαντικότερες πλατφόρμες ανοικτού κώδικα στον χώρο του AutoML. Συγκεκριμένα, αναλύονται τα frameworks MLJAR, H2O AutoML, FLAML, Auto-sklearn και Auto-WEKA, με έμφαση στη λειτουργία τους, στις τεχνικές βελτιστοποίησης που χρησιμοποιούν, καθώς και στη συνάφειά τους με την προτεινόμενη εφαρμογή.

2.1 MLJAR

Το MLJAR αποτελεί μία σύγχρονη πλατφόρμα Αυτοματοποιημένης Μηχανικής Μάθησης ανοικτού κώδικα, η οποία σχεδιάστηκε με στόχο την απλοποίηση της διαδικασίας ανάπτυξης μοντέλων μηχανικής μάθησης, δίνοντας ιδιαίτερη έμφαση στην ερμηνευσιμότητα και στην αυτοματοποίηση των επιμέρους σταδίων της διαδικασίας [16]. Το framework υποστηρίζει τόσο προβλήματα ταξινόμησης όσο και παλινδρόμησης, ενώ είναι σχεδιασμένο ώστε να μπορεί να χρησιμοποιηθεί ακόμη και από χρήστες με περιορισμένη εμπειρία στον χώρο της μηχανικής μάθησης.

Η λειτουργία του MLJAR βασίζεται στη δημιουργία πλήρως αυτοματοποιημένων pipelines μηχανικής μάθησης. Κατά τη διαδικασία εκπαίδευσης, το framework αναλαμβάνει αυτόματα την

προεπεξεργασία των δεδομένων, τη διαχείριση ελλιπών τιμών, την επιλογή χαρακτηριστικών και τη βελτιστοποίηση υπερπαραμέτρων. Παράλληλα, εκπαιδεύει πολλαπλά μοντέλα και συγκρίνει την απόδοσή τους με βάση συγκεκριμένες μετρικές αξιολόγησης.

Ένα από τα σημαντικότερα χαρακτηριστικά του MLJAR είναι η υποστήριξη διαφορετικών τρόπων λειτουργίας (modes), οι οποίοι προσαρμόζονται στις ανάγκες του χρήστη. Ενδεικτικά, η λειτουργία "Explain" δίνει έμφαση στην ερμηνευσιμότητα των μοντέλων, η λειτουργία "Perform" στοχεύει στη βελτιστοποίηση της απόδοσης, ενώ η λειτουργία "Compete" προσανατολίζεται σε πιο απαιτητικά σενάρια [16].

Επιπλέον, το MLJAR παρέχει αναλυτικές αναφορές αξιολόγησης, οι οποίες περιλαμβάνουν μετρικές απόδοσης, διαγράμματα σημαντικότητας χαρακτηριστικών και εργαλεία ερμηνείας των αποτελεσμάτων. Η δυνατότητα αυτή συμβάλλει σημαντικά στην κατανόηση της συμπεριφοράς των μοντέλων και καθιστά το framework ιδιαίτερα χρήσιμο σε εφαρμογές όπου η ερμηνευσιμότητα αποτελεί κρίσιμο παράγοντα.

Στο πλαίσιο της παρούσας εργασίας, το MLJAR χρησιμοποιήθηκε ως μία από τις βασικές πλατφόρμες του συστήματος AutoML App, λόγω της υψηλής απόδοσης που παρουσιάζει σε προβλήματα classification και regression, καθώς και λόγω της ευκολίας ενσωμάτωσής του σε διαδικτυακές εφαρμογές.

2.2 H2O AutoML

Το H2O AutoML αποτελεί μία από τις πιο δημοφιλείς και διαδεδομένες πλατφόρμες Αυτοματοποιημένης Μηχανικής Μάθησης ανοικτού κώδικα [15]. Το framework αναπτύχθηκε από την εταιρεία H2O.ai και σχεδιάστηκε με στόχο την αυτοματοποίηση της διαδικασίας εκπαίδευσης και βελτιστοποίησης μοντέλων μηχανικής μάθησης σε περιβάλλοντα μεγάλης κλίμακας.

Η πλατφόρμα υποστηρίζει πληθώρα αλγορίθμων, όπως Gradient Boosting Machines, Random Forests, Generalized Linear Models και XGBoost, ενώ παράλληλα δημιουργεί ensembles μέσω τεχνικών stacked ensembling [15]. Η χρήση ensemble μοντέλων επιτρέπει τη συνδυαστική αξιοποίηση των πλεονεκτημάτων διαφορετικών αλγορίθμων, οδηγώντας συχνά σε βελτιωμένη προγνωστική απόδοση.

Ένα από τα βασικά χαρακτηριστικά του H2O AutoML είναι η δυνατότητα κατανεμημένης επεξεργασίας (distributed computing), η οποία επιτρέπει την εκπαίδευση μοντέλων σε μεγάλα datasets με αυξημένες υπολογιστικές απαιτήσεις. Για τον λόγο αυτό, το framework χρησιμοποιείται ευρέως τόσο στην ακαδημαϊκή έρευνα όσο και σε βιομηχανικά περιβάλλοντα όπου απαιτείται διαχείριση μεγάλου όγκου δεδομένων.

Στην παρούσα εργασία, το H2O AutoML ενσωματώθηκε στο AutoML App με στόχο την αξιοποίηση των ensemble τεχνικών και της υψηλής ακρίβειας που προσφέρει σε σύνθετα προβλήματα ταξινόμησης και παλινδρόμησης.

2.3 FLAML

Το FLAML (Fast Lightweight AutoML) είναι ένα framework Αυτοματοποιημένης Μηχανικής Μάθησης που αναπτύχθηκε από τη Microsoft Research με στόχο τη γρήγορη και αποδοτική

δημιουργία μοντέλων μηχανικής μάθησης με χαμηλό υπολογιστικό κόστος [10, 11].

Σε αντίθεση με άλλα AutoML frameworks που δίνουν έμφαση στην εξαντλητική αναζήτηση μοντέλων, το FLAML επικεντρώνεται στη βελτιστοποίηση της σχέσης απόδοσης και χρόνου εκπαίδευσης χρησιμοποιώντας τεχνικές όπως το Cost-Frugal Optimization (CFO) και το BlendSearch [10]. Οι τεχνικές αυτές επιτρέπουν την αποτελεσματική εξερεύνηση του χώρου αναζήτησης υπερπαραμέτρων με σημαντικά μικρότερη κατανάλωση υπολογιστικών πόρων.

Ένα ακόμη σημαντικό πλεονέκτημα του FLAML είναι η δυνατότητα λειτουργίας υπό αυστηρούς χρονικούς περιορισμούς. Ο χρήστης μπορεί να ορίσει διαθέσιμο χρονικό προϋπολογισμό (time budget), εντός του οποίου το framework αναζητά τη βέλτιστη δυνατή λύση. Η προσέγγιση αυτή το καθιστά ιδιαίτερα κατάλληλο για εφαρμογές πραγματικού χρόνου και διαδικτυακά περιβάλλοντα.

Στο πλαίσιο της παρούσας εργασίας, το FLAML επιλέχθηκε λόγω της υψηλής ταχύτητας εκπαίδευσης και της αποτελεσματικής λειτουργίας του σε περιορισμένα χρονικά budgets, καθιστώντας το ιδιαίτερα κατάλληλο για web-based AutoML εφαρμογές.

2.4 Auto-sklearn

Το Auto-sklearn αποτελεί μία από τις σημαντικότερες πλατφόρμες AutoML στον χώρο της μηχανικής μάθησης και βασίζεται στη βιβλιοθήκη scikit-learn [7, 25]. Το framework αναπτύχθηκε με στόχο την πλήρη αυτοματοποίηση της επιλογής αλγορίθμων και της βελτιστοποίησης υπερπαραμέτρων μέσω προηγμένων τεχνικών Bayesian Optimization.

Η βασική καινοτομία του Auto-sklearn είναι ο συνδυασμός Bayesian Optimization με τεχνικές meta-learning. Μέσω του meta-learning, το σύστημα αξιοποιεί γνώσεις από προηγούμενα σύνολα δεδομένων ώστε να εντοπίσει ταχύτερα υποσχόμενες περιοχές του χώρου αναζήτησης για ένα νέο πρόβλημα [7]. Η προσέγγιση αυτή μειώνει τον χρόνο αναζήτησης και επιτρέπει την αποτελεσματικότερη εύρεση κατάλληλων συνδυασμών αλγορίθμων και υπερπαραμέτρων.

Παράλληλα, το Auto-sklearn δημιουργεί αυτόματα ensemble μοντέλα συνδυάζοντας τα καλύτερα αποτελέσματα που προκύπτουν κατά τη διάρκεια της διαδικασίας βελτιστοποίησης. Η χρήση ensemble learning συμβάλλει στη βελτίωση της γενίκευσης των μοντέλων και συχνά οδηγεί σε υψηλότερη προγνωστική απόδοση συγκριτικά με τη χρήση ενός μεμονωμένου αλγορίθμου [7].

Ένα ακόμη πλεονέκτημα του framework είναι η στενή ενσωμάτωσή του με το οικοσύστημα της scikit-learn, γεγονός που διευκολύνει την αξιοποίησή του από ερευνητές και επαγγελματίες που ήδη χρησιμοποιούν τις αντίστοιχες βιβλιοθήκες [25]. Ωστόσο, η εγκατάσταση και η συντήρησή του ενδέχεται να παρουσιάζουν δυσκολίες λόγω εξαρτήσεων λογισμικού και απαιτήσεων συμβατότητας μεταξύ βιβλιοθηκών.

Παρότι το Auto-sklearn δεν ενσωματώθηκε στην τρέχουσα έκδοση της εφαρμογής AutoML App, παρουσιάζεται στην παρούσα εργασία λόγω της ιδιαίτερης σημασίας του στον χώρο του AutoML και της ευρείας χρήσης του ως benchmark framework στη σχετική βιβλιογραφία.

2.5 Auto-WEKA

Το Auto-WEKA [12] αποτελεί ένα από τα πρωτοποριακά frameworks στον χώρο της Αυτοματοποιημένης Μηχανικής Μάθησης, το οποίο έθεσε τις βάσεις για την αντιμετώπιση του προβλήματος CASH (Combined Algorithm Selection and Hyperparameter Optimization) μέσω τεχνικών Bayesian optimization.

Το πρόβλημα CASH αφορά την ταυτόχρονη επιλογή του καταλληλότερου αλγορίθμου μηχανικής μάθησης και των βέλτιστων υπερπαραμέτρων του. Η διαδικασία αυτή είναι ιδιαίτερα απαιτητική, καθώς ο χώρος αναζήτησης περιλαμβάνει μεγάλο αριθμό πιθανών αλγορίθμων και συνδυασμών παραμέτρων. Το Auto-WEKA αποτέλεσε μία από τις πρώτες πλατφόρμες που αντιμετώπισαν αποτελεσματικά το συγκεκριμένο πρόβλημα μέσω αυτοματοποιημένων τεχνικών αναζήτησης [12].

Η πλατφόρμα βασίζεται στο οικοσύστημα του WEKA και αξιοποιεί μεθόδους Sequential Model-Based Optimization για τη βελτιστοποίηση της διαδικασίας επιλογής μοντέλων. Μέσω αυτής της προσέγγισης είναι δυνατό να εξερευνηθεί αποδοτικά ένας μεγάλος χώρος υποψηφίων λύσεων χωρίς να απαιτείται εξαντλητική αναζήτηση.

Η συμβολή του Auto-WEKA θεωρείται ιδιαίτερα σημαντική για την εξέλιξη του πεδίου του AutoML, καθώς απέδειξε στην πράξη ότι η αυτοματοποίηση της επιλογής αλγορίθμων και υπερπαραμέτρων μπορεί να οδηγήσει σε ανταγωνιστικά αποτελέσματα χωρίς εκτεταμένη ανθρώπινη παρέμβαση. Πολλά μεταγενέστερα frameworks, όπως το Auto-sklearn, βασίστηκαν σε αντίστοιχες αρχές και επεκτάσεις των ιδεών που εισήγαγε το Auto-WEKA.

Αν και το Auto-WEKA διαδραμάτισε καθοριστικό ρόλο στην ανάπτυξη της έρευνας για το AutoML, δεν χρησιμοποιείται στην τρέχουσα υλοποίηση του AutoML App. Η επιλογή αυτή οφείλεται στην ισχυρή εξάρτησή του από το οικοσύστημα της Java και το περιβάλλον του WEKA, ενώ η προτεινόμενη πλατφόρμα επικεντρώνεται σε σύγχρονα AutoML frameworks που βασίζονται στην Python, επιτρέποντας ευκολότερη ενσωμάτωση σε web-based υποδομές και αξιοποίηση των πιο πρόσφατων βιβλιοθηκών μηχανικής μάθησης.

2.6 Συγκριτική Αξιολόγηση Πλατφορμών AutoML

Οι πλατφόρμες AutoML που παρουσιάστηκαν διαφοροποιούνται σημαντικά ως προς τη στρατηγική βελτιστοποίησης, τις υπολογιστικές απαιτήσεις και τον βαθμό αυτοματοποίησης που προσφέρουν. Το MLJAR δίνει έμφαση στην ευκολία χρήσης και στην ερμηνευσιμότητα των αποτελεσμάτων, ενώ το H2O AutoML επικεντρώνεται στην επίτευξη υψηλής ακρίβειας μέσω ensemble τεχνικών και κατανεμημένης επεξεργασίας.
Από την άλλη πλευρά, το FLAML στοχεύει στη μείωση του χρόνου εκπαίδευσης και του υπολογιστικού κόστους, καθιστώντας το κατάλληλο για περιβάλλοντα με περιορισμένους πόρους. Το Auto-sklearn αξιοποιεί προηγμένες τεχνικές meta-learning και Bayesian optimization, ενώ το Auto-WEKA αποτελεί μία ιστορικά σημαντική πλατφόρμα που συνέβαλε καθοριστικά στη θεμελίωση του πεδίου του AutoML.
Η επιλογή του κατάλληλου framework εξαρτάται από τις απαιτήσεις της εκάστοτε εφαρμογής. Στην παρούσα εργασία επιλέχθηκαν τα MLJAR, H2O AutoML και FLAML, καθώς προσφέρουν ισορροπία μεταξύ ακρίβειας, ταχύτητας εκπαίδευσης και ευκολίας ενσωμάτωσης σε

διαδικτυακές εφαρμογές βασισμένες στην Python.

Πίνακας 2.1: Σύγκριση των κυριότερων AutoML frameworks

Framework	Ensemble	Meta-Learning	Distributed	Python
MLJAR	Ναι	Όχι	Όχι	Ναι
H2O AutoML	Ναι	Όχι	Ναι	Ναι
FLAML	Όχι	Όχι	Όχι	Ναι
Auto-sklearn	Ναι	Ναι	Όχι	Ναι
Auto-WEKA	Όχι	Όχι	Όχι	Όχι

Κεφάλαιο 3

Εφαρμογές Ιστού Αυτοματοποιημένης Μηχανικής Μάθησης

Η συνεχής ανάπτυξη της Μηχανικής Μάθησης και των τεχνολογιών AutoML έχει οδηγήσει στην εμφάνιση πληθώρας διαδικτυακών πλατφορμών που επιτρέπουν την ανάπτυξη και αξιοποίηση μοντέλων μέσω φιλικών γραφικών διεπαφών. Οι πλατφόρμες αυτές στοχεύουν στη μείωση της πολυπλοκότητας της διαδικασίας μηχανικής μάθησης, καθιστώντας την προσβάσιμη και σε χρήστες χωρίς εξειδικευμένες προγραμματιστικές γνώσεις.

Οι σύγχρονες AutoML web εφαρμογές υποστηρίζουν λειτουργίες όπως αυτόματη επιλογή μοντέλων, βελτιστοποίηση υπερπαραμέτρων, αξιολόγηση επιδόσεων και ανάπτυξη μοντέλων σε cloud υποδομές. Παράλληλα, παρέχουν δυνατότητες ενσωμάτωσης μέσω REST APIs, διευκολύνοντας την αξιοποίησή τους σε πραγματικά πληροφοριακά συστήματα.

Οι διαδικτυακές εφαρμογές AutoML μπορούν να διακριθούν σε δύο βασικές κατηγορίες. Η πρώτη περιλαμβάνει εμπορικές πλατφόρμες που παρέχονται ως υπηρεσίες υπολογιστικού νέφους (cloud services) από μεγάλους οργανισμούς τεχνολογίας, ενώ η δεύτερη αφορά εξειδικευμένες εφαρμογές που αναπτύσσονται για συγκεκριμένους αλγορίθμους ή κατηγορίες προβλημάτων μηχανικής μάθησης. Η διάκριση αυτή είναι ιδιαίτερα σημαντική, καθώς οι δύο κατηγορίες εξυπηρετούν διαφορετικές ανάγκες χρηστών και παρουσιάζουν διαφορετικές απαιτήσεις ως προς την αρχιτεκτονική και την υποδομή τους.

3.1 Εμπορικές Πλατφόρμες AutoML

Οι μεγάλες cloud πλατφόρμες αποτελούν χαρακτηριστικά παραδείγματα αυτοματοποιημένης μηχανικής μάθησης σε παραγωγικά περιβάλλοντα.
Η πλατφόρμα Google Vertex AI παρέχει υπηρεσίες AutoML για προβλήματα ταξινόμησης, παλινδρόμησης και υπολογιστικής όρασης, με υποστήριξη αυτοματοποιημένης εκπαίδευσης και ανάπτυξης μοντέλων στο cloud [26].
Αντίστοιχα, το Azure Automated ML της Microsoft επιτρέπει την αυτόματη δημιουργία και αξιολόγηση μοντέλων, υποστηρίζοντας επιλογή αλγορίθμων και βελτιστοποίηση υπερπαραμέτρων μέσα από το οικοσύστημα Azure Machine Learning [27].
Το Amazon SageMaker Autopilot της AWS αυτοματοποιεί τη διαδικασία δημιουργίας μοντέλων, συμπεριλαμβανομένης της προεπεξεργασίας δεδομένων, της εκπαίδευσης και της επιλο-

γής βέλτιστου μοντέλου [28].
Οι παραπάνω πλατφόρμες αποτελούν αντιπροσωπευτικά παραδείγματα βιομηχανικών λύσεων AutoML με έμφαση στην επεκτασιμότητα και την ενσωμάτωση σε cloud περιβάλλοντα.

3.2 AutoKNN

Το AutoKNN αποτελεί διαδικτυακή εφαρμογή AutoML που αυτοματοποιεί τον αλγόριθμο k-Nearest Neighbors για προβλήματα ταξινόμησης [29]. Οι χρήστες μπορούν να φορτώνουν δεδομένα, να ορίζουν χαρακτηριστικά και να εκπαιδεύουν μοντέλα μέσω web διεπαφής ή REST API.
Η εφαρμογή υποστηρίζει λειτουργία "auto mode", η οποία επιλέγει αυτόματα τη βέλτιστη τιμή του k και τη μετρική απόστασης, μειώνοντας την ανάγκη χειροκίνητου tuning. Παράλληλα παρέχει βασικά metrics αξιολόγησης (Precision, Recall, F1-score) και δυνατότητα αποθήκευσης μοντέλων για επαναχρησιμοποίηση.
Αρχιτεκτονικά, βασίζεται σε PHP backend, MySQL βάση δεδομένων και Python modules με χρήση scikit-learn για την εκπαίδευση μοντέλων.

3.3 AutoDTrees

Το AutoDTrees είναι web εφαρμογή AutoML για την αυτοματοποίηση του αλγορίθμου Decision Trees [30]. Υποστηρίζει φόρτωση δεδομένων CSV, επιλογή χαρακτηριστικών και εκπαίδευση μοντέλων μέσω γραφικής διεπαφής ή API.
Ο χρήστης μπορεί να ρυθμίσει βασικές παραμέτρους όπως max depth και min samples leaf, ενώ η αξιολόγηση πραγματοποιείται με k-fold cross-validation. Επιπλέον, παρέχεται δυνατότητα οπτικοποίησης του δέντρου για καλύτερη ερμηνευσιμότητα.
Η υλοποίηση βασίζεται σε PHP backend, MySQL και Python/scikit-learn modules.

