ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ:
«ΔΙΑΧΕΙΡΙΣΗ ΠΛΗΡΟΦΟΡΙΩΝ ΣΕ ΒΙΒΛΙΟΘΗΚΕΣ, ΑΡΧΕΙΑ, ΜΟΥΣΕΙΑ»

ΤΜΗΜΑ ΑΡΧΕΙΟΝΟΜΙΑΣ, ΒΙΒΛΙΟΘΗΚΟΝΟΜΙΑΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΗΣΗΣ
ΣΧΟΛΗ ΔΙΟΙΚΗΤΙΚΩΝ, ΟΙΚΟΝΟΜΙΚΩΝ ΚΑΙ ΚΟΙΝΩΝΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

DEPARTMENT OF ARCHIVAL, LIBRARY AND INFORMATION STUDIES
SCHOOL OF MANAGEMENT, ECONOMICS AND SOCIAL SCIENCES

Διπλωματική Εργασία

Αυτοματοποιημένη θεματική κατηγοριοποίηση στο ενεργό αρχείο του ΠΑΔΑ

Βασίλειος Βαλλιάνος (ΑΜ: 226682002)

Επιβλέπων: Ιωάννης Τριανταφύλλου

Αθήνα, Νοέμβριος 2023

Επιτροπή Εξέτασης

1. Ιωάννης Τριανταφύλλου

Ioannis Triantafyllou

Digitally signed by Ioannis Triantafyllou
Date: 2023.11.13 16:39:44 +02'00'

2. Δημήτριος Κουής

Dimitrios Kouis

Digitally signed by Dimitrios Kouis
Date: 2023.11.13 17:12:34 +02'00'

3. Ιωάννης Στογιαννίδης

Ioannis Stogiannidis

Digitally signed by Ioannis Stogiannidis
Date: 2023.11.13 17:14:50 +02'00'

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Βασίλειος Βαλλιάνος του Γεωργίου, με αριθμό μητρώου 226682002, φοιτητής του Προγράμματος Μεταπτυχιακών Σπουδών «Διαχείριση Πληροφοριών σε Βιβλιοθήκες, Αρχεία και Μουσεία» του Τμήματος Αρχειονομίας, Βιβλιοθηκονομίας και Συστημάτων Πληροφόρησης της Σχολής διοικητικών, Οικονομικών και Κοινωνικών Επιστημών του Πανεπιστημίου Δυτικής Αττικής, δηλώνω ότι:

«Είμαι συγγραφέας αυτής της μεταπτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο Δηλών

Βασίλειος Βαλλιάνος

Ευχαριστίες – Αφιερώσεις

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή κ. Ιωάννη Τριανταφύλλου για την αμέριστη βοήθεια και καθοδήγηση του καθόλη τη διάρκεια της εκπόνησης της παρούσας εργασίας.

Επίσης, ευχαριστώ τη σύζυγο και τα παιδιά μου για τη συμπαράσταση και την υπομονή τους.

04/11/2023

Βασίλειος Βαλλιάνος

Περίληψη στα ελληνικά

Η εργασία αφορά στη διερεύνηση της εφαρμογής της ταξινόμησης των εγγράφων του Πανεπιστημίου Δυτικής Αττικής που σχετίζονται με το σύνολο των δραστηριοτήτων του. Για την δημιουργία των απαιτούμενων Datasets θα αντληθούν δεδομένα από το σύνολο των εγγράφων που έχουν αναρτηθεί από το Πανεπιστήμιο στην υπηρεσία της «Διαύγειας». Θα διερευνηθούν τα χαρακτηριστικά των αλγορίθμων και τεχνικών που χρησιμοποιούνται για την επεξεργασία φυσικής γλώσσας και την εξαγωγή σημαντικών χαρακτηριστικών καθώς και μέθοδοι ταξινόμησης μηχανικής και βαθιάς μάθησης, κάνοντας χρήση της γλώσσας προγραμματισμού Python και διαφόρων βιβλιοθηκών της, έτσι ώστε να προκύψει ένα εργαλείο αυτοματοποιημένης κατηγοριοποίησης των εγγράφων του Πανεπιστημίου.

Λέξεις Κλειδιά: μηχανική μάθηση, βαθιά μάθηση, κατηγοριοποίηση κειμένου, επεξεργασία φυσικής γλώσσας, devmax.df, chi square

Περίληψη στα αγγλικά

This dissertation is about the research of the application of the classification of all the documents relating to the activities of the University of Western Attica. To create the required Dataset, data will be extracted from all the documents that have been