3.4 kClusterHub

Το kClusterHub είναι πλατφόρμα AutoML για clustering δεδομένων με χρήση partition-based αλγορίθμων [31]. Υποστηρίζει k-means, k-modes και k-prototypes, επιτρέποντας ανάλυση αριθμητικών, κατηγορικών και μεικτών δεδομένων.
Μέσω λειτουργίας auto mode, η εφαρμογή επιλέγει αυτόματα κατάλληλο αλγόριθμο και προτείνει αριθμό clusters με τη μέθοδο elbow. Παρέχεται επίσης οπτικοποίηση και εξαγωγή αποτελεσμάτων.
Η αρχιτεκτονική περιλαμβάνει PHP backend, MySQL και Python modules με scikit-learn και kmodes.

3.5 WebApriori

Το WebApriori αποτελεί web εφαρμογή εξόρυξης κανόνων συσχέτισης βασισμένη στον αλγόριθμο Apriori [32]. Επιτρέπει την ανάλυση συναλλακτικών δεδομένων μέσω παραμετροποίη-

σης support, confidence και lift.
Το σύστημα εξάγει frequent itemsets και κανόνες συσχέτισης, οι οποίοι αξιολογούνται μέσω πολλαπλών μετρικών. Υποστηρίζεται επίσης απομάκρυνση redundant κανόνων και εξαγωγή αποτελεσμάτων σε JSON για εξωτερική χρήση.
Αρχιτεκτονικά αποτελείται από Python Apriori engine, PHP backend και MySQL βάση δεδομένων, με REST API για εξωτερική πρόσβαση.

3.6 Συζήτηση και Συσχέτιση με την Παρούσα Εργασία

Οι εφαρμογές που παρουσιάστηκαν αποδεικνύουν ότι η αυτοματοποίηση διαδικασιών μηχανικής μάθησης μέσω διαδικτυακών πλατφορμών αποτελεί μια ώριμη και ιδιαίτερα χρήσιμη προσέγγιση. Παρά τις διαφορές τους ως προς το πεδίο εφαρμογής, όλες επιδιώκουν τη μείωση της πολυπλοκότητας που συνοδεύει την ανάπτυξη μοντέλων μηχανικής μάθησης.
Οι εμπορικές πλατφόρμες παρέχουν ολοκληρωμένα περιβάλλοντα AutoML μεγάλης κλίμακας, απαιτώντας ωστόσο σημαντικούς υπολογιστικούς πόρους και συνήθως συνδρομητικές υπηρεσίες. Αντίθετα, εφαρμογές όπως οι AutoKNN, AutoDTrees, kClusterHub και WebApriori εστιάζουν σε συγκεκριμένες τεχνικές εξόρυξης γνώσης και μηχανικής μάθησης, προσφέροντας απλούστερη αρχιτεκτονική και μεγαλύτερη ευελιξία ως προς την ανάπτυξη εξειδικευμένων λύσεων.
Η παρούσα εργασία ακολουθεί παρόμοια φιλοσοφία, συνδυάζοντας τη λειτουργικότητα διαδικτυακής εφαρμογής με τη χρήση σύγχρονων AutoML frameworks, όπως τα MLJAR, H2O AutoML και FLAML. Με τον τρόπο αυτό επιδιώκεται η παροχή ενός ενιαίου περιβάλλοντος εκπαίδευσης, αξιολόγησης και αξιοποίησης μοντέλων μηχανικής μάθησης, χωρίς να απαιτείται εκτεταμένη εμπειρία προγραμματισμού από τον τελικό χρήστη.

Κεφάλαιο 4

Γλώσσες και Τεχνολογίες Ανάπτυξης

Στο κεφάλαιο αυτό παρουσιάζονται αναλυτικά οι γλώσσες προγραμματισμού, τα πλαίσια εργασίας (frameworks), οι βιβλιοθήκες και τα εργαλεία λογισμικού που επιλέχθηκαν για την ανάπτυξη της εφαρμογής AutoML. Η επιλογή των συγκεκριμένων τεχνολογιών βασίστηκε σε κριτήρια σταθερότητας, απόδοσης, ευκολίας ενσωμάτωσης και δυνατότητας διαχείρισης απαιτητικών υπολογιστικών διεργασιών, όπως αυτές που προκύπτουν στον τομέα της αυτοματοποιημένης μηχανικής μάθησης [8, 6].

4.1 Τεχνολογίες Διεπαφής Χρήστη (Frontend)

Για τη δημιουργία του περιβάλλοντος αλληλεπίδρασης με τον χρήστη χρησιμοποιήθηκαν σύγχρονες τεχνολογίες ιστού, με στόχο την ανάπτυξη μιας δυναμικής, αποκρινόμενης (responsive) και φιλικής προς τον χρήστη διεπαφής.

- **HTML5 (HyperText Markup Language):** Αποτελεί τη βασική γλώσσα σήμανσης για τη δομή και την οργάνωση του περιεχομένου των ιστοσελίδων της εφαρμογής, διασφαλίζοντας τη σωστή ιεραρχία των στοιχείων (φόρμες, πίνακες, κουμπιά).
- **CSS3 (Cascading Style Sheets):** Χρησιμοποιήθηκε για τη μορφοποίηση, τη σχεδίαση και την οπτική παρουσίαση των στοιχείων της διεπαφής, εξασφαλίζοντας μια ομοιόμορφη και σύγχρονη αισθητική σε όλη την έκταση της εφαρμογής.
- **Bootstrap Framework:** Πρόκειται για ένα από τα πιο δημοφιλή open-source CSS frameworks για την ανάπτυξη responsive ιστοσελίδων. Χρησιμοποιήθηκε για τη γρήγορη και αποτελεσματική σχεδίαση της διεπαφής, καθώς προσφέρει έτος και βελτιστοποιημένα συστατικά (όπως πλέγματα διάταξης, φόρμες εισαγωγής, κουμπιά και αναδυόμενα παράθυρα) που προσαρμόζονται αυτόματα σε οποιοδήποτε μέγεθος οθόνης.
- **JavaScript (Vanilla JS):** Χρησιμοποιήθηκε για την προσθήκη δυναμικότητας και διαδραστικότητας στη διεπαφή. Μέσω της JavaScript υλοποιείται ο έλεγχος εγκυρότητας των δεδομένων εισόδου, η διαχείριση των γεγονότων (event handling), η δυναμική τροποποίηση του περιεχομένου της σελίδας ανάλογα με τις επιλογές του χρήστη, καθώς και η εκτέλεση ασύγχρονων αιτημάτων (AJAX/Fetch) προς το backend για την επικοινωνία χωρίς την ανάγκη ανανέωσης της σελίδας.

4.2 Τεχνολογίες Backend και Διαχείρισης Δεδομένων

Το επίπεδο του server και της διαχείρισης των δεδομένων υλοποιήθηκε με τεχνολογίες που εξασφαλίζουν ασφάλεια, ταχύτητα στην επεξεργασία των αιτημάτων και σταθερή διατήρηση της πληροφορίας.

- **PHP (Hypertext Preprocessor):** Μια ευρέως διαδεδομένη γλώσσα προγραμματισμού script στην πλευρά του διακομιστή (server-side). Χρησιμοποιήθηκε για την ανάπτυξη της επιχειρησιακής λογικής (business logic) του Backend API, τη διαχείριση των HTTP αιτημάτων, την ασφαλή αυθεντικοποίηση και διαχείριση των συνεδριών των χρηστών (Session-based Authentication), καθώς και για τον συντονισμό των απαραίτητων αλληλεπιδράσεων με τη βάση δεδομένων και το σύστημα αρχείων.
- **SQL (Structured Query Language):** Χρησιμοποιήθηκε για τη σχεδίαση και τη διαχείριση της σχεσιακής βάσης δεδομένων του συστήματος. Μέσω της SQL πραγματοποιείται η αποθήκευση, η οργάνωση και η ανάκτηση των δομημένων δεδομένων της εφαρμογής, όπως είναι τα στοιχεία των χρηστών, τα μεταδεδομένα των αρχείων, οι συσχετίσεις των εκπαιδευμένων μοντέλων και το ιστορικό των προβλέψεων.

4.3 Γλώσσα Προγραμματισμού Python και Πλαίσια AutoML

Η γλώσσα προγραμματισμού **Python** επιλέχθηκε ως η αποκλειστική τεχνολογία για το επίπεδο εκτέλεσης των διεργασιών επιστήμης δεδομένων, καθώς διαθέτει το πιο ώριμο και ισχυρό οικοσύστημα βιβλιοθηκών για Μηχανική Μάθηση, με βασικό πυλώνα τη βιβλιοθήκη Scikit-learn [25]. Στο πλαίσιο της εφαρμογής, χρησιμοποιήθηκαν τα ακόλουθα εξειδικευμένα πλαίσια αυτοματοποιημένης μηχανικής μάθησης (AutoML Frameworks):

- **FLAML (Fast and Lightweight AutoML):** Μια βιβλιοθήκη ανοικτού κώδικα της Microsoft, η οποία έχει σχεδιαστεί για την εύρεση ποιοτικών μοντέλων μηχανικής μάθησης με εξαιρετικά χαμηλό υπολογιστικό κόστος [10]. Χρησιμοποιεί αποδοτικούς αλγορίθμους αναζήτησης υπερπαραμέτρων και βελτιστοποίησης, καθιστώντας την ιδανική για περιορισμένους χρονικούς προϋπολογισμούς (time budgets) [11].
- **MLJAR AutoML:** Ένα προηγμένο πλαίσιο εργασίας που εστιάζει στην παραγωγή μοντέλων υψηλής ακρίβειας για tabular δεδομένα [16]. Διακρίνεται για τις δυνατότητες αυτόματης μηχανικής χαρακτηριστικών (feature engineering), τη δημιουργία συνόλων μοντέλων (ensembling) και την παραγωγή λεπτομερών αναφορών επεξηγηματικής τεχνητής νοημοσύνης (XAI) βασισμένων σε τιμές SHAP [33].
- **H2O AutoML:** Μια ισχυρή, κλιμακώσιμη και ευρέως διαδεδομένη πλατφόρμα μηχανικής μάθησης [15]. Είναι βελτιστοποιημένη για παράλληλη επεξεργασία και διαχείριση μεγάλων συνόλων δεδομένων, ενώ αυτοματοποιεί πλήρως τη διαδικασία εκπαίδευσης και αξιολόγησης μιας ευρείας γκάμας αλγορίθμων (όπως Gradient Boosting Machines, Deep Learning, και Stacked Ensembles).

4.4 Εργαλεία Φιλοξενίας και Ελέγχου Εκδόσεων

- **Apache Web Server:** Χρησιμοποιήθηκε ως ο διακομιστής ιστού για τη φιλοξενία και την εκτέλεση της PHP εφαρμογής, προσφέροντας ένα ασφαλές και σταθερό περιβάλλον διαχείρισης της κυκλοφορίας δικτύου.
- **Git & GitHub:** Το Git χρησιμοποιήθηκε ως το σύστημα ελέγχου εκδόσεων (version control) για την παρακολούθηση των αλλαγών στον κώδικα κατά τη διάρκεια της ανάπτυξης, ενώ η πλατφόρμα GitHub χρησιμοποιήθηκε για την αποθήκευση του αποθετηρίου (repository) και την ανοικτή διάθεση του πηγαίου κώδικα στην κοινότητα.

Κεφάλαιο 5

Πειραματική Σύγκριση των Πλατφόρμων Αυτοματοποιημένης Μηχανικής Μάθησης

Πριν καταλήξουμε στην τελική επιλογή και την ενσωμάτωση των βιβλιοθηκών στην εφαρμογή `AutoML App`, κρίθηκε αναγκαίο να πραγματοποιηθεί μια εκτενής πειραματική αξιολόγηση και σύγκριση των AutoML frameworks που υποστηρίζει το σύστημα. Συγκεκριμένα, εξετάστηκαν το **H2O AutoML**, το **FLAML** και το **MLJAR**. Σκοπός των πειραμάτων είναι να μελετηθεί και να συγκριθεί η συμπεριφορά αυτών των τριών συστημάτων όταν λειτουργούν κάτω από τις ίδιες ακριβώς συνθήκες, χρησιμοποιώντας τα ίδια σύνολα δεδομένων, τον ίδιο διαχωρισμό εκπαίδευσης/ελέγχου (train/test split) και τους ίδιους χρονικούς περιορισμούς (time budgets).

5.1 Σύνολα Δεδομένων (Datasets)

Για τη διεξαγωγή των πειραμάτων επιλέχθηκαν τρία δημόσια και ευρέως αναγνωρισμένα σύνολα δεδομένων (benchmark datasets). Η επιλογή έγινε με γνώμονα την πολυμορφία, ώστε να δοκιμαστούν τα frameworks σε διαφορετικές συνθήκες όσον αφορά το μέγεθος των δειγμάτων, το πλήθος των χαρακτηριστικών και το είδος του προβλήματος:

- **Iris Dataset:** Πρόκειται για ένα κλασικό και μικρό σύνολο δεδομένων, το οποίο αφορά προβλήματα πολυκατηγορικής ταξινόμησης (multi-class classification).
- **Letter Recognition Dataset:** Ένα σύνολο δεδομένων μεσαίας κλίμακας με αρκετά περισσότερα χαρακτηριστικά και αυξημένη πολυπλοκότητα, κατάλληλο για να ελεγχθεί η συμπεριφορά των αλγορίθμων σε πιο δύσκολα προβλήματα ταξινόμησης.
- **Boston Housing Dataset:** Ένα σύνολο δεδομένων που αφορά πρόβλημα παλινδρόμησης (regression), όπου ο στόχος είναι η πρόβλεψη συνεχών τιμών (τιμές κατοικιών) με βάση πολλαπλά αριθμητικά χαρακτηριστικά.

Τα βασικά χαρακτηριστικά των συνόλων δεδομένων που χρησιμοποιήθηκαν συνοψίζονται συγκεντρωτικά στον Πίνακα 5.1.

Πίνακας 5.1: Χαρακτηριστικά των συνόλων δεδομένων της πειραματικής αξιολόγησης.

Σύνολο Δεδομένων (Dataset)	**Δείγματα (Instances)**	**Χαρακτηριστικά (Features)**	**Κλάσεις (Classes)**	**Τύπος Προβλήματος**
Iris	150	4	3	Ταξινόμηση
Letter Recognition	20.000	16	26	Ταξινόμηση
Boston Housing	506	13	–	Παλινδρόμηση

5.2 Τρόπος Διεξαγωγής των Πειραμάτων (Experimental Setup)

Για να είναι η σύγκριση μεταξύ των πλατφορμών απόλυτα δίκαιη και σωστή, ακολουθήθηκε μια σταθερή διαδικασία για όλα τα frameworks:

1. **Διαχωρισμός Δεδομένων:** Κάθε dataset χωρίστηκε σε υποσύνολο εκπαίδευσης (training set) και υποσύνολο ελέγχου (testing set) με σταθερή αναλογία 80/20. Επίσης, χρησιμοποιήθηκε ένας σταθερός τυχαίος σπόρος (fixed random seed) ώστε τα αποτελέσματα να είναι επαναλήψιμα.

2. **Όχι Χειροκίνητες Παρεμβάσεις:** Δεν έγινε καμία απολύτως χειροκίνητη επεξεργασία ή επιλογή χαρακτηριστικών (feature engineering) στα δεδομένα. Αφήσαμε το κάθε framework να διαχειριστεί αυτόνομα τα δεδομένα στην αρχική τους μορφή.

3. **Χρονικοί Περιορισμοί (Time Budget):** Όλα τα frameworks έτρεξαν με τους ίδιους ακριβώς χρονικούς περιορισμούς ανά dataset, οι οποίοι ορίστηκαν στα 60, 180 και 120 δευτερόλεπτα αντίστοιχα. Ο χρόνος που καταγράφεται αφορά τον πραγματικό χρόνο που χρειάστηκε η κάθε πλατφόρμα για να εκπαιδεύσει τα μοντέλα της (runtime).

5.3 Μετρικές Αξιολόγησης και Αποδοτικότητα

Για τα προβλήματα ταξινόμησης υπολογίστηκαν οι γνωστές μετρικές Accuracy, Precision, Recall και F1-score [3]. Για λόγους ομοιομορφίας και για να είναι πιο εύκολη η σύγκριση, στον τελικό πίνακα παρουσιάζεται η μετρική **Accuracy**.
Αντίστοιχα, για το πρόβλημα της παλινδρόμησης υπολογίστηκαν οι μετρικές RMSE, MSE, MAE καθώς και ο συντελεστής προσδιορισμού R^2 [1, 2]. Ως κύρια μετρική σύγκρισης επιλέχθηκε ο R^2, ο οποίος δείχνει πόσο καλά προσαρμόζεται το μοντέλο στα δεδομένα.
Επειδή όμως η υψηλή επίδοση συχνά απαιτεί πολύ χρόνο, κρίθηκε σκόπιμο να εισάγουμε και μια δική μας μετρική, την υπολογιστική αποδοτικότητα (**Efficiency**). Αυτή ορίζεται μαθηματικά ως το πηλίκο της επίδοσης του μοντέλου προς τον χρόνο εκπαίδευσης σε δευτερόλεπτα:

$$\text{Efficiency} = \frac{\text{Μετρική Επίδοσης (Score)}}{\text{Χρόνος Εκπαίδευσης (Time (s))}}$$

Η μετρική αυτή μας βοηθάει να καταλάβουμε το trade-off (συμβιβασμό) ανάμεσα στο πόσο καλό είναι ένα μοντέλο και στο πόσο γρήγορα καταφέρνει να εκπαιδευτεί.

5.4 Αποτελέσματα και Ανάλυση

Τα συνολικά αποτελέσματα από τις μετρήσεις των πειραμάτων παρουσιάζονται αναλυτικά στον Πίνακα 5.2.

Πίνακας 5.2: Συνολικά αποτελέσματα πειραματικής σύγκρισης υπό σταθερούς χρονικούς περιορισμούς.

Dataset	Framework	Algorithm	Score	Time(s)	Eff.
Iris (Acc.) *Budget:60s*	H2O	GLM	**1.00**	61.55	0.0162
	FLAML	Extra Trees	0.93	68.74	0.0135
	MLJAR	Ensemble	0.96	17.97	**0.0534**
Letter (Acc.) *Budget:180s*	H2O	Stacked Ens.	0.92	186.66	0.0049
	FLAML	CatBoost	0.94	201.84	0.0047
	MLJAR	Ensemble	**0.95**	87.32	**0.0109**
Boston (R^2) *Budget:120s*	H2O	Stacked Ens.	0.869	121.64	0.0072
	FLAML	Extra Trees	0.866	120.01	0.0072
	MLJAR	Ensemble	**0.88**	20.43	**0.0431**

5.5 Συζήτηση και Συμπεράσματα Πειράματων

Κοιτάζοντας τα αποτελέσματα, γίνεται αμέσως σαφές ότι κανένα framework δεν κερδίζει σε όλες τις δοκιμές. Αυτό είναι λογικό, καθώς η κάθε πλατφόρμα χρησιμοποιεί διαφορετικές εσωτερικές στρατηγικές αναζήτησης και αλγορίθμους:

- **H2O AutoML:** Δείχνει να έχει πολύ καλή προγνωστική ισχύ, ειδικά στα πιο δύσκολα προβλήματα. Αυτό συμβαίνει γιατί ψάχνει πολύ τον χώρο των λύσεων και δημιουργεί σύνθετα μοντέλα (Stacked Ensembles). Το μειονέκτημά του είναι ότι είναι αρκετά βαρύ και «εξαντλεί» σχεδόν πάντα όλο τον χρόνο που του δίνουμε.
- **FLAML:** Αυτή η πλατφόρμα εστιάζει καθαρά στο να είναι ελαφριά και γρήγορη. Επιλέγει κυρίως αποδοτικούς αλγορίθμους βασισμένους σε δέντρα (όπως Extra Trees και CatBoost) και καταφέρνει να βγάλει πολύ καλά σκορ χωρίς να καταναλώνει πολλούς πόρους.
- **MLJAR:** Πέτυχε τον καλύτερο συνδυασμό αποτελέσματος και χρόνου. Σε όλα τα πειράματα είχε την **καλύτερη υπολογιστική αποδοτικότητα (Efficiency)**, αφού κατάφερε να φτάσει σε κορυφαία αποτελέσματα (μέσω Ensembles) σε πολύ λιγότερο χρόνο από τα άλλα δύο frameworks (ενδεικτικά χρειάστηκε μόλις 17.97s στο Iris και 20.43s στο Boston Housing).

Αυτή ακριβώς η διαπίστωση —ότι δηλαδή το κάθε framework έχει τα δικά του δυνατά σημεία— αποτέλεσε και το βασικό κίνητρο για τον τρόπο που σχεδιάστηκε η εφαρμογή μας, το `AutoML`

`App`. Αντί να περιορίσουμε τον χρήστη σε μία μόνο επιλογή, του δίνουμε τη δυνατότητα να τρέξει τις πλατφόρμες παράλληλα. Με αυτόν τον τρόπο, εκμεταλλευόμαστε τα πλεονεκτήματα της κάθε μίας, ώστε τελικά να επιλέγεται αυτόματα το καλύτερο δυνατό μοντέλο για το πρόβλημα που μελετάται.

Κεφάλαιο 6

Σχεδίαση και Υλοποίηση του AutoML App

Στο παρόν κεφάλαιο παρουσιάζεται αναλυτικά η σχεδίαση και η υλοποίηση της διαδικτυακής εφαρμογής AutoML App, η οποία αναπτύχθηκε στο πλαίσιο της παρούσας διπλωματικής εργασίας. Αρχικά γίνεται ανάλυση των λειτουργικών και μη λειτουργικών απαιτήσεων του συστήματος, καθώς και των βασικών user stories που καθόρισαν τη σχεδίαση της εφαρμογής. Στη συνέχεια παρουσιάζεται η αρχιτεκτονική του συστήματος, η σχεδίαση της σχεσιακής βάσης δεδομένων, καθώς και η υλοποίηση του backend και του frontend μέρους της εφαρμογής.

6.1 Ανάλυση Απαιτήσεων

Η ανάπτυξη της εφαρμογής AutoML App βασίστηκε σε ένα σύνολο λειτουργικών και μη λειτουργικών απαιτήσεων, οι οποίες προέκυψαν από την ανάγκη δημιουργίας ενός φιλικού και αυτοματοποιημένου περιβάλλοντος μηχανικής μάθησης για χρήστες χωρίς εξειδικευμένες γνώσεις προγραμματισμού.
Κύριος στόχος του συστήματος είναι η παροχή μιας ενιαίας διαδικτυακής πλατφόρμας, μέσω της οποίας οι χρήστες μπορούν να εκτελούν διαδικασίες Αυτοματοποιημένης Μηχανικής Μάθησης (AutoML) χωρίς άμεση αλληλεπίδραση με κώδικα ή εξειδικευμένα εργαλεία.

6.1.1 Λειτουργικές Απαιτήσεις

Οι βασικές λειτουργικές απαιτήσεις του συστήματος περιλαμβάνουν:

- Εγγραφή και αυθεντικοποίηση χρηστών μέσω session-based μηχανισμού.
- Μεταφόρτωση συνόλων δεδομένων σε μορφή CSV.
- Υποστήριξη δημόσιων και ιδιωτικών datasets.
- Επιλογή μεταβλητής στόχου και χαρακτηριστικών εισόδου.
- Εκπαίδευση μοντέλων μέσω πολλαπλών AutoML frameworks.
- Υποστήριξη προβλημάτων classification και regression.
- Αυτόματη αξιολόγηση και σύγκριση μοντέλων.

- Αποθήκευση βέλτιστων εκπαιδευμένων μοντέλων.
- Εκτέλεση προβλέψεων σε νέα δεδομένα.
- Διαχείριση datasets, μοντέλων και αποτελεσμάτων προβλέψεων.
- Επικοινωνία frontend–backend μέσω HTTP αιτημάτων και JSON.

6.1.2 Μη Λειτουργικές Απαιτήσεις

Οι μη λειτουργικές απαιτήσεις του συστήματος περιλαμβάνουν:

- Ευχρηστία και φιλικότητα προς τον τελικό χρήστη.
- Επεκτασιμότητα για ενσωμάτωση νέων AutoML frameworks.
- Αξιοπιστία στην αποθήκευση δεδομένων και μοντέλων.
- Ασφάλεια μέσω session-based authentication.
- Υποστήριξη ταυτόχρονων χρηστών.
- Καθαρός διαχωρισμός frontend και backend.
- Αποδοτική διαχείριση αρχείων και υπολογιστικών διεργασιών.

6.1.3 User Stories

Οι απαιτήσεις αποτυπώθηκαν επίσης μέσω user stories, τα οποία περιγράφουν τις βασικές λειτουργίες από την οπτική του τελικού χρήστη:

- Ως χρήστης, θέλω να ανεβάζω datasets ώστε να τα χρησιμοποιώ για εκπαίδευση μοντέλων.
- Ως χρήστης, θέλω να μπορώ να δημοσιεύω datasets ώστε να είναι διαθέσιμα και σε άλλους χρήστες.
- Ως χρήστης, θέλω να επιλέγω διαφορετικά AutoML frameworks ώστε να συγκρίνω την απόδοσή τους.
- Ως χρήστης, θέλω να αποθηκεύω εκπαιδευμένα μοντέλα για μελλοντική χρήση.
- Ως χρήστης, θέλω να εκτελώ προβλέψεις σε νέα δεδομένα.
- Ως χρήστης, θέλω να διαχειρίζομαι τα datasets, τα μοντέλα και τα αποτελέσματά μου.
- Ως χρήστης, θέλω να βλέπω μετρικές αξιολόγησης των μοντέλων μέσω γραφικού περιβάλλοντος.

6.2 Αρχιτεκτονική Συστήματος

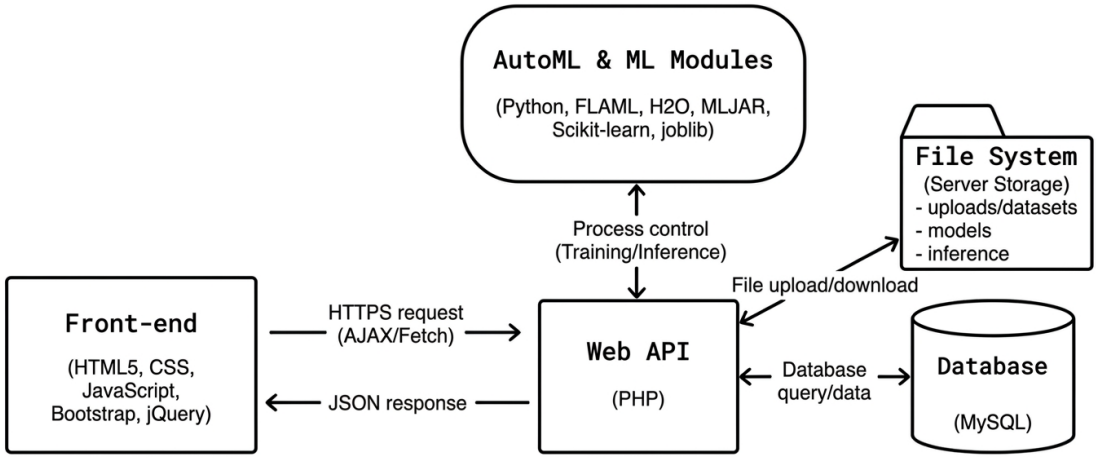
Η εφαρμογή AutoML App ακολουθεί μια πολυεπίπεδη αρχιτεκτονική (multi-layer architecture), η οποία αποτελείται από διακριτά αλλά στενά συνεργαζόμενα υποσυστήματα. Η συγκεκριμένη αρχιτεκτονική επιλέχθηκε με σκοπό να διασφαλίζεται ο σαφής διαχωρισμός ευθυνών (separation of concerns), η οριζόντια επεκτασιμότητα του συστήματος και η ανεξαρτησία των επιμέρους τεχνολογικών συνιστωσών (frontend, backend και περιβάλλοντος εκτέλεσης μηχανικής μάθησης).

6.2.1 Διάγραμμα Αρχιτεκτονικής Τεσσάρων Επιπέδων

Η δομή της πλατφόρμας οργανώνεται σε τέσσερα θεμελιώδη επίπεδα (Layers), καθένα από τα οποία επιτελεί έναν εξειδικευμένο ρόλο:

- **Frontend Layer (Επίπεδο Παρουσίασης):** Υλοποιήθηκε με χρήση προτύπων HTML5, CSS3, JavaScript (jQuery) και του responsive σκελετού Bootstrap 5. Είναι υπεύθυνο για τη δυναμική οπτικοποίηση των δεδομένων, τη διαχείριση των client-side αλληλεπιδράσεων και την ασύγχρονη επικοινωνία.
- **Backend Layer (Επίπεδο Λογικής):** Βασίζεται σε PHP και λειτουργεί ως REST-like Web API. Αναλαμβάνει τον έλεγχο ταυτοποίησης (Authentication), την επικύρωση των δεδομένων (Validation), τη διαχείριση της συνεδρίας (Session management) και τον συντονισμό (Orchestration) των διεργασιών.
- **Python Execution Layer (Επίπεδο Μηχανικής Μάθησης):** Αποτελεί τον υπολογιστικό πυρήνα της εφαρμογής. Λειτουργεί σε απομονωμένο περιβάλλον και εκτελεί ασύγχρονα τις διαδικασίες Αυτοματοποιημένης Μηχανικής Μάθησης (AutoML), ενσωματώνοντας και συγκρίνοντας τα frameworks MLJAR [16], FLAML [10, 11] και H2O.ai [15], καθώς και τις διαδικασίες παραγωγής προβλέψεων (inference). Το script `worker.py` λειτουργεί ως κεντρικός ενορχηστρωτής (orchestrator).
- **Data & Storage Layer (Επίπεδο Δεδομένων και Αποθήκευσης):** Αποτελείται από τη σχεσιακή βάση δεδομένων (MySQL) για τη διατήρηση των δομημένων δεδομένων και το File Storage (Σύστημα Αρχείων του διακομιστή) για τη φυσική αποθήκευση των αρχείων CSV και των σειριοποιημένων (serialized) μοντέλων μηχανικής μάθησης.

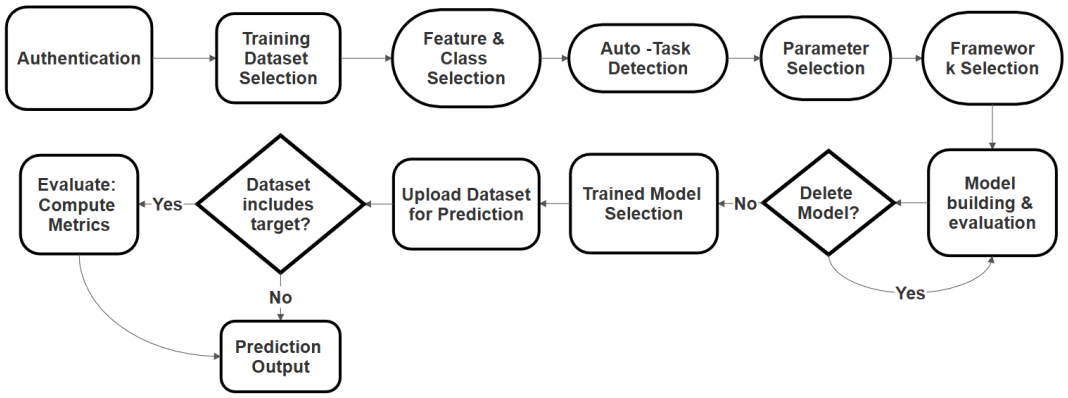
Η επικοινωνία μεταξύ του Frontend και του Backend πραγματοποιείται αποκλειστικά μέσω ασύγχρονων HTTP αιτημάτων (AJAX Requests) με ανταλλαγή δεδομένων σε τυποποιημένη μορφή JSON, εκμηδενίζοντας την ανάγκη για πλήρη ανανέωση της σελίδας.



Σχήμα 6.1: Αρχιτεκτονική τεσσάρων επιπέδων του AutoML App.

6.2.2 Διάγραμμα Ροής Λειτουργίας (Flowchart)

Η δυναμική συμπεριφορά του συστήματος και η αλληλουχία των ενεργειών κατά τη διαδικασία εκπαίδευσης και πρόβλεψης αποτυπώνονται στο διάγραμμα ροής της εφαρμογής. Η ροή εργασίας ξεκινά από την εισαγωγή και την τοπική client-side ανάλυση του συνόλου δεδομένων και ολοκληρώνεται με την ασύγχρονη παραγωγή και παρουσίαση των βέλτιστων μοντέλων.



Σχήμα 6.2: Διάγραμμα ροής δεδομένων και ασύγχρονης εκτέλεσης στο AutoML App.

Η επεξεργασία, όπως φαίνεται στο Σχήμα 6.2, ακολουθεί τα εξής στάδια:

1. Ο χρήστης εισάγει ένα dataset (μεταφόρτωση ή επιλογή από βιβλιοθήκη), το frontend εκτελεί τοπική ανάλυση και ο χρήστης ορίζει τις παραμέτρους (Target, Features, Metric, Frameworks, Time Limit).

2. Με την εκκίνηση, το Backend καταχωρεί μια νέα εργασία (`job`) στη βάση δεδομένων με κατάσταση *Pending* ή *Running* και επιστρέφει άμεσα τον έλεγχο στο Frontend.

3. Το Frontend ενεργοποιεί το overlay αναμονής και ξεκινά περιοδικό έλεγχο (Polling ανά 3 δευτερόλεπτα) μέσω του endpoint `check_job_status.php`.

4. Στο background, το Python layer εκτελεί την εκπαίδευση, εξάγει το βέλτιστο μοντέλο, αποθηκεύει τις μετρικές στη βάση δεδομένων και ενημερώνει την κατάσταση της εργασίας σε *Success*.

5. Μόλις το Polling ανιχνεύσει την ολοκλήρωση, το Frontend ανακτά τα αποτελέσματα σε μορφή JSON, απενεργοποιεί το overlay και εμφανίζει το modal με τη σύνοψη και τη σήμανση του νικηφόρου framework.

6.3 Σχεδίαση Σχεσιακής Βάσης Δεδομένων

Για την ασφαλή, συνεπή και αποδοτική διατήρηση των δεδομένων της πλατφόρμας χρησιμοποιήθηκε το σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων (ΣΣΔΒΔ) MySQL. Η σχεδίαση του σχήματος προσανατολίστηκε στην πλήρη υποστήριξη του κύκλου ζωής των AutoML διεργασιών, διασφαλίζοντας την ακεραιότητα των δεδομένων μέσω αυστηρών περιορισμών (constraints) και ξένων κλειδιών.

6.3.1 Οντότητες και Μοντέλο Οντοτήτων - Συσχετίσεων

Το πληροφοριακό σύστημα δομείται γύρω από έξι βασικές οντότητες, οι οποίες μοντελοποιούν τους χρήστες, τα αρχεία τους, τις background εργασίες και τα παραγόμενα αποτελέσματα:

- `users`: Αποθηκεύει τα κρυπτογραφημένα στοιχεία ταυτοποίησης (password hashing μέσω `password_hash()`), το όνομα, το email και την κατάσταση επαλήθευσης του λογαριασμού κάθε χρήστη.

- `datasets`: Καταγράφει τα μεταφορτωμένα αρχεία CSV, αποθηκεύοντας τη φυσική διαδρομή τους (file path), το όνομα, το μέγεθος, την ημερομηνία εισαγωγής και το επίπεδο ορατότητας (ιδιωτικό/δημόσιο).

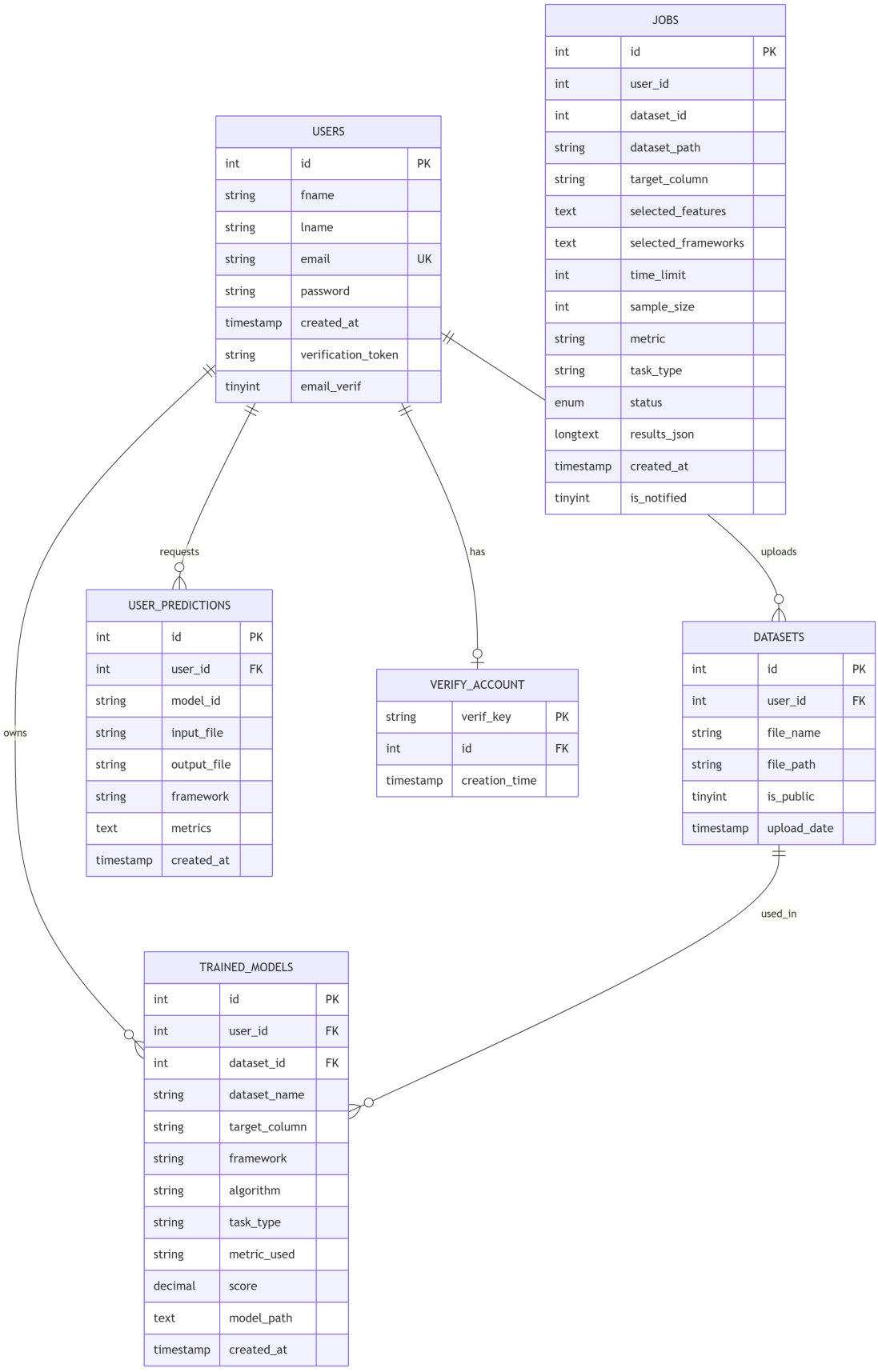
- `jobs`: Λειτουργεί ως πίνακας ελέγχου για τις ασύγχρονες διεργασίες. Καταγράφει τον τύπο της εργασίας (train/predict), την τρέχουσα κατάσταση (*pending, running, success, failed*) και τα χρονικά ορόσημα.

- `trained_models`: Αποθηκεύει τα serialized αρχεία των βέλτιστων μοντέλων που παρήχθησαν, τον αλγόριθμο, το framework, τη μετρική αξιολόγησης και το τελικό σκορ.

- `user_predictions`: Διαχειρίζεται το ιστορικό των προβλέψεων που έχουν εκτελεστεί, συνδέοντας το αρχείο εισόδου, το χρησιμοποιούμενο μοντέλο και το παραγόμενο αρχείο αποτελεσμάτων.

- `verify_account`: Αποθηκεύει τα προσωρινά κρυπτογραφικά κλειδιά (tokens) και τις ημερομηνίες λήξης για τη διαδικασία ενεργοποίησης των νέων λογαριασμών μέσω email.

Οι μεταξύ τους συσχετίσεις βασίζονται σε foreign keys που επιβάλλουν αναφορική ακεραιότητα (referential integrity). Για τη διατήρηση της συνοχής κατά τη διαγραφή εγγραφών, εφαρμόζονται κανόνες `ON DELETE CASCADE` (π.χ. όταν διαγράφεται ένας χρήστης, διαγράφονται αυτόματα τα datasets και τα μοντέλα του) ή `ON DELETE SET NULL`, όπου απαιτείται διατήρηση του ιστορικού.
Το Μοντέλο Οντοτήτων - Συσχετίσεων (ER Diagram) που αποτυπώνει τη λογική σχεδίαση της βάσης δεδομένων `automl_platform` παρουσιάζεται στο Σχήμα 6.3.



Σχήμα 6.3: Διάγραμμα Οντοτήτων - Συσχετίσεων (ER) της βάσης δεδομένων automl_platform.

6.3.2 Σχεσιακό Σχήμα και Ανάλυση Δομής Πινάκων

Για την ολοκληρωμένη καταγραφή της σχεδίασης, στον Πίνακα 6.1 παρουσιάζεται το σχεσιακό σχήμα της βάσης, αποτυπώνοντας συγκεντρωτικά τα πρωτεύοντα κλειδιά (PK), τα ξένα κλειδιά (FK) καθώς και την πληθυντικότητα (cardinality) των μεταξύ τους συσχετίσεων.

Πίνακας 6.1: Σύνοψη Πινάκων, Κλειδιών και Συσχετίσεων της Βάσης Δεδομένων.

Πίνακας	**Πρωτεύον Κλειδί (PK)**	**Ξένο Κλειδί (FK)**	**Συσχέτιση**
`users`	`id`	-	-
`datasets`	`id`	`user_id` → `users.id`	1:N (Ένας χρήστης κατέχει πολλά datasets)
`jobs`	`id`	-	-
`trained_models`	`id`	`user_id` → `users.id` `dataset_id` → `datasets.id`	1:N (Ένας χρήστης έχει πολλά μοντέλα) 1:N (Ένα dataset παράγει πολλά μοντέλα)
`user_predictions`	`id`	`user_id` → `users.id` `model_id` → `trained_models.id`	1:N (Ένας χρήστης εκτελεί προβλέψεις) 1:N (Ένα μοντέλο παράγει πολλές προβλέψεις)
`verify_account`	`verif_key`	`user_id` → `users.id`	1:1 (Αποκλειστική σύνδεση με έναν λογαριασμό χρήστη)

6.4 Υλοποίηση Backend Web API

Το backend της εφαρμογής αναπτύχθηκε σε PHP, ακολουθώντας την αρχιτεκτονική ενός Web API βασισμένου στο πρωτόκολλο HTTP και στη μεταφορά δεδομένων σε μορφή JSON (JSON-based Web API), προκειμένου να εξυπηρετεί με ασφάλεια και ταχύτητα τις ανάγκες του Frontend.

6.4.1 Διαχείριση Χρηστών και Μηχανισμός Αυθεντικοποίησης

Η διαδικασία εισόδου των χρηστών υλοποιείται μέσω του endpoint `login_process.php`. Το API δέχεται ασύγχρονα αιτήματα με σώμα σε μορφή JSON, πραγματοποιώντας αποκωδικοποίηση μέσω της `json_decode(file_get_contents('php://input'))`.
Η ασφάλεια των ευαίσθητων δεδομένων διασφαλίζεται μέσω των εξής βημάτων:

1. **Προετοιμασία Ερωτημάτων (Prepared Statements):** Για την αποφυγή επιθέσεων τύπου SQL Injection, η αναζήτηση στη βάση γίνεται με χρήση παραμετροποιημένων queries (`$mysqli->prepare()`).

2. **Κρυπτογραφικός Έλεγχος:** Η επαλήθευση των κωδικών δεν γίνεται με plain-text σύγκριση, αλλά μέσω της βιβλιοθήκης `password_verify()`, η οποία ελέγχει την εισαγωγή του χρήστη έναντι του αποθηκευμένου κρυπτογραφικού κατακερματισμού (hash).

3. **Έλεγχος Κατάστασης Λογαριασμού:** Το σύστημα ελέγχει την ιδιότητα `email_verif`. Αν ο λογαριασμός δεν έχει ενεργοποιηθεί, επιστρέφει κατάλληλο κωδικό σφάλματος HTTP 403 (Forbidden).

4. **Διατήρηση Κατάστασης Συνεδρίας (Session Management):** Μετά την επιτυχή ταυτοποίηση, εκκινείται σύστημα συνεδρίας (`session_start()`) και αποθηκεύονται οι μεταβλητές `$_SESSION['user_id']` και `$_SESSION['user_name']`. Με τον τρόπο αυτό, η αυθεντικοποίηση βασίζεται σε κατάσταση (stateful authentication) που διαχειρίζεται ο εξυπηρετητής (server-side sessions).

6.4.2 Δρομολόγηση Διεργασιών Εκπαίδευσης (Job Ingestion)

Η εκκίνηση μιας AutoML διεργασίας πραγματοποιείται μέσω του endpoint `start_training.php`. Το συγκεκριμένο αρχείο υποδέχεται τις παραμέτρους που επέλεξε ο χρήστης από το γραφικό περιβάλλον και τις καταχωρεί στη βάση δεδομένων με κατάσταση `pending`.
Στο παρακάτω τμήμα κώδικα παρουσιάζεται ο τρόπος ασφαλούς παραλαβής, φιλτραρίσματος και εγγραφής της αίτησης εκπαίδευσης στον πίνακα `jobs`:

```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $user_id = $_SESSION['user_id'];
    $dataset_id = $_POST['dataset_id'];
    $dataset_path = $_POST['dataset_path'];
    $target = $_POST['target'];
    $features = $_POST['features'];
    $frameworks = $_POST['frameworks'];
    $time_limit = (int)$_POST['time_limit'];
    $sample_size = isset($_POST['sample_size']) ? (int)$_POST['
       sample_size'] : null;
    $metric = $_POST['metric'];
    $task_type = $_POST['task_type'];

    $sql = "INSERT INTO jobs (user_id, dataset_id, dataset_path,
       target_column,
            selected_features, selected_frameworks, time_limit,
               sample_size,
            metric, task_type, status) VALUES (?, ?, ?, ?, ?, ?, ?, ?,
               ?, ?, 'pending')";

    $stmt = $mysqli->prepare($sql);
    $stmt->bind_param("iissssiiss", $user_id, $dataset_id,
       $dataset_path,
                      $target, $features, $frameworks, $time_limit,
                      $sample_size, $metric, $task_type);

    if ($stmt->execute()) {
        echo json_encode(["status" => "success", "job_id" => $mysqli->
           insert_id]);
    } else {
```

```
        echo json_encode(["status" => "error", "message" => $mysqli->
            error]);
    }
}
```

Όπως παρατηρείται στον κώδικα, εφαρμόζεται ρητή μετατροπή τύπων δεδομένων (Type Casting) για κρίσιμες μεταβλητές όπως το `time_limit`, ενώ διασφαλίζεται η ακεραιότητα των δεδομένων μέσω παραμετροποιημένων ερωτημάτων (`bind_param`). Η επιτυχής καταχώρηση επιστρέφει ένα JSON αντικείμενο με το μοναδικό `job_id` της διεργασίας, το οποίο επιτρέπει στο Frontend να παρακολουθεί ασύγχρονα την πορεία της εκπαίδευσης.

6.4.3 Διασύνδεση με το Επίπεδο Μηχανικής Μάθησης (Inference Integration)

Η πιο κρίσιμη λειτουργία του Backend API αφορά την εκτέλεση προβλέψεων σε πραγματικό χρόνο μέσω του `run_inference.php`. Το συγκεκριμένο endpoint γεφυρώνει το περιβάλλον Web της PHP με το υπολογιστικό επίπεδο της Python (Python Execution Layer). Αναλαμβάνει την παραλαβή του νέου αρχείου CSV, την επαλήθευση της ύπαρξης του σειριοποιημένου (serialized) μοντέλου σε μορφή `.joblib` με βάση τις προδιαγραφές του Scikit-Learn οικοσυστήματος [25], την ασφαλή κλήση του Python script μέσω του λειτουργικού συστήματος και την επεξεργασία της JSON απάντησης.
Στο παρακάτω δομικό απόσπασμα αποτυπώνεται ο μηχανισμός δυναμικής σύνθεσης της εντολής, η ασφαλής εκτέλεση στο Linux shell και η στρατηγική απομόνωσης του JSON string από τα streams εξόδου:

```
$model_filename = $_POST['model_path'] ?? '';
$framework = $_POST['framework'] ?? '';
$base_models_dir = "/var/www/html/webkmeans/kclusterhub/automl/models";

$model_filename = str_replace(['\\', '/'], '/', $model_filename);
$full_model_path = $base_models_dir . '/' . ltrim($model_filename, '/')
    ;

if (is_dir($full_model_path) && strtolower($framework) === 'flaml') {
    $full_model_path = rtrim($full_model_path, '/') . '/flaml_model.
        joblib';
}

if (!file_exists($full_model_path)) {
    throw new Exception("Το μοντέλοδενβρέθηκε  .");
}

$python_path = "/var/www/html/webkmeans/kclusterhub/automl/server/
    python/venv/bin/python3";
$script_path = realpath(__DIR__ . '/../python/predict.py');

$cmd = sprintf('%s %s %s %s %s %s 2>&1',
    $python_path, escapeshellarg($script_path), escapeshellarg(
        $full_model_path),
```

```
    escapeshellarg($temp_file), escapeshellarg($output_file),
       escapeshellarg($framework)
);

$python_output = shell_exec($cmd);

$start_pos = strrpos($python_output, '{"status":');
if ($start_pos !== false) {
    $json_data = substr($python_output, $start_pos);
    $end_pos = strrpos($json_data, '}');
    if ($end_pos !== false) {
        $json_data = substr($json_data, 0, $end_pos + 1);
    }
    $response_from_python = json_decode(trim($json_data), true);
} else {
    throw new Exception("H Python δενεπέστρεψε  JSON.");
}
```

Η αρχιτεκτονική προσέγγιση του συγκεκριμένου endpoint επιλύει δύο βασικά προβλήματα των ετερογενών συστημάτων:

1. **Ασφάλεια Συστήματος:** Η χρήση της συνάρτησης `escapeshellarg()` αποτρέπει επιθέσεις τύπου Command Injection, αποστειρώνοντας τις παραμέτρους εισόδου πριν αυτές προωθηθούν στο Linux shell.

2. **Αξιοπιστία Επικοινωνίας (Output Sanitization):** Η Python ενδέχεται κατά την εκκίνηση του εικονικού περιβάλλοντος ή των βιβλιοθηκών να παράγει προειδοποιήσεις (warnings) ή μηνύματα αποσφαλμάτωσης στο standard output. Με τη χρήση της `strrpos()` και τον εντοπισμό των αγκυλών, το API απομονώνει με ακρίβεια το JSON αντικείμενο, αγνοώντας τυχόν θόρυβο κειμένου, διασφαλίζοντας έτσι ότι η `json_decode()` θα εκτελεστεί επιτυχώς.

Μετά την επιτυχή λήψη της κατάστασης `SUCCESS`, οι μετρικές αποθηκεύονται στον πίνακα `user_predictions` και το API επιστρέφει στο Frontend ένα καθαρό JSON object με το URL λήψης του αρχείου των νέων προβλέψεων.

6.4.4 Αρχιτεκτονική Endpoints

Το backend αποτελείται από ένα σύνολο εξειδικευμένων σεναρίων PHP, καθένα από τα οποία εκτελεί μια συγκεκριμένη λειτουργία:

- **Διαχείριση Συνεδρίας / Χρηστών:** `register_process.php`, `login_process.php`, `logout.php`, `verify_process.php`, `pass-reset.php`, `update_password.php`.
- **Διαχείριση Δεδομένων (Datasets):** `save_dataset.php`, `get_user_datasets.php`, `download_dataset.php`, `delete_dataset.php`.
- **Διαχείριση Μοντέλων και Διεργασιών:** `start_training.php`, `get_job_status.php`, `get_user_models.php`, `download_model.php`, `delete_model.php`.

- **Εκτέλεση Συμπερασμού (Inference):** `run_inference.php`, `get_user_predictions.php`, `delete_prediction.php`.
- **Υποστηρικτικές Λειτουργίες:** `global_functions.php`, `check_notifications.php`, `mark_as_seen.php`, `google-callback.php` καθώς και ενσωμάτωση του υποφακέλου `PHPMailer` για την αυτοματοποιημένη αποστολή μηνυμάτων ηλεκτρονικού ταχυδρομείου μέσω του `mailer_config.php` και `phpmailer_func.php`.

6.5 Υλοποίηση Επιπέδου Εκτέλεσης Python (Python Execution Layer)

Το υπολογιστικό επίπεδο της εφαρμογής αναπτύχθηκε εξ ολοκλήρου σε περιβάλλον Python, αξιοποιώντας ένα απομονωμένο εικονικό περιβάλλον (Virtual Environment - venv). Η συγκεκριμένη αρχιτεκτονική επιλογή διασφαλίζει τη σταθερότητα των εκδόσεων των βιβλιοθηκών Μηχανικής Μάθησης και αποτρέπει διενέξεις (conflicts) με τα πακέτα του λειτουργικού συστήματος.
Το επίπεδο εκτέλεσης υποδιαιρείται σε δύο κύριες λειτουργίες:

1. **Υπηρεσία Παρασκηνίου (Background Worker):** Ένας διαρκής μηχανισμός (daemon-like service) που ελέγχει ασύγχρονα τη βάση δεδομένων και συντονίζει την παράλληλη ή σειριακή εκτέλεση των AutoML frameworks.
2. **Σενάρια Αυτοματοποιημένης Εκπαίδευσης και Συμπερασμού (Train & Inference Scripts):** Εξειδικευμένα scripts που αναλαμβάνουν την προεπεξεργασία, την εύρεση βέλτιστου μοντέλου και την παραγωγή προβλέψεων.

6.5.1 Ασύγχρονος Συντονιστής Διεργασιών (Background Worker)

Το αρχείο `worker.py` αποτελεί την «καρδιά» του συστήματος χρονοπρογραμματισμού και δρομολόγησης. Λειτουργεί ως ένας ασύγχρονος βρόχος που διαχειρίζεται την ουρά των αιτημάτων, απομονώνει την εκτέλεση των AutoML frameworks και αναδεικνύει το βέλτιστο μοντέλο.
Snippet 1: Διαχείριση της ουράς εργασιών Ο Worker εκτελείται συνεχώς, ανιχνεύοντας νέα αιτήματα στη βάση δεδομένων:

```
while True:
    job = get_job() # Έλεγχος για νέες εργασίες με κατάσταση 'pending'
    if job:
        update_job_status(job['id'], 'processing')
        # ... έναρξη διαδικασίας εκπαίδευσης ...
    time.sleep(5) # Καθυστέρηση μεταξύ των ελέγχων για εξοικονόμηση πόρων
```

Snippet 2: Εκτέλεση των AutoML Frameworks Η κλήση των εξωτερικών scripts γίνεται μέσω του υποσυστήματος `subprocess`, διασφαλίζοντας ότι κάθε framework εκτελείται σε απομονωμένο περιβάλλον διεργασίας:

```
process = subprocess.run([
    VENV_PYTHON, script_path, dataset_full_path,
    job['target_column'], job['selected_features'],
```

```
    str(job['time_limit']), str(job['metric']), job['task_type'], str(job_id)
], capture_output=True, text=True, encoding='utf-8')
# Ανάλυση αποτελεσμάτων μέσω stdout για την ανάκτηση του best_score
```

Snippet 3: Επιλογή του βέλτιστου μοντέλου (Champion Model Selection) Μετά την εκτέλεση όλων των frameworks, το σύστημα συγκρίνει τα σκορ βάσει της μετρικής που επέλεξε ο χρήστης:

```
is_minimizing = job['metric'].lower() in ['rmse', 'mse', 'mae', 'logloss']
# Αν η μετρική είναι σφάλμα (minimizing), αναζητούμε την ελάχιστη τιμή
if (current_score < best_score) if is_minimizing else (current_score > best_score):
    best_score = current_score
    best_model_data = res_data # Ανάδειξη νικητή και αποθήκευση στη DB
```

Η αρχιτεκτονική αυτή επιλύει κρίσιμα προβλήματα διαχείρισης πόρων:

- **Απομόνωση Σφαλμάτων (Fault Isolation):** Αν ένα framework καταρρεύσει (π.χ. λόγω έλλειψης μνήμης), ο Worker συνεχίζει κανονικά την εκτέλεση των υπολοίπων.
- **Δυναμική Βελτιστοποίηση:** Μέσω της μεταβλητής `is_minimizing`, το σύστημα προσαρμόζει αυτόματα τη λογική σύγκρισης (για σφάλματα αναζητά την ελάχιστη τιμή, για accuracy τη μέγιστη).
- **Διαχείριση Αποθηκευτικού Χώρου:** Ο Worker διαγράφει άμεσα τα αρχεία των μοντέλων που «ηττήθηκαν» στη σύγκριση, διατηρώντας μόνο το βέλτιστο (Garbage Collection).

6.6 Υλοποίηση Επιπέδου AutoML (Python Backends)

Το βασικότερο χαρακτηριστικό του συστήματος είναι η υποστήριξη πολλαπλών μηχανών Αυτοματοποιημένης Μηχανικής Μάθησης (AutoML Frameworks), η θεωρητική και συστηματική προσέγγιση των οποίων εναρμονίζεται με τις σύγχρονες μεθοδολογίες αυτοματοποίησης αγωών δεδομένων [8, 6]. Η εκπαίδευση πραγματοποιείται μέσω αυτόνομων σεναρίων Python, τα οποία καλούνται ασύγχρονα από τον Background Worker. Προκειμένου να διασφαλιστεί η σταθερότητα της πλατφόρμας, σε κάθε script ενσωματώθηκαν εξειδικευμένες τεχνικές προεπεξεργασίας, μετασχηματισμού και σειριοποίησης.

6.6.1 Μηχανή FLAML (Fast Lightweight AutoML)

Το σενάριο `flaml_train.py` αποτελεί την υλοποίηση με χρήση της βιβλιοθήκης FLAML, η οποία διακρίνεται στη βιβλιογραφία για την υπολογιστική της αποδοτικότητα και την οικονομία πόρων [10, 11]. Λόγω πρόσφατων ενημερώσεων στη βιβλιοθήκη Pandas, η εισαγωγή του τύπου `StringDtype` προκαλούσε σφάλματα ασυμβατότητας με τους εσωτερικούς Scikit-Learn αλγορίθμους του FLAML. Η επίλυση επιτεύχθηκε με τη ρητή μετατροπή των χαρακτηριστικών σε αντικείμενα (`object`):

```
# Αντιμετώπιση ασυμβατότητας StringDtype με Scikit-Learn / FLAML
X = X.astype(object)

# Κατηγορική κωδικοποίηση χαρακτηριστικών με διαχείριση NaN τιμών
label_encoders = {}
for col in X.columns:
```

```
    if X[col].dtype == 'object' or str(X[col].dtype).startswith('string'):
        le = LabelEncoder()
        X[col] = X[col].astype(str).replace('nan', 'Unknown')
        X[col] = le.fit_transform(X[col])
        label_encoders[col] = le
```

Επιπλέον, επειδή το FLAML δεν αποθηκεύει αυτόματα τους Encoders, για την αποφυγή του φαινομένου «Data Leakage» κατά το inference, το μοντέλο και οι Encoders εγκλωβίζονται σε μια ενιαία δομή δεδομένων `.joblib`:

```
# Εγκιβωτισμός μοντέλου και μετασχηματιστών σε κοινό αρχείο
model_data = {
    "model": automl,
    "encoders": label_encoders,
    "target_encoder": target_encoder,
    "features": list(X.columns),
    'target_col': target_column
}
model_filename = f"model_job_{job_id}.joblib"
joblib.dump(model_data, model_filename)
```

6.6.2 Μηχανή H2O AutoML

Η βιβλιοθήκη H2O AutoML βασίζεται σε μια αυτόνομη εικονική μηχανή Java (JVM Instance), προσφέροντας οριζόντια κλιμάκωση και κατανεμημένη εκπαίδευση [15]. Η βασική πρόκληση στο σενάριο `h2o_train.py` ήταν η ορθή μετατροπή των δομών δεδομένων σε `H2OFrame` και ο ρητός ορισμός της μεταβλητής-στόχου ως παράγοντα (`asfactor()`) για προβλήματα ταξινόμησης, καθώς χωρίς αυτή τη ρύθμιση το H2O εκτελούσε λανθασμένα παλινδρόμηση:

```
# Αρχικοποίηση H2O Cluster και μετατροπή δομών δεδομένων
h2o.init(nthreads=-1, max_mem_size="4G")
h2o.no_progress()

train = h2o.H2OFrame(train_df)
test = h2o.H2OFrame(test_df)

# Διασφάλιση εκτέλεσης ταξινόμησης αντί για παλινδρόμηση
if task_type == 'classification':
    train[target_column] = train[target_column].asfactor()
    test[target_column] = test[target_column].asfactor()
```

Για τη διατήρηση της ακεραιότητας των πόρων του συστήματος, το script εξάγει το βέλτιστο μοντέλο σε συμπιεσμένη μορφή (MOJO/Binary) και μεριμνά για τον υποχρεωτικό τερματισμό του cluster στο block `finally`:

```
# Εξαγωγή βέλτιστου μοντέλου και ασφαλής τερματισμός JVM
model_path = h2o.save_model(model=leader, path=model_dir, force=True)
shutil.make_archive(model_zip_full_path, 'zip', root_dir=model_dir)

finally:
    try:
        h2o.cluster().shutdown() # Αποδέσμευση μνήμης RAM από το σύστημα
```

```
    except:
        pass
```

6.6.3 Μηχανή MLJAR AutoML

Το framework MLJAR διακρίνεται για την ικανότητα παραγωγής αναλυτικών αναφορών επεξηγηματικής τεχνητής νοημοσύνης (XAI) και τη βελτιστοποίηση μέσω σύνθετων συνόλων (Ensembles) [16]. Στο σενάριο `mljar_train.py`, η βιβλιοθήκη αρχικοποιείται σε κατάσταση `Explain` (ή εναλλακτικά `Compete` για μέγιστη ακρίβεια μέσω Ensembles), ενώ υλοποιείται ένας δυναμικός μηχανισμός ανάγνωσης του παραγόμενου `leaderboard.csv` για την εξαγωγή του νικηφόρου αλγορίθμου, φιλτράροντας τις εσωτερικές Baseline μετρικές:

```
# Δυναμική παραμετροποίηση MLJAR AutoML
automl = AutoML(
    results_path=results_path,
    total_time_limit=time_limit,
    mode="Explain",
    ml_task=ml_task,
    eval_metric=eval_metric,
    random_state=123
)
automl.fit(X_train, y_train)

# Αναλυτική εξαγωγή του βέλτιστου αλγορίθμου από το Leaderboard
if os.path.exists(lb_path):
    lb = pd.read_csv(lb_path)
    lb = lb[lb["model_type"] != "Baseline"] # Αφαίρεση στατιστικού baseline
    if user_metric in ["rmse", "mae", "mse", "logloss"]:
        lb = lb.sort_values("metric_value", ascending=True)
    else:
        lb = lb.sort_values("metric_value", ascending=False)
    best_algo = lb.iloc[0]["model_type"]
```

Καθώς το MLJAR παράγει έναν ολόκληρο κατάλογο αρχείων, το script συμπιέζει αυτόματα (zip) όλα τα παραγόμενα artifacts και τα απαραίτητα metadata, καθιστώντας τα έτοιμα για μεταφορά προς τη βάση δεδομένων και το Frontend:

```
# Σειριοποίηση Target Encoder και συμπίεση φακέλου MLJAR
if task_type == 'classification' and target_encoder is not None:
    joblib.dump(target_encoder, os.path.join(results_path, "
   mljar_target_encoder.joblib"))

% Συμπίεση του συνόλου των artifacts σε ενιαίο αρχείο .zip
shutil.make_archive(model_name, 'zip', results_path)
shutil.rmtree(results_path) # Καθαρισμός προσωρινών αρχείων
```

6.6.4 Μηχανισμός Παραγωγής Προβλέψεων (Inference Engine)

Η διαδικασία της εκτέλεσης προβλέψεων πάνω σε νέα, άγνωστα δεδομένα υλοποιείται από το αυτόνομο σενάριο `predict.py`. Το συγκεκριμένο script σχεδιάστηκε με γνώμονα την καθολική συμβατότητα, καθώς είναι σε θέση να αναγνωρίζει δυναμικά το framework προέλευσης

του μοντέλου (`h2o`, `mljar` ή `flaml`), να αποσυμπιέζει τις απαραίτητες δομές σε προσωρινούς καταλόγους ανάλογα με το Process ID (`os.getpid()`) για την αποφυγή race conditions, και να εφαρμόζει τις κατάλληλες μεθοδολογίες inference.
Ένα από τα πλέον εξελιγμένα χαρακτηριστικά του `predict.py` είναι η **Έξυπνη Ανίχνευση Μεταβλητής Στόχου (Dynamic Target Detection)** και η **Δυναμική Ευθυγράμμιση Χαρακτηριστικών**. Κατά τη διαδικασία παραγωγής προβλέψεων, η δομή των εισαγόμενων δεδομένων ενδέχεται να διαφέρει από το dataset εκπαίδευσης. Το script επιλύει αυτό το πρόβλημα μέσω της ανάγνωσης των μεταδεδομένων εκπαίδευσης (`h2o_info.json`, `mljar_info.json` ή του joblib bundle) και της αναδιάταξης των στηλών του νέου αρχείου CSV, ώστε να ταυτίζονται απόλυτα σε όνομα και σειρά με τα χαρακτηριστικά εκπαίδευσης. Αν κάποιο χαρακτηριστικό λείπει, εφαρμόζεται ένας μηχανισμός fallback (απόδοση μηδενικής τιμής), διασφαλίζοντας ότι η ροή δεν θα διακοπεί.
Στο παρακάτω απόσπασμα παρουσιάζεται η αρχιτεκτονική διακλάδωσης του `predict.py` για τη διαχείριση των H2O, MLJAR και FLAML μοντέλων, καθώς και η λογική της αυστηρής ευθυγράμμισης των χαρακτηριστικών εισόδου:

```
        # --- FRAMEWORK LOGIC & DYNAMIC TARGET DETECTION ---
        if framework == "h2o":
            import h2o
            h2o.init(nthreads=-1, max_mem_size="2G")
            h2o_started = True

            extract_path = os.path.join(os.path.dirname(model_path), f"
    temp_h2o_{os.getpid()}")
            if os.path.exists(extract_path): shutil.rmtree(extract_path)
            with zipfile.ZipFile(model_path, 'r') as zip_ref:
                zip_ref.extractall(extract_path)

            info_path = os.path.join(extract_path, "h2o_info.json")
            trained_features = []
            if os.path.exists(info_path):
                with open(info_path, 'r') as f:
                    info = json.load(f)
                    target_col_name = info.get('target_col')
                    trained_features = info.get('features', [])

            model_files = [f for f in os.listdir(extract_path) if not f.
    endswith('.json')]
            if not model_files: raise Exception("No H2O model file found in
    zip.")
            model = h2o.load_model(os.path.join(extract_path, model_files[0]))

            X_input = df.copy()
            if target_col_name and target_col_name in X_input.columns:
                X_input = X_input.drop(columns=[target_col_name])
            X_input = X_input.loc[:, ~X_input.columns.str.contains('^Unnamed|^
    index', case=False)]

            if len(X_input.columns) >= len(trained_features):
                X_final_pd = X_input.iloc[:, :len(trained_features)]
                X_final_pd.columns = trained_features
            else:
```

```
            X_final_pd = X_input

        h2o_X = h2o.H2OFrame(X_final_pd)
        preds_h2o = model.predict(h2o_X)
        preds_pd = preds_h2o.as_data_frame()
        predictions = preds_pd.iloc[:, 0].values

    elif framework == "mljar":
        extract_path = os.path.join(os.path.dirname(model_path), f"
temp_mljar_{os.getpid()}")
        if os.path.exists(extract_path): shutil.rmtree(extract_path)
        with zipfile.ZipFile(model_path, 'r') as zip_ref:
            zip_ref.extractall(extract_path)

        info_path = os.path.join(extract_path, "mljar_info.json")
        if os.path.exists(info_path):
            with open(info_path, 'r') as f:
                info = json.load(f)
                target_col_name = info.get('target_col')

        data_info_path = os.path.join(extract_path, "data_info.json")
        trained_features = []
        if os.path.exists(data_info_path):
            with open(data_info_path, 'r') as f:
                d_info = json.load(f)
                trained_features = d_info.get('columns', [])

        if target_col_name and target_col_name in trained_features:
            trained_features = [f for f in trained_features if f !=
target_col_name]

        X_input = df.copy()
        X_input = X_input.loc[:, ~X_input.columns.str.contains('^Unnamed|^
index', case=False)]

        X_final = pd.DataFrame(index=X_input.index)
        for col in trained_features:
            if col in X_input.columns:
                X_final[col] = X_input[col]
            else:
                X_final[col] = 0  # Fallback μηχανισμός αν λείπει κάποια
στήλη

        X_final = X_final[trained_features]
        automl = AutoML(results_path=extract_path)
        preds_raw = automl.predict(X_final)
        preds_flat = np.array(preds_raw).flatten()

        encoder_path = os.path.join(extract_path, "mljar_target_encoder.
joblib")
        if os.path.exists(encoder_path):
            target_encoder = joblib.load(encoder_path)
            preds_int = np.array([int(round(float(v))) for v in preds_flat
])
```

```
                predictions = target_encoder.inverse_transform(preds_int)
            else:
                predictions = preds_flat
```

Επιπλέον, το script υλοποιεί έναν μηχανισμό **Αυτόματου Υπολογισμού Μετρικών Ποιότητας (On-the-fly Evaluation)** σε περίπτωση που το εισαγόμενο αρχείο περιέχει ήδη τη στήλη-στόχο (π.χ. κατά τη διαδικασία επικύρωσης με ένα test set). Για την αποφυγή σφαλμάτων που σχετίζονται με τη γραφή των χαρακτήρων (Case-Sensitivity), το σύστημα μετατρέπει τις στήλες σε πεζά στοιχεία (`lowercase`) και πραγματοποιεί strip των κενών διαστημάτων, εντοπίζοντας με ασφάλεια τη μεταβλητή-στόχο (π.χ. ανίχνευση της στήλης «CLASS» όταν ο στόχος ορίστηκε ως «class»).
Στη συνέχεια, εξετάζεται δυναμικά ο τύπος του προβλήματος. Αν τα δεδομένα είναι αριθμητικά, το πρόβλημα αντιμετωπίζεται ως Παλινδρόμηση (Regression) και υπολογίζονται οι μετρικές RMSE, MSE, MAE και R^2 Score. Σε αντίθετη περίπτωση, το πρόβλημα αντιμετωπίζεται ως Ταξινόμηση (Classification) και υπολογίζονται οι μετρικές Accuracy, F1-Score, Precision και Recall. Η μαθηματική επικύρωση και ο υπολογισμός των μετρικών αυτών βασίζονται στις τυποποιημένες ακαδημαϊκές προδιαγραφές αξιολόγησης μοντέλων [3] και υλοποιούνται μέσω της βιβλιοθήκης Scikit-Learn [25]:

```
        # --- SMART METRICS SETUP & CASE-SENSITIVITY HANDLING ---
        if target_col_name:
            cols_lowercase = [str(c).lower().strip() for c in df.columns]
            target_lower = str(target_col_name).lower().strip()

            if target_lower in cols_lowercase:
                matched_idx = cols_lowercase.index(target_lower)
                actual_target_in_file = df.columns[matched_idx]
                has_target = True

        # --- METRICS CALCULATION ---
        metrics = {}
        if has_target:
            try:
                from sklearn.metrics import (accuracy_score, f1_score,
    precision_score, recall_score, mean_squared_error, mean_absolute_error,
    r2_score)
                y_true = df[actual_target_in_file]
                try:
                    y_true_num = pd.to_numeric(y_true, errors='raise')
                    y_pred_num = pd.to_numeric(pd.Series(clean_preds), errors=
    'raise')
                    is_regression = True
                except:
                    is_regression = False

                if is_regression:
                    mse = mean_squared_error(y_true_num, y_pred_num)
                    metrics = {
                        "type": "regression",
                        "target_detected": actual_target_in_file,
                        "RMSE": round(float(np.sqrt(mse)), 4),
                        "MSE": round(float(mse), 4),
```

```
                    "MAE": round(float(mean_absolute_error(y_true_num,
y_pred_num)), 4),
                    "R2 Score": round(float(r2_score(y_true_num,
y_pred_num)), 4)
                }
            else:
                y_true_str = y_true.astype(str).str.strip()
                y_pred_str = pd.Series(clean_preds).astype(str).str.strip
()
                metrics = {
                    "type": "classification",
                    "target_detected": actual_target_in_file,
                    "Accuracy": f"{round(float(accuracy_score(y_true_str,
y_pred_str)) * 100, 2)}%",
                    "F1-Score": round(float(f1_score(y_true_str,
y_pred_str, average='weighted', zero_division=0)), 4),
                    "Precision": round(float(precision_score(y_true_str,
y_pred_str, average='weighted', zero_division=0)), 4),
                    "Recall": round(float(recall_score(y_true_str,
y_pred_str, average='weighted', zero_division=0)), 4)
                }
        except Exception as me:
            metrics = {"error": f"Metrics failed: {str(me)}"}
```

Στο τελικό στάδιο εκτέλεσης, οι καθαρισμένες προβλέψεις ενσωματώνονται στο αρχικό DataFrame ως νέα στήλη `prediction_result` και το αρχείο εξάγεται στην ορισμένη διαδρομή εξόδου σε μορφή CSV. Το script επιστρέφει μια αυστηρά δομημένη JSON απάντηση προς το υποσύστημα της PHP, η οποία περιλαμβάνει την κατάσταση της διεργασίας (`SUCCESS/ERROR`), τις παραχθείσες μετρικές και τυχόν μηνύματα σφαλμάτων, διασφαλίζοντας την απρόσκοπτη επικοινωνία μεταξύ του Python Execution Layer και της Web διεπαφής. Στο μπλοκ `finally`, πραγματοποιείται αυτόματος καθαρισμός (Garbage Collection) των προσωρινών καταλόγων αποσυμπίεσης, αποτρέποντας τη συσσώρευση άχρηστων δεδομένων στον δίσκο του εξυπηρετητή.

6.7 Υλοποίηση του Frontend

Η διεπαφή χρήστη (Frontend) του AutoML App αναπτύχθηκε ως μια σύγχρονη, single-page-like διαδικτυακή εφαρμογή (SPA) με βάση τα πρότυπα HTML5, CSS3 και JavaScript (ES6+). Για τη διαχείριση του DOM, τον χειρισμό των γεγονότων (Event Handling) και την υλοποίηση της ασύγχρονης επικοινωνίας χρησιμοποιήθηκε η βιβλιοθήκη jQuery. Η μορφοποίηση και η responsive συμπεριφορά βασίστηκαν στο framework Bootstrap 5, διασφαλίζοντας συμβατότητα με πολλαπλές αναλύσεις οθόνης μέσω Grid System και Media Queries.

6.7.1 Αρχιτεκτονική Δομή και Οργάνωση Αρχείων

Η ανάπτυξη του Frontend οργανώθηκε με αυστηρό διαχωρισμό δομής, εμφάνισης και λογικής (Separation of Concerns). Το client-side τμήμα της εφαρμογής δομείται από τα ακόλουθα κεντρικά αρχεία:

- `index.php` και `login.php`: Λειτουργούν ως τα δημόσια σημεία εισόδου, ενσωματώνοντας φόρμες επικύρωσης και το Google OAuth 2.0 API.
- `dashboard.php`: Ο κεντρικός σκελετός HTML/PHP του εσωτερικού περιβάλλοντος, ο οποίος περιλαμβάνει τη βασική διάταξη Sidebar–Content Layout και τα layouts των επιμέρους ενοτήτων σε μορφή κρυφών blocks (`style="display:none;"`).
- `dashboard.css`: Το αρχείο στυλ όπου ορίζονται οι CSS μεταβλητές (CSS Variables) για τη χρωματική παλέτα, οι κανόνες διάταξης του flexbox, το animation των overlays αναμονής και τα media queries για αναλύσεις κάτω από 992px (ενεργοποίηση Bootstrap Offcanvas).
- `dashboard.js`: Ο υπολογιστικός πυρήνας του Frontend, ο οποίος ενορχηστρώνει όλη τη client-side λογική και τις AJAX κλήσεις.

6.7.2 Δυναμική Διαχείριση Διεπαφής (dashboard.js)

Η εναλλαγή μεταξύ των λειτουργικών ενοτήτων (*Train Model, My Datasets, My Models, Predict, My Predictions*) πραγματοποιείται δυναμικά χωρίς επαναφόρτωση της σελίδας, βελτιώνοντας την εμπειρία χρήστη (User Experience). Η λογική αυτή βασίζεται σε έναν κεντρικό Event Listener που ανιχνεύει τα κλικ στο sidebar:

```
$(document).on('click', '.sidebar-link', function(e) {
    e.preventDefault();
    const section = $(this).data('section');

    // Ενημέρωσηενεργήςκλάσηςστομενού
    $('.sidebar-link').removeClass('active');
    $('[data-section="' + section + '"]').addClass('active');

    // Dynamic Section Toggle με Fade Animation
    $('.content-section').hide();
    $('#section-' + section).fadeIn(200);

    // Lazy Loading Δεδομένωνανάλογαμετο    Section
    if (section === 'datasets') loadUserDatasets();
    if (section === 'models') loadUserModels();
    if (section === 'my-predictions') loadUserPredictions();
    if (section === 'predict') updatePredictModelsDropdown();
});
```

6.7.3 Τοπική Ανάλυση Δεδομένων μέσω FileReader API

Ένα από τα σημαντικότερα τεχνικά χαρακτηριστικά του Frontend είναι η αποφυγή άσκοπων network requests για την προεπισκόπηση και ανάλυση των αρχείων. Όταν ο χρήστης επιλέγει ένα αρχείο CSV, το `dashboard.js` χρησιμοποιεί το native JavaScript `FileReader API` για την τοπική ανάγνωση των δεδομένων στον browser:

- **Parsing και Tokenization:** Το αρχείο διαβάζεται ως κείμενο (`readAsText`) και απομονώνονται οι πρώτες γραμμές. Ο διαχωρισμός των στηλών γίνεται δυναμικά (ανίχνευση `,`, `;` ή `\t`).
- **Δυναμική Δημιουργία DOM:** Με βάση την κεφαλίδα (header row), παράγονται αυτόματα `checkboxes` για την επιλογή των Features και `options` για το dropdown της μεταβλητής Target.
- **Client-side Type Inference:** Μέσω επαναληπτικού ελέγχου των τιμών της επιλεγμένης στήλης Target, το Frontend υπολογίζει το πλήθος των μοναδικών τιμών (cardinality). Αν οι μοναδικές τιμές είναι αριθμητικές και ξεπερνούν ένα προκαθορισμένο όριο, η διεπαφή προεπιλέγει αυτόματα το *Regression Task*, αλλιώς ενεργοποιεί το *Classification Task*.

6.7.4 Ασύγχρονη Επικοινωνία και Μηχανισμός Polling

Η επικοινωνία με το PHP Backend είναι πλήρως ασύγχρονη και βασίζεται σε HTTP POST και GET αιτήματα με ανταλλαγή δεδομένων αποκλειστικά σε μορφή JSON.
Λόγω της χρονικά απαιτητικής φύσης των διεργασιών AutoML, η υποβολή της φόρμας εκπαίδευσης (`start_training.php`) δεν κρατά την HTTP σύνδεση ανοιχτή (non-blocking). Το Backend επιστρέφει άμεσα ένα `job_id` και το Frontend εισέρχεται σε κατάσταση Long Polling, ακολουθώντας σύγχρονες αρχιτεκτονικές προσεγγίσεις για τη διαχείριση ασύγχρονων Web υπηρεσιών [34].
Η τεχνική υλοποίηση της παρακολούθησης βασίζεται στη χρήση της `setInterval()` στην πλευρά του client. Συγκεκριμένα, η εφαρμογή εκτελεί περιοδικά (κάθε 3 δευτερόλεπτα) ασύγχρονες AJAX κλήσεις για τον έλεγχο της κατάστασης της εργασίας μέσω του `get_job_status.php?job_`

- **Κατάσταση Pending / Running:** Το Frontend διατηρεί κλειδωμένη τη διεπαφή εμφανίζοντας ένα Bootstrap Spinner Overlay, εμποδίζοντας διπλές υποβολές.
- **Κατάσταση Success:** Ο timer ακυρώνεται μέσω της `clearInterval()`, το overlay κλείνει και το JSON response, που περιέχει την τελική κατάταξη (ranking) των μοντέλων, γίνεται parse στην πλευρά του client.
- **Δυναμικό Rendering Αποτελεσμάτων:** Το JavaScript κατασκευάζει on-the-fly έναν πίνακα HTML, ταξινομώντας τα frameworks με βάση τη μετρική αξιολόγησης, και εισάγει το αποτέλεσμα σε ένα Bootstrap Modal (`trainResultsModal`) χρησιμοποιώντας τη μέθοδο `$().html()`.

6.7.5 Ασφάλεια και Διαχείριση Client-Side Κατάστασης

Το Frontend υλοποιεί ελέγχους ασφαλείας πριν την αποστολή οποιουδήποτε αιτήματος στο Backend. Πριν από κάθε AJAX POST (π.χ. στην ενότητα `Predict Analysis`), ο κώδικας JavaScript επικυρώνει (validate) τη συμβατότητα του επιλεγμένου μοντέλου με τα χαρακτηριστικά του νέου CSV αρχείου.
Επιπλέον, όλες οι φόρμες περιλαμβάνουν client-side validation για την αποφυγή υποβολής κενών ή λανθασμένων παραμέτρων (π.χ. έλεγχος αν ο μέγιστος χρόνος εκπαίδευσης είναι θετικός ακέραιος), μειώνοντας τον φόρτο εργασίας και τα σφάλματα στο επίπεδο του διακομιστή.

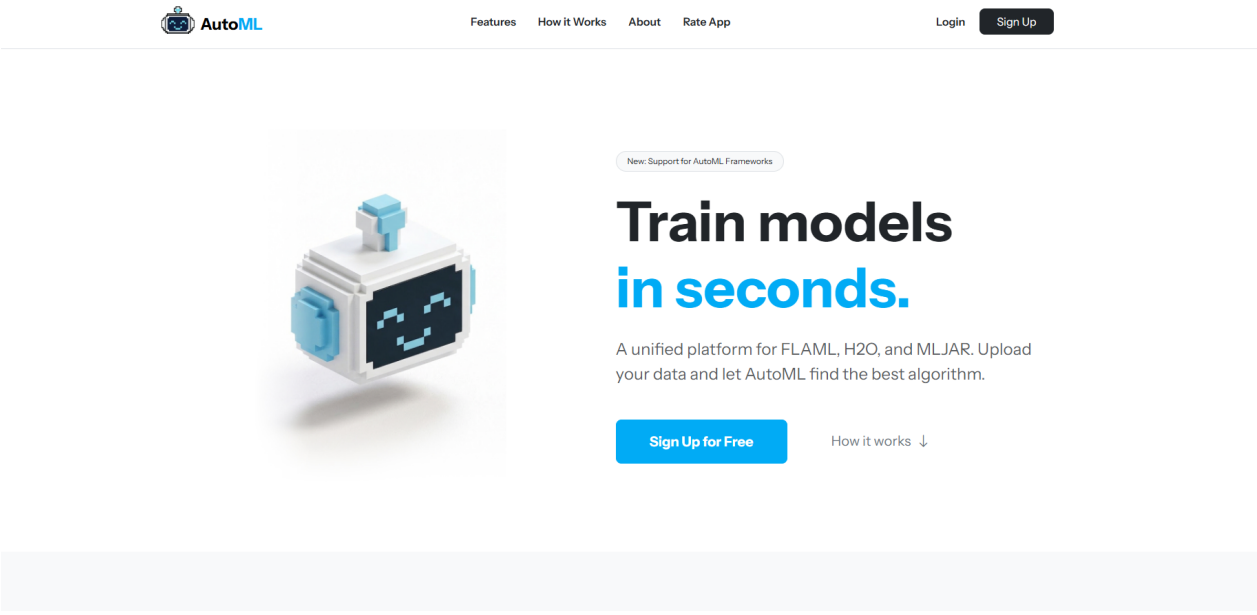
Κεφάλαιο 7

Παρουσίαση της Εφαρμογής AutoML App

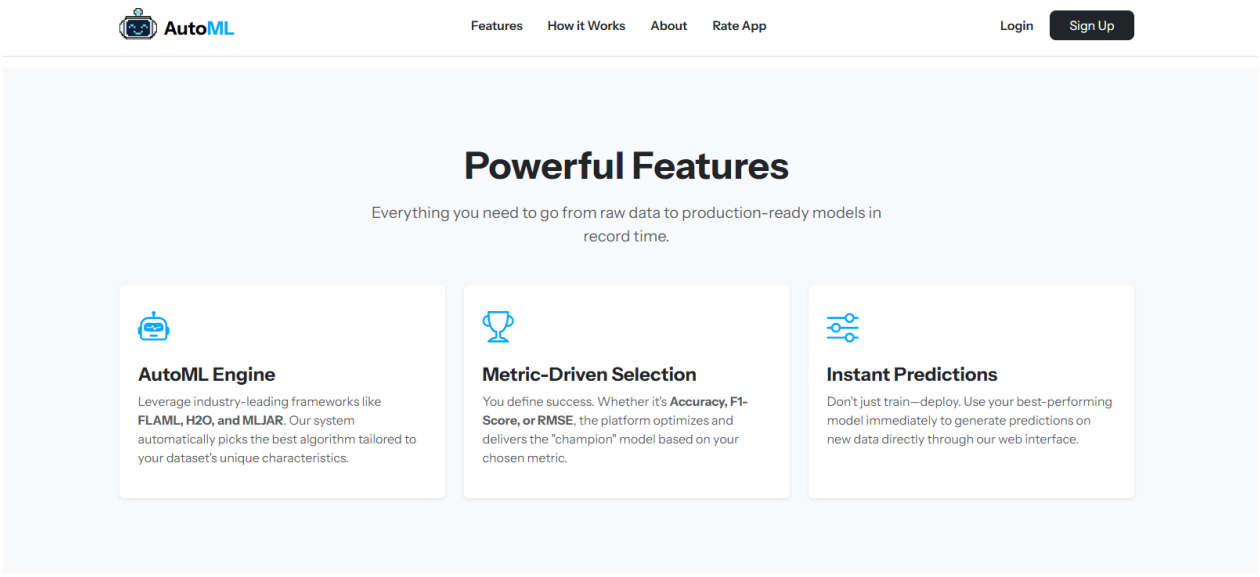
Στο παρόν κεφάλαιο πραγματοποιείται η αναλυτική παρουσίαση της τελικής διεπαφής και λειτουργίας της εφαρμογής AutoML App. Στόχος είναι η περιγραφή από την οπτική του τελικού χρήστη, αναλύοντας τη γραφική διεπαφή, τις επιλογές παραμετροποίησης και τις λειτουργικές ενότητες.

7.1 Αρχική Σελίδα και Πλοήγηση

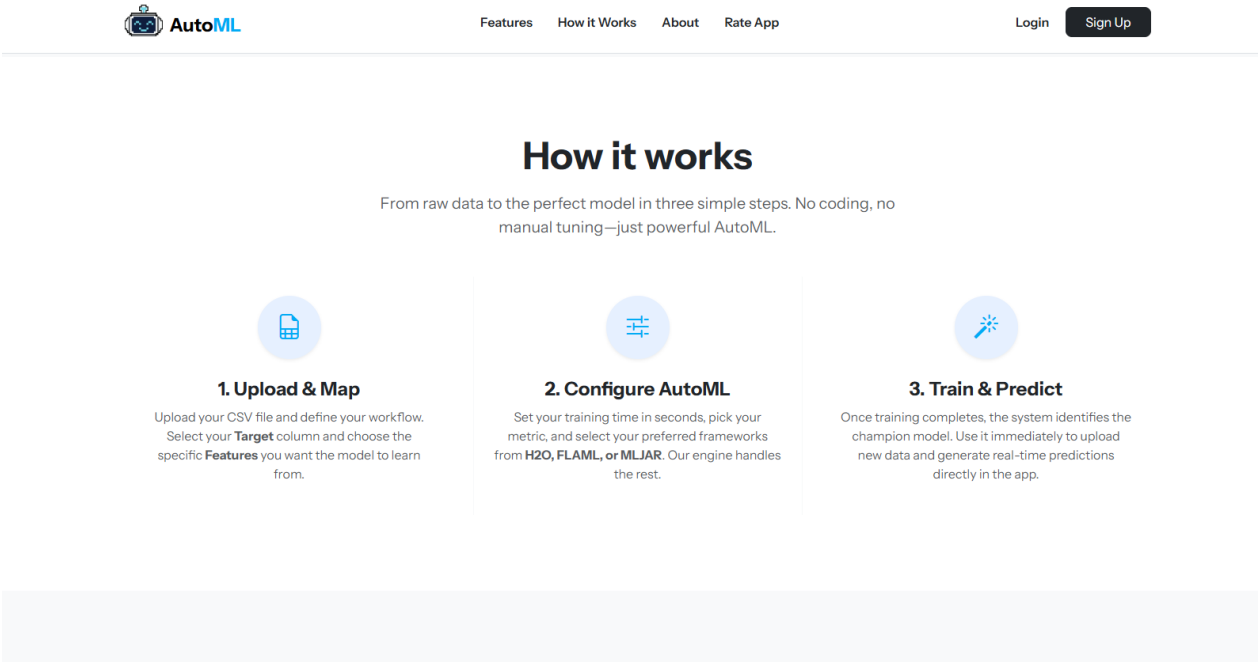
Η πλατφόρμα εισάγει τον χρήστη μέσω μιας ενιαίας σελίδας (Landing Page) με δυναμική πλοήγηση. Η σελίδα περιλαμβάνει διακριτά sections στα οποία ο χρήστης μεταβαίνει μέσω ομαλού scroll:



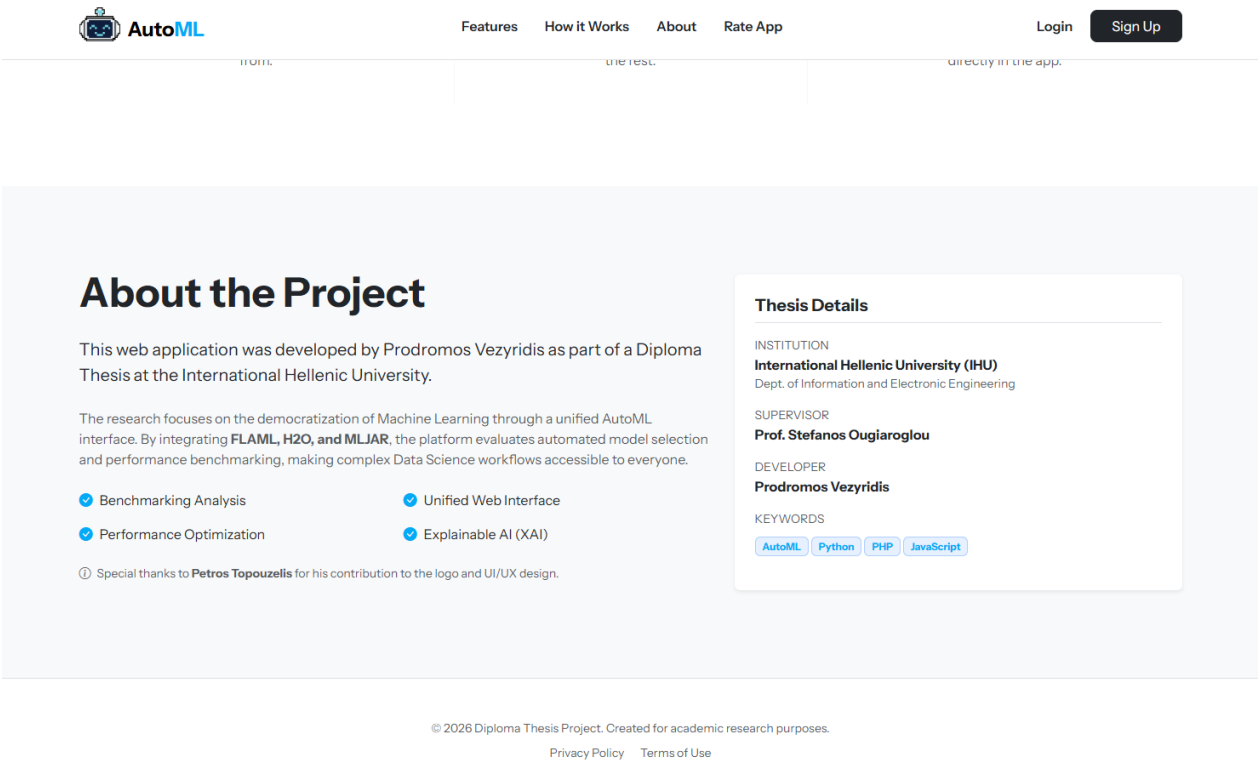
Σχήμα 7.1: Κεντρικό section (Hero) της αρχικής σελίδας.



Σχήμα 7.2: Ενότητα Features (Χαρακτηριστικά).



Σχήμα 7.3: Ενότητα How it Works (Πώς λειτουργεί).

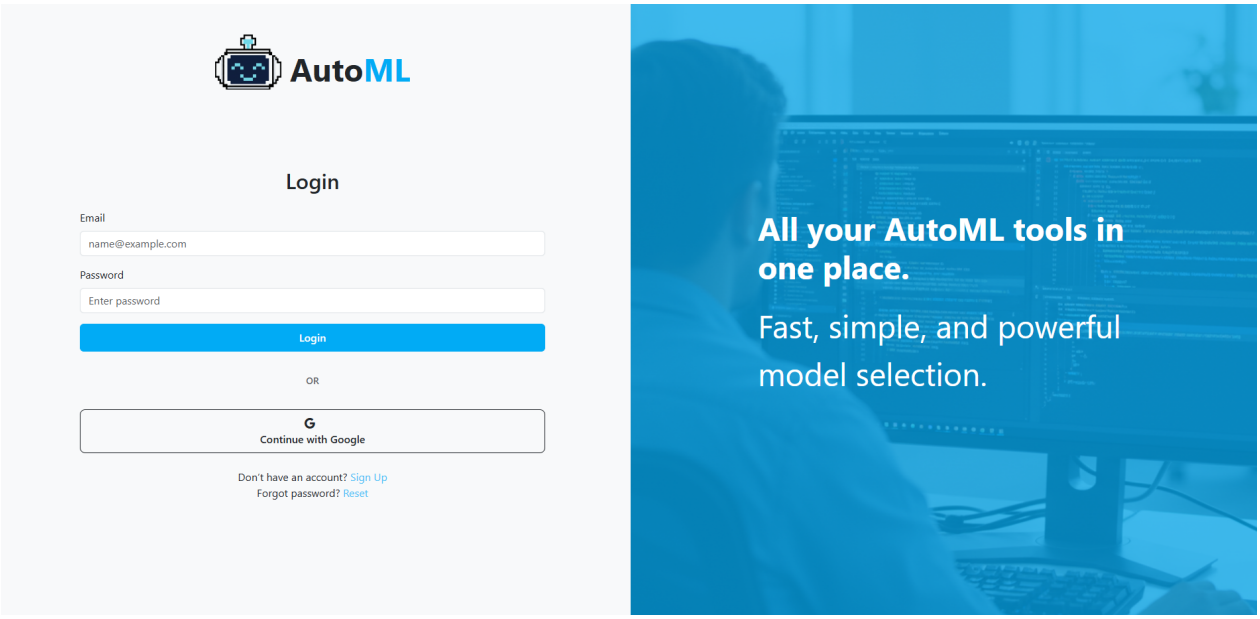


Σχήμα 7.4: Ενότητα About (Σχετικά με την εφαρμογή).

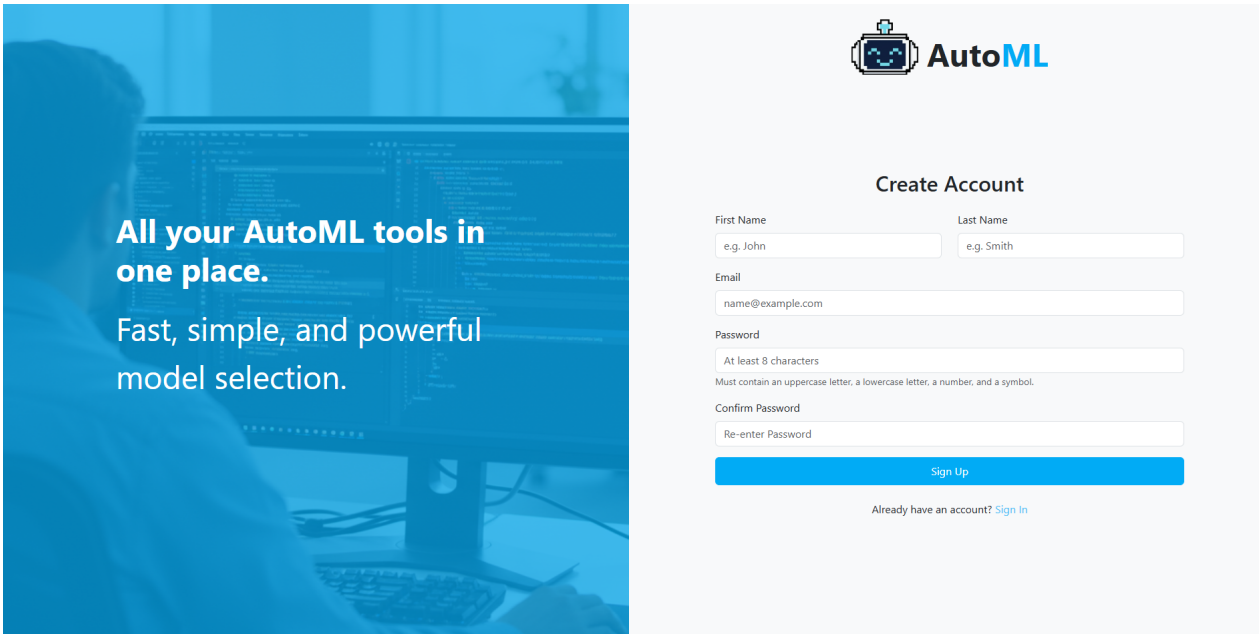
Σημείωση Branding: Ο οπτικός σχεδιασμός και η αισθητική αρτιότητα της διεπαφής πραγματοποιήθηκαν με την πολύτιμη συνεισφορά του κ. Πέτρου Τοπουζέλη, ο οποίος επιμελήθηκε το branding και το UI/UX της πλατφόρμας.

7.2 Αυθεντικοποίηση Χρήστη

Οι σελίδες *Login* και *Register* ακολουθούν μια responsive διάταξη, διασφαλίζοντας ομοιογένεια στην εμπειρία χρήστη:



Σχήμα 7.5: Διεπαφή Σύνδεσης (Login).



Σχήμα 7.6: Διεπαφή Εγγραφής (Register).

7.3 Ροή Εργασίας από την Οπτική του Χρήστη

Η τυπική αλληλεπίδραση ενός χρήστη με την πλατφόρμα εκτυλίσσεται μέσω της ακόλουθης λειτουργικής σειράς:

1. **Εισαγωγή και Προεπισκόπηση Δεδομένων:** Ο χρήστης μεταφορτώνει το CSV αρχείο. Το σύστημα επεξεργάζεται άμεσα το αρχείο (client-side) και εμφανίζει προεπισκόπηση.
2. **Παραμετροποίηση και Αυτόματη Αναγνώριση:** Ο χρήστης επιλέγει τη μεταβλητή-στόχο, το σύστημα αναγνωρίζει τον τύπο του προβλήματος, και ο χρήστης ορίζει τα features, το time budget και τις μετρικές.
3. **Εκτέλεση και Αποτελέσματα:** Η εκπαίδευση εκτελείται ασύγχρονα στο background. Μετά την ολοκλήρωση, το σύστημα αναδεικνύει το βέλτιστο μοντέλο.
4. **Παραγωγή Προβλέψεων:** Χρήση του παραχθέντος μοντέλου για την εξαγωγή προβλέψεων σε νέα δεδομένα.

7.4 Διαχείριση Πόρων και Λειτουργικές Ενότητες (Modules)

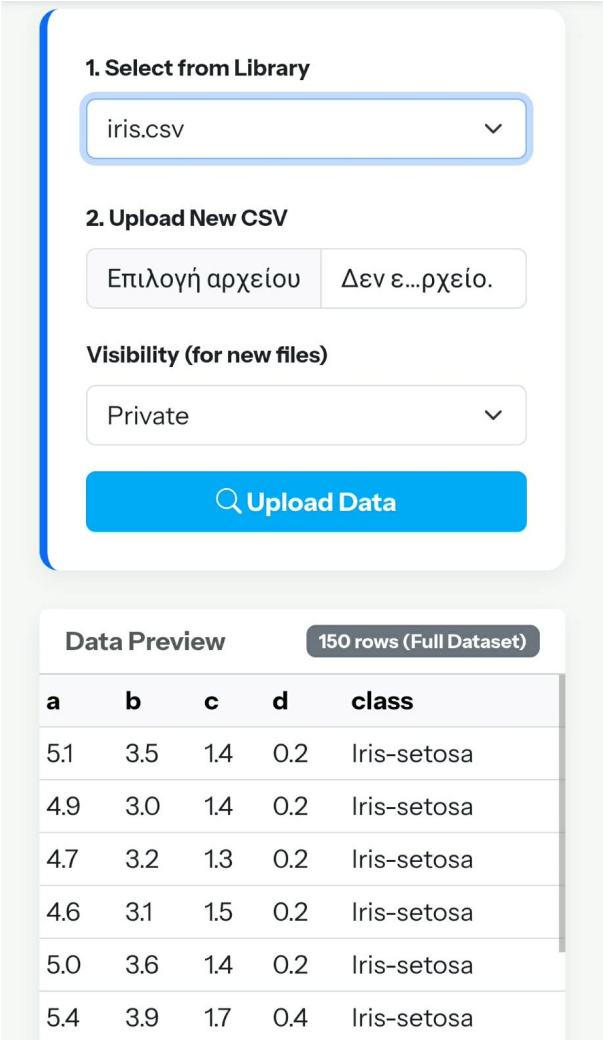
Η εφαρμογή περιλαμβάνει επιπλέον modules για τη διαχείριση της πληροφορίας:

7.4.1 Modules Διαχείρισης (Datasets, Models, Predictions)

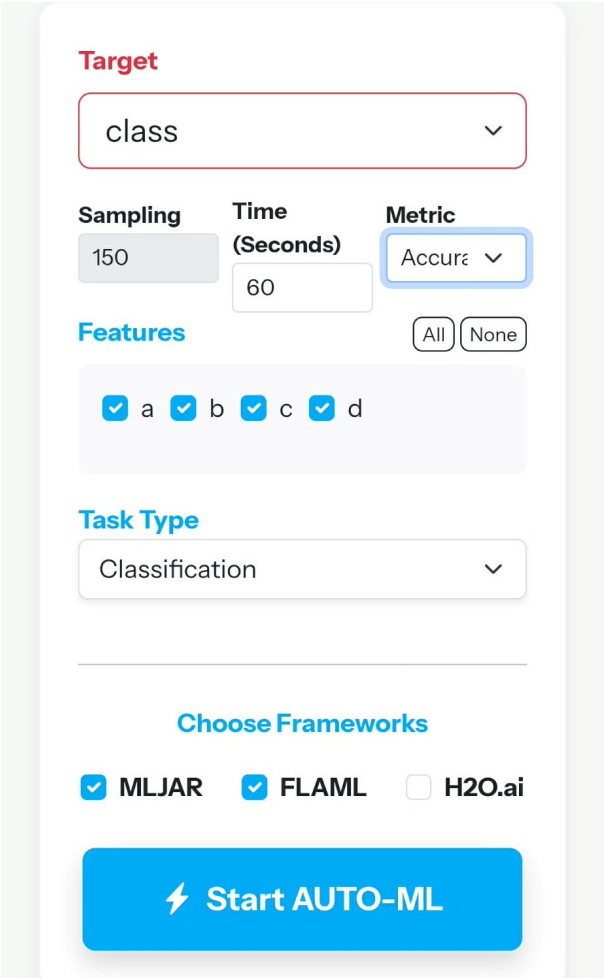
- **My Datasets:** Συγκεντρωτική βιβλιοθήκη για τον διαχωρισμό ιδιωτικών και δημόσιων συνόλων δεδομένων.
- **My Models:** Χώρος αποθήκευσης των βέλτιστων μοντέλων, με δυνατότητα λήψης (serialized files) και περαιτέρω χρήσης για προβλέψεις.
- **My Predictions:** Ιστορικό αρχείο επιτυχών προβλέψεων, όπου ο χρήστης μπορεί να ανατρέξει σε αποτελέσματα και να κατεβάσει τα εξαγόμενα αρχεία.

7.4.2 Εκπαίδευση Μοντέλου (Train Model)

Αποτελεί τον πυρήνα της αλληλεπίδρασης με την πλατφόρμα. Εδώ ο χρήστης πραγματοποιεί τη μεταφόρτωση του αρχείου CSV ή επιλέγει ένα ήδη αποθηκευμένο σύνολο δεδομένων, λαμβάνοντας άμεση προεπισκόπηση των περιεχομένων του σε μορφή δυναμικού πίνακα. Μόλις ο χρήστης επιλέξει τη μεταβλητή-στόχο, η διεπαφή ενημερώνεται αυτόματα εμφανίζοντας τον τύπο του προβλήματος (classification ή regression) που αναγνώρισε το σύστημα.
Κατόπιν, ο χρήστης προχωρά στη ρύθμιση των υπόλοιπων παραμέτρων του πειράματος: επιλέγει τα επιθυμητά χαρακτηριστικά εισόδου (features), ορίζει τον διαθέσιμο χρόνο εκπαίδευσης (time budget), επιλέγει τη συγκεκριμένη μετρική αξιολόγησης βάσει της οποίας θα γίνει η βελτιστοποίηση και επιλέγει ποια AutoML frameworks θα εκτελεστούν παράλληλα. Προαιρετικά, παρέχεται η δυνατότητα ενεργοποίησης της δειγματοληψίας δεδομένων (data sampling) για τη βελτιστοποίηση των απαιτούμενων υπολογιστικών πόρων.



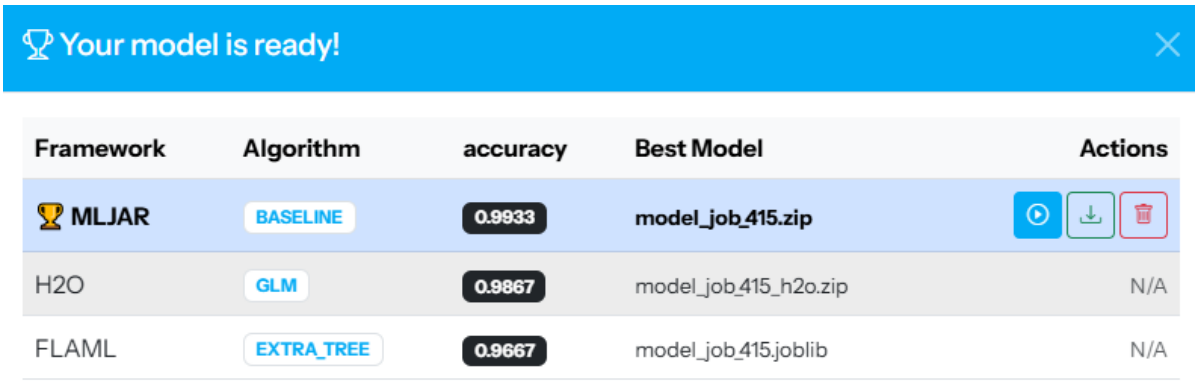
(a) Επιλογή και προεπισκόπηση CSV.



(b) Παραμετροποίηση εκπαίδευσης.

Σχήμα 7.7: Η διαδικασία προετοιμασίας της εκπαίδευσης: (a) Μεταφόρτωση και επισκόπηση του συνόλου δεδομένων και (b) Ρύθμιση παραμέτρων και αυτόματη επιλογή τύπου προβλήματος.

Μετά την ολοκλήρωση της εκτέλεσης, το module αυτό εμφανίζει ένα αναδυόμενο παράθυρο (modal window), στο οποίο πραγματοποιείται η τελική κατάταξη (ranking) και σύγκριση των αποτελεσμάτων όλων των frameworks με βάση αποκλειστικά τη μετρική αξιολόγησης που είχε αρχικά επιλέξει ο χρήστης. Με τον τρόπο αυτό, το σύστημα αναδεικνύει με ειδική σήμανση το νικηφόρο μοντέλο και το αποθηκεύει αυτόματα στον λογαριασμό του.

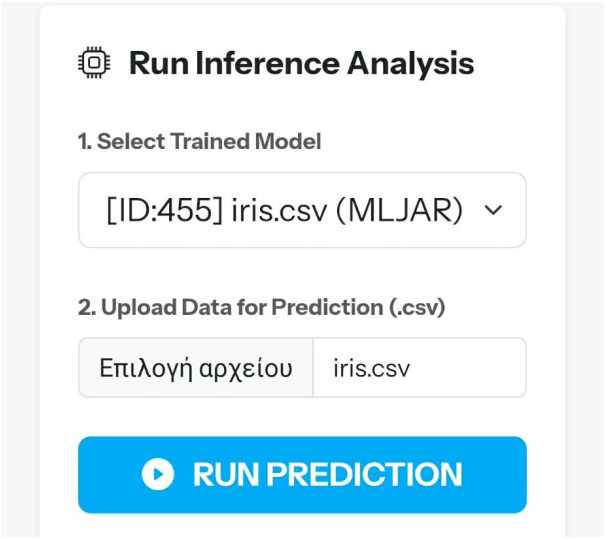


Σχήμα 7.8: Αναδυόμενο παράθυρο (Modal) τελικών αποτελεσμάτων αξιολόγησης και κατάταξης των frameworks βάσει της επιλεγμένης μετρικής.

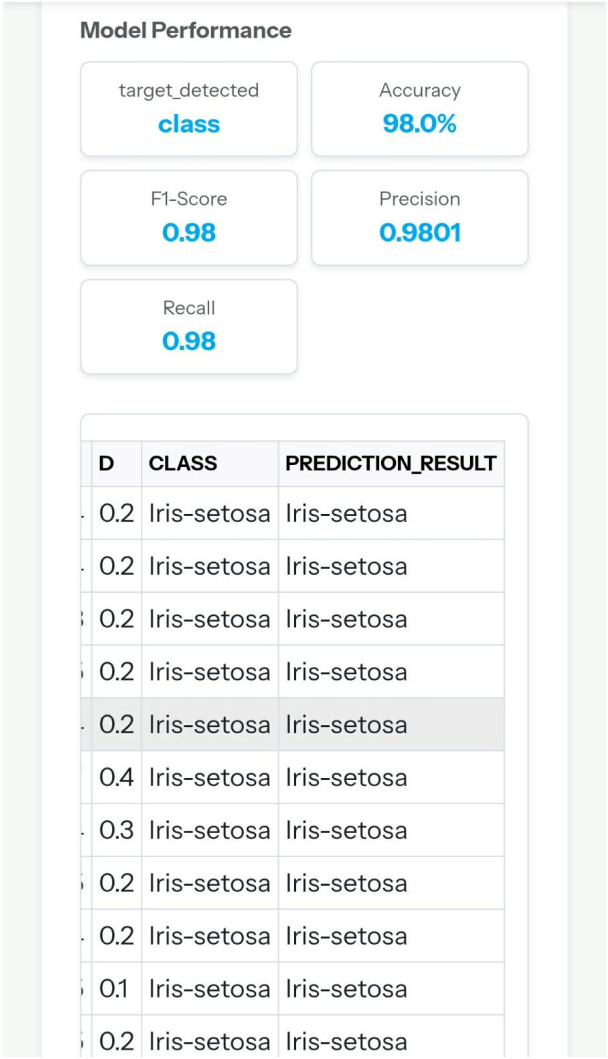
7.4.3 Παραγωγή Προβλέψεων (Predict)

Επιτρέπει στον χρήστη να επιλέξει ένα από τα ήδη εκπαιδευμένα μοντέλα του και να εισάγει ένα νέο αρχείο δεδομένων για πρόβλεψη. Η διεπαφή προσαρμόζεται έξυπνα ανάλογα με τη δομή του εισαγόμενου αρχείου:

- Αν το νέο αρχείο περιέχει τη στήλη-στόχο, η ενότητα εκτελεί διαδικασία επικύρωσης (Validation), εμφανίζοντας τις προβλέψεις και υπολογίζοντας αυτόματα τις μετρικές αξιολόγησης (σφάλματα ή ορθότητα).
- Αν η στήλη-στόχος απουσιάζει, το σύστημα εκτελεί καθαρή πρόβλεψη (Predict) και παράγει μια νέα, συμπληρωμένη έκδοση του αρχείου με τις εκτιμώμενες τιμές.



(a) Εισαγωγή δεδομένων και επιλογή μοντέλου.

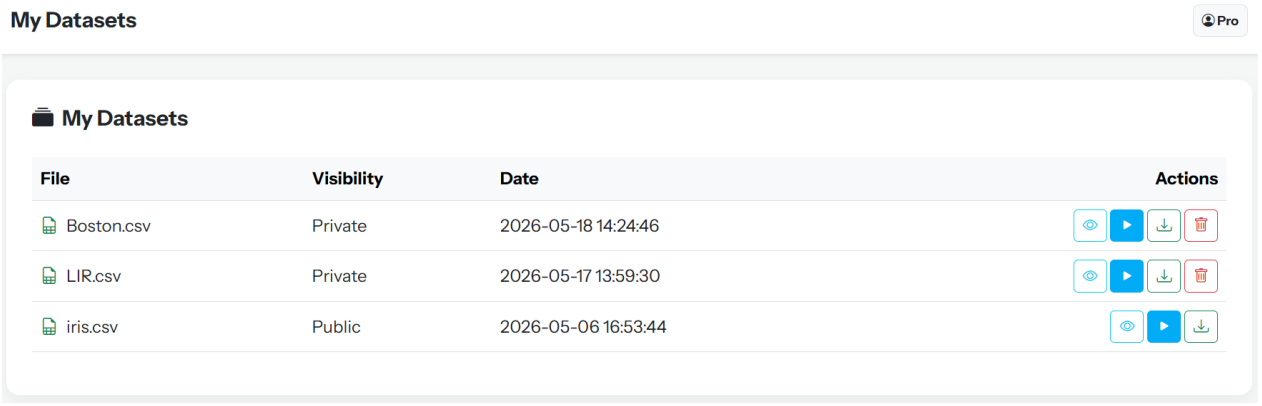


(b) Αποτελέσματα και μετρικές.

Σχήμα 7.9: Η διαδικασία παραγωγής προβλέψεων (Inference): (a) Επιλογή εκπαιδευμένου μοντέλου και μεταφόρτωση νέου συνόλου δεδομένων και (b) Προεπισκόπηση αποτελεσμάτων πρόβλεψης και αυτόματος υπολογισμός μετρικών αξιολόγησης.

7.4.4 Τα Σύνολα Δεδομένων μου (My Datasets)

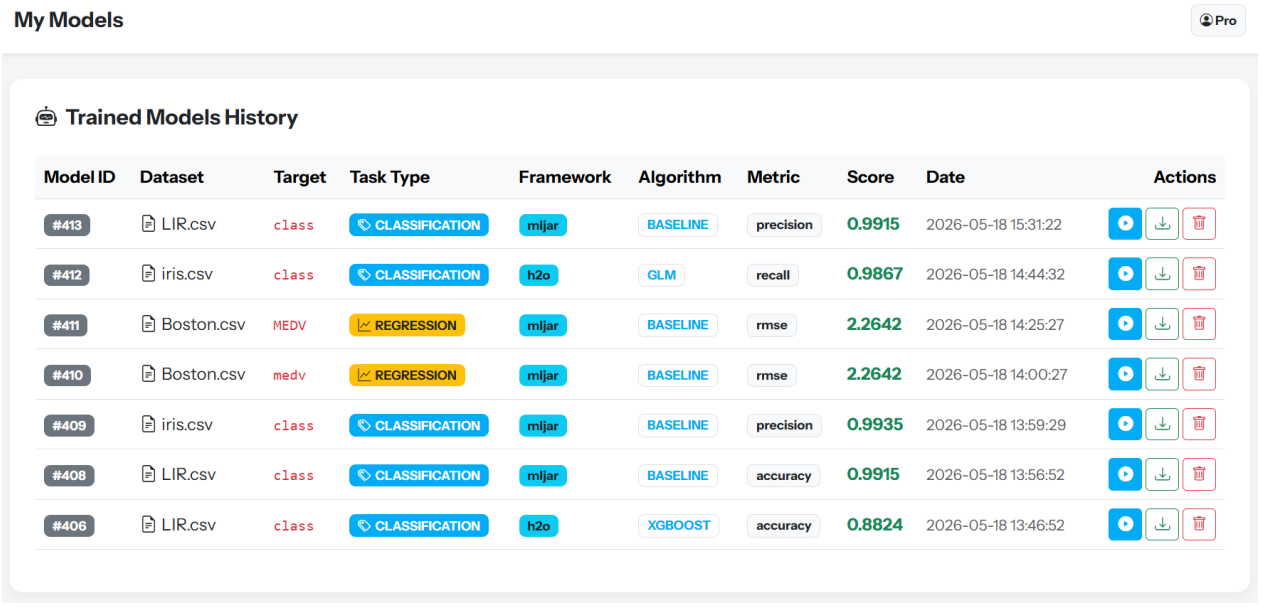
Μια συγκεντρωτική βιβλιοθήκη όπου ο χρήστης διαχειρίζεται τα αρχεία του. Μέσα από έναν διαδραστικό πίνακα, μπορεί να ελέγξει το επίπεδο ορατότητας (Private/Public), να εκτελέσει άμεση προεπισκόπηση των πρώτων 10 γραμμών οποιουδήποτε CSV αρχείου μέσω καθολικού modal ή να επιλέξει ένα αρχείο και να μεταφερθεί αυτόματα στην οθόνη εκπαίδευσης με το συγκεκριμένο dataset προεπιλεγμένο.



Σχήμα 7.10: Πίνακας διαχείρισης και επισκόπησης ιδιωτικών και δημόσιων συνόλων δεδομένων.

7.4.5 Τα Μοντέλα μου (My Models)

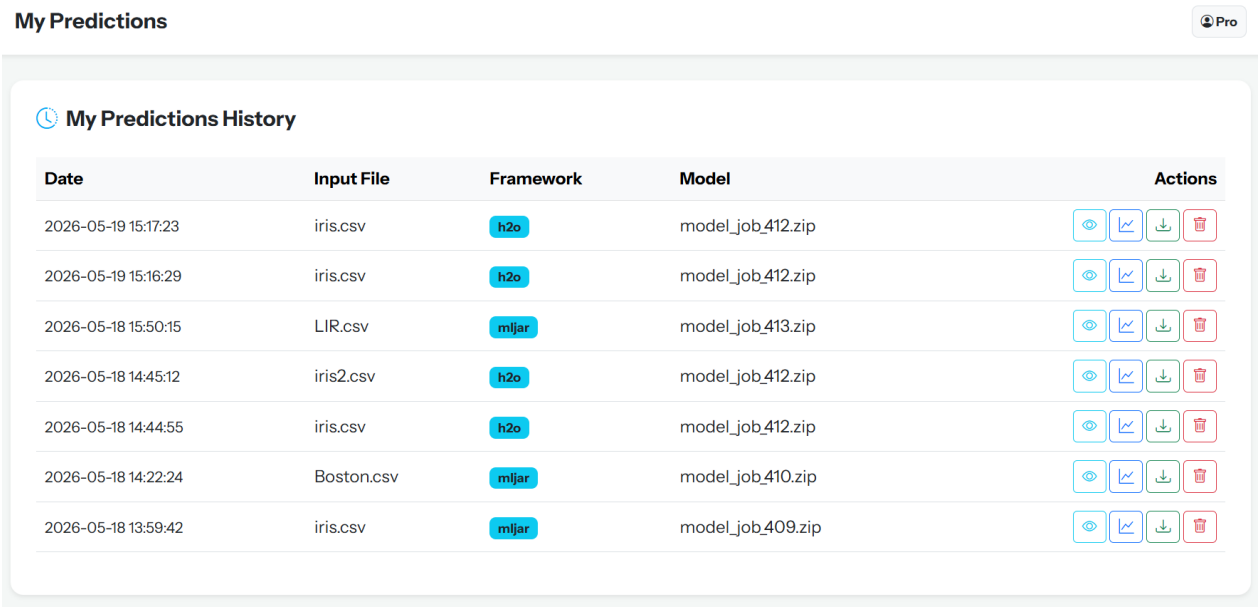
Ο χώρος αποθήκευσης των βέλτιστων μοντέλων που έχουν παραχθεί από τις background εκπαιδεύσεις του χρήστη. Ο χρήστης μπορεί να επιβλέψει τις μετρικές απόδοσης κάθε μοντέλου, τον αλγόριθμο, το framework, να το διαγράψει, να το κατεβάσει τοπικά στον υπολογιστή του (ως serialized αρχείο) ή να μεταβεί απευθείας στην οθόνη προβλέψεων χρησιμοποιώντας το συγκεκριμένο μοντέλο ως επιλεγμένο.



Σχήμα 7.11: Διεπαφή διαχείρισης, λήψης και επισκόπησης των βέλτιστων εκπαιδευμένων μοντέλων.

7.4.6 Οι Προβλέψεις μου (My Predictions)

Το ιστορικό αρχείο όλων των προβλέψεων που έχουν εκτελεστεί επιτυχώς από τον χρήστη στην πλατφόρμα. Ο χρήστης μπορεί να ανατρέξει σε παλαιότερα αποτελέσματα, να εξετάσει τις μετρικές τους, να κατεβάσει το τελικό παραγόμενο αρχείο (το οποίο περιλαμβάνει τα αρχικά δεδομένα μαζί με τη νέα στήλη των προβλέψεων) ή να διαγράψει εγγραφές για να καθαρίσει το ιστορικό του.



Σχήμα 7.12: Ιστορικό αρχείο αποθηκευμένων αποτελεσμάτων πρόβλεψης και λήψης εξαγόμενων αρχείων.

Κεφάλαιο 8

Εμπειρία Χρήσης και Αξιολόγηση

Στο παρόν κεφάλαιο παρουσιάζονται και αναλύονται τα αποτελέσματα της αξιολόγησης της χρηστικότητας (usability evaluation) της προτεινόμενης εφαρμογής AutoML App, με ιδιαίτερη έμφαση στην εμπειρία του χρήστη (user experience). Για τον σκοπό αυτό, διανεμήθηκε στους χρήστες της εφαρμογής ένα ερωτηματολόγιο βασισμένο στην Κλίμακα Χρηστικότητας Συστημάτων (System Usability Scale - SUS) [35]. Οι συμμετέχοντες στην αξιολόγηση ήταν κυρίως προπτυχιακοί φοιτητές που παρακολουθούσαν το μάθημα «Εξόρυξη Δεδομένων» στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος (ΔΙ.ΠΑ.Ε.).

8.1 Η Κλίμακα Χρηστικότητας Συστημάτων (System Usability Scale - SUS)

Η Κλίμακα Χρηστικότητας Συστημάτων (SUS) αποτελεί μια ευρέως διαδεδομένη, γρήγορη και αξιόπιστη μέθοδο για την αποτίμηση της χρηστικότητας ενός ψηφιακού συστήματος. Αποτελείται από δέκα (10) συγκεκριμένες ερωτήσεις, όπου για κάθε μία ο χρήστης καλείται να επιλέξει μία απάντηση σε μια πενταβάθμια κλίμακα Likert, η οποία κυμαίνεται από το «Διαφωνώ Απόλυτα» (τιμή 1) έως το «Συμφωνώ Απόλυτα» (τιμή 5).
Πιο συγκεκριμένα, οι δέκα ερωτήσεις που περιλαμβάνονται στο ερωτηματολόγιο SUS είναι οι ακόλουθες:

1. Πιστεύω ότι θα ήθελα να χρησιμοποιώ αυτόν τον ιστότοπο συχνά.
2. Βρήκα τον ιστότοπο περιττά περίπλοκο.
3. Θεωρώ ότι ο ιστότοπος ήταν εύκολος στη χρήση.
4. Πιστεύω ότι θα χρειαζόμουν την υποστήριξη ενός τεχνικού για να μπορέσω να χρησιμοποιήσω αυτόν τον ιστότοπο.
5. Βρήκα ότι οι διάφορες λειτουργίες του ιστότοπου ήταν πολύ καλά ενσωματωμένες και συνδεδεμένες μεταξύ τους.
6. Θεωρώ ότι υπήρχε μεγάλη ασυνέπεια σε αυτόν τον ιστότοπο.

7. Φαντάζομαι ότι οι περισσότεροι άνθρωποι θα μπορούσαν να μάθουν να χρησιμοποιούν αυτόν τον ιστότοπο πολύ γρήγορα.
8. Βρήκα τον ιστότοπο πολύ δύσχρηστο και δύσκολο στη λειτουργία του.
9. Ένιωσα μεγάλη ασφάλεια και αυτοπεποίθηση κατά τη χρήση του ιστότοπου.
10. Χρειάστηκε να μάθω πολλά πράγματα προτού μπορέσω να ξεκινήσω να χρησιμοποιώ αυτόν τον ιστότοπο.

Για τον υπολογισμό του τελικού σκορ SUS, οι απαντήσεις των χρηστών αντιστοιχίζονται αρχικά σε μια κλίμακα 0-4. Για τις περιττές ερωτήσεις (1, 3, 5, 7, 9), αφαιρείται η μονάδα (1) από την απάντηση του χρήστη. Αντίθετα, για τις άρτιες ερωτήσεις (2, 4, 6, 8, 10), η τιμή της απάντησης του χρήστη αφαιρείται από τον αριθμό πέντε (5). Στη συνέχεια, τα επιμέρους αποτελέσματα αθροίζονται και το τελικό άθροισμα πολλαπλασιάζεται με το 2.5. Η διαδικασία αυτή παράγει ένα τελικό σκορ αξιολόγησης σε μια κλίμακα από το 0 έως το 100 [35].

8.2 Αποτελέσματα Αξιολόγησης και Συζήτηση

Στο πλαίσιο των δοκιμών χρηστικότητας της εφαρμογής AutoML App, συγκεντρώθηκαν απαντήσεις από συνολικά **15 συμμετέχοντες**. Το υπολογισμένο σκορ SUS για κάθε συμμετέχοντα ξεχωριστά, καθώς και ο τελικός γενικός μέσος όρος, παρουσιάζονται αναλυτικά στον Πίνακα 8.1.

Πίνακας 8.1: Αποτελέσματα της Κλίμακας Χρηστικότητας Συστημάτων (SUS) για την εφαρμογή AutoML App

Συμμετέχων (User)	**Σκορ SUS (SUS Score)**
User 1	65.0
User 2	85.0
User 3	60.0
User 4	80.0
User 5	47.5
User 6	87.5
User 7	92.5
User 8	80.0
User 9	97.5
User 10	82.5
User 11	82.5
User 12	90.0
User 13	85.0
User 14	97.5
User 15	85.0
Μέσος Όρος Σκορ	**81.17**

Ο τελικός δείκτης χρηστικότητας προκύπτει από τον μέσο όρο των επιμέρους σκορ όλων των συμμετεχόντων, αποδίδοντας μια συνολική βαθμολογία ίση με **81.17**. Σύμφωνα με τη διεθνή

βιβλιογραφία [35], ένα σκορ SUS άνω του 68 υποδεικνύει ένα αποδεκτό σύστημα με ικανοποιητική χρηστικότητα, ενώ ένα σκορ που ξεπερνά το 80 θεωρείται εξαιρετικό (excellent), αντιστοιχώντας σε βαθμολογική αξιολόγηση επιπέδου "Α". Συνεπώς, τα εμπειρικά αποτελέσματα αποδεικνύουν ότι η προτεινόμενη εφαρμογή AutoML App επιτυγχάνει υψηλά επίπεδα χρηστικότητας, καθώς το σύστημα έγινε αντιληπτό ως εξαιρετικά διαισθητικό, φιλικό και εύκολο στη χρήση από το κοινό-στόχο.

Μια ενδιαφέρουσα παρατήρηση στα δεδομένα της αξιολόγησης αφορά τον χρήστη 5 (User 5), ο οποίος σημείωσε τη χαμηλότερη βαθμολογία (47.5). Η χαμηλή αυτή τιμή οφείλεται ξεκάθαρα σε μια συστηματική τάση συγκατάθεσης (acquiescence bias), καθώς ο συγκεκριμένος συμμετέχων επέλεξε τη μέγιστη απάντηση (5) σε όλες ανεξαιρέτως τις ερωτήσεις, χωρίς να λάβει υπόψη αν η πρόταση ήταν διατυπωμένη θετικά ή αρνητικά (π.χ. συμφώνησε απόλυτα τόσο ότι η εφαρμογή είναι εύκολη, όσο και ότι είναι περιττά περίπλοκη). Παρά την ύπαρξη αυτής της ακραίας τιμής (outlier), ο ισχυρός γενικός μέσος όρος των 81.17 μονάδων υπογραμμίζει τη σταθερή και υψηλή απόδοση της πλατφόρμας στον τομέα της εμπειρίας χρήστη.

Κεφάλαιο 9

Συμπεράσματα και Μελλοντικές Επεκτάσεις

Στο παρόν καταληκτικό κεφάλαιο πραγματοποιείται μια συνολική αποτίμηση της διπλωματικής εργασίας. Αρχικά, επιχειρείται μια ανασκόπηση των στόχων που επιτεύχθηκαν και παρουσιάζονται συγκεντρωτικά τα συμπεράσματα που προέκυψαν από την υλοποίηση και την αξιολόγηση του συστήματος. Στη συνέχεια, αναδεικνύεται η πρωτότυπη συνεισφορά της εργασίας και, τέλος, προτείνονται συγκεκριμένες κατευθύνσεις για μελλοντικές επεκτάσεις και βελτιώσεις της πλατφόρμας.

9.1 Συμπεράσματα

Η παρούσα διπλωματική εργασία εστιάστηκε στη σχεδίαση, υλοποίηση και αξιολόγηση μιας ολοκληρωμένης διαδικτυακής εφαρμογής Αυτοματοποιημένης Μηχανικής Μάθησης (AutoML App). Η πλατφόρμα αυτή επιτυγχάνει την ενσωμάτωση πολλαπλών open-source AutoML frameworks σε ένα ενιαίο, συγκεντρωτικό και φιλικό προς τον χρήστη περιβάλλον, προσαρμοσμένο για εργασίες ταξινόμησης (classification) και παλινδρόμησης (regression). Η εφαρμογή επιτρέπει στους χρήστες να μεταφορτώνουν σύνολα δεδομένων, να εκτελούν αυτοματοποιημένες ροές μηχανικής μάθησης, να συγκρίνουν την απόδοση των μοντέλων και να επιλέγουν το μοντέλο με την υψηλότερη απόδοση.
Μέσα από την ανάπτυξη και τη δοκιμή του συστήματος προέκυψαν τα ακόλουθα βασικά συμπεράσματα:

- **Απλοποίηση της Ανάπτυξης Μοντέλων:** Η ενσωμάτωση διαφορετικών εργαλείων AutoML κάτω από μια κοινή γραφική διεπαφή (Web Interface) απλοποιεί δραστικά τη διαδικασία ανάπτυξης μοντέλων και ενισχύει την προσβασιμότητα για μη εξειδικευμένους χρήστες (non-expert users).

- **Ανταγωνιστική Απόδοση:** Τα πειραματικά αποτελέσματα αποδεικνύουν ότι το σύστημα παρέχει ανταγωνιστική απόδοση σε διαφορετικά σύνολα δεδομένων, επιτρέποντας την εύκολη ανάδειξη του βέλτιστου framework ανά περίπτωση.

- **Ικανοποίηση Χρηστών:** Η αξιολόγηση της χρηστικότητας (usability evaluation) μέσω

της κλίμακας SUS επιβεβαίωσε το υψηλό επίπεδο ικανοποίησης των χρηστών, αναδεικνύοντας την εφαρμογή ως ένα εύχρηστο και αποτελεσματικό εργαλείο.

9.2 Ουσιαστική Συνεισφορά της Εργασίας

Η συνεισφορά της παρούσας διατριβής κινείται σε δύο βασικούς άξονες:

1. **Πειραματική Σύγκριση και Αξιολόγηση:** Πραγματοποιήθηκε μια εκτενής πειραματική ανάλυση και σύγκριση της συμπεριφοράς των AutoML εργαλείων, καταγράφοντας την αποτελεσματικότητά τους σε διάφορα σενάρια δεδομένων.

2. **Υλοποίηση Διαδικτυακής Εφαρμογής (Web App):** Σχεδιάστηκε και αναπτύχθηκε μια πλήρως λειτουργική full-stack πλατφόρμα, η οποία προσφέρει μια συγκεντρωτική ροή εργασίας (workflow). Η εφαρμογή επιλύει το πρόβλημα του κατακερματισμού των εργαλείων AutoML, παρέχοντας στους χρήστες ένα κοινό περιβάλλον για τη διαχείριση δεδομένων, την εκπαίδευση, τη σύγκριση και την παραγωγή προβλέψεων.

9.3 Μελλοντικές Επεκτάσεις

Η εφαρμογή AutoML App έχει σχεδιαστεί με modular αρχιτεκτονική, γεγονός που επιτρέπει την εύκολη επέκτασή της. Οι κυριότερες κατευθύνσεις για μελλοντική έρευνα και ανάπτυξη περιλαμβάνουν:

9.3.1 Ενσωμάτωση Επιπλέον Frameworks

Η μελλοντική έρευνα θα εστιαστεί στην επέκταση του συστήματος με την προσθήκη επιπλέον AutoML frameworks, όπως το Auto-sklearn [7], το TPOT [13] και το LightAutoML [21]. Η προσθήκη αυτή αναμένεται να ενισχύσει περαιτέρω την ποικιλομορφία των παραγόμενων μοντέλων και να διευρύνει τις δυνατότητες σύγκρισης αποδόσεων της πλατφόρμας.

9.3.2 Ερμηνευσιμότητα Μοντέλων (Explainable AI)

Επιπλέον, οι μελλοντικές βελτιώσεις θα διερευνήσουν τεχνικές ερμηνευσιμότητας μοντέλων (model explainability), συμπεριλαμβανομένων των επεξηγήσεων που βασίζονται στις τιμές SHAP (SHapley Additive exPlanations) [33], καθώς και την οπτικοποίηση της σπουδαιότητας των χαρακτηριστικών (feature importance visualization). Στόχος είναι η παροχή καλύτερης ερμηνεύσιμότητας και διαφάνειας για τα μοντέλα που παράγονται αυτόματα από το σύστημα.

9.3.3 Βελτιώσεις Κλιμακωσιμότητας (Scalability)

Παράλληλα, θα εξεταστούν βελτιώσεις σχετικά με την κλιμακωσιμότητα του συστήματος μέσω της πιθανής μεταφοράς και εγκατάστασής του σε περιβάλλον υπολογιστικού νέφους (Cloud Deployment), καθώς και της κατανεμημένης επεξεργασίας των AutoML ροών εργασίας. Αυτό θα επιτρέψει στο σύστημα να διαχειρίζεται μεγαλύτερα σύνολα δεδομένων και πιο απαιτητικά πειράματα από υπολογιστική άποψη.

9.3.4 Εμπειρία Χρήστη και Experiment Tracking

Τέλος, σχεδιάζονται περαιτέρω βελτιώσεις στην εμπειρία χρήστη (user experience) και στην ενσωμάτωση μηχανισμών παρακολούθησης πειραμάτων (experiment tracking). Η προσθήκη αυτή θα προσφέρει στους χρήστες ένα ακόμη πιο ολοκληρωμένο, διαδραστικό και λεπτομερές περιβάλλον για τη διαχείριση ολόκληρου του κύκλου ζωής των πειραμάτων μηχανικής μάθησης.

Βιβλιογραφία

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2009.

[3] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[4] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1137–1143, 1995.

[5] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.

[6] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated Machine Learning: Methods, Systems, Challenges*. Springer Publishing Company, Incorporated, 1st ed., 2019.

[7] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, (Cambridge, MA, USA), p. 2755–2763, MIT Press, 2015.

[8] T. Nagarajah and G. Poravi, "A review on automated machine learning (automl) systems," in *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*, pp. 1–6, IEEE, 2019.

[9] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *NeurIPS Proceedings*, 2012.

[10] C. Wang, Q. Wu, M. Weimer, and E. Zhu, "Flaml: A fast and lightweight automl library," in *Conference on Machine Learning and Systems*, 2019.

[11] C. Wang, Q. Wu, X. Liu, and L. Quintanilla, "Automated machine learning & tuning with flaml," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, (New York, NY, USA), p. 4828–4829, Association for Computing Machinery, 2022.

[12] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-weka: Combined selection and hyperparameter optimization of classification algorithms," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 847–855, Association for Computing Machinery, 2013.

[13] R. S. Olson and J. H. Moore, *TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning*, pp. 151–160. Cham: Springer International Publishing, 2019.

[14] P. Gijsbers and J. Vanschoren, "Gama: A general automated machine learning assistant," *Journal of Open Source Software*, vol. 6, no. 59, p. 2925, 2021.

[15] E. LeDell and S. Poirier, "H2O AutoML: Scalable automatic machine learning," *7th ICML Workshop on Automated Machine Learning (AutoML)*, July 2020.

[16] MLJAR, "Mljar automl documentation," 2024. `https://mljar.com/`.

[17] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola, "Autogluon-tabular: Robust and accurate automl for structured data," in *arXiv preprint arXiv:2003.06505*, 2020.

[18] H. Jin, Q. Song, and X. Hu, "Auto-keras: An efficient neural architecture search system," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1946–1956, 2019.

[19] L. Zimmer, M. Lindauer, and F. Hutter, "Auto-pytorch: Multi-tabular automated machine learning," in *Journal of Machine Learning Research*, vol. 22, pp. 1–6, 2021.

[20] A. Zela, M. Rasekh, T. Elsken, and F. Hutter, "Naslib: A modular and flexible neural architecture search library," *arXiv preprint arXiv:2003.04611*, 2020.

[21] A. Vakhrushev, A. Ryzhkov, M. Savchenko, D. Simakov, R. Damdinov, and A. Tuzhilin, "Lightautoml: Automl solution for a large financial services ecosystem," *CoRR*, vol. abs/2109.01528, 2021.

[22] C. Catlin, "Auto-ts: Automated time series forecasting." `https://github.com/WinedarkSea/Auto_TS`, 2023.

[23] N. O. Nikitin, P. Vychuzhanin, M. Sarafanov, I. S. Polonskaia, and A. V. Kalyuzhnaya, "Structural evolutionary learning for data-driven pipeline generation," in *Proceedings of the 2021 Genetic and Evolutionary Computation Conference Companion*, pp. 313–314, 2021.

[24] Salesforce Engineering, "Transmogrifai: Automated machine learning for structured data," *Salesforce Open Source*, 2018. `https://transmogrif.ai/`.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[26] Google Cloud, "Vertex ai documentation," 2024. `https://cloud.google.com/vertex-ai`.

[27] Microsoft, "Azure machine learning automated ml," 2024. `https://learn.microsoft.com/en-us/azure/machine-learning/concept-automated-ml`.

[28] Amazon Web Services, "Amazon sagemaker autopilot," 2024. https://docs.aws.amazon.com/sagemaker/latest/dg/autopilot.html.

[29] K. Batsios, S. Ougiaroglou, and C. Bratsas, "Autoknn: An automl web application for k-nearest neighbours classification," in *2025 16th International Conference on Information, Intelligence, Systems & Applications (IISA)*, pp. 1–5, 2025.

[30] M. Zografos and S. Ougiaroglou, "Simplifying decision tree classification through the autodtrees web application and service," in *Generative Intelligence and Intelligent Tutoring Systems* (A. Sifaleras and F. Lin, eds.), (Cham), pp. 162–173, Springer Nature Switzerland, 2024.

[31] K. Gratsos, S. Ougiaroglou, and D. Margaris, "kclusterhub: An automl-driven tool for effortless partition-based clustering over varied data types," *Future Internet*, vol. 15, no. 10, 2023.

[32] K. Malliaridis, S. Ougiaroglou, and D. A. Dervos, "Webapriori: A web application for association rules mining," in *International Conference on Intelligent Tutoring Systems*, 2020.

[33] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 4768–4777, Curran Associates Inc., 2017.

[34] C. Pautasso, O. Zimmermann, and F. Leymann, "Restful web services vs. "big" web services - making the right architectural decisions," pp. 805–814, 04 2008.

[35] J. Brooke *et al.*, "Sus-a quick and dirty usability scale," *Usability evaluation in industry*, vol. 189, no. 194, pp. 4–7, 1996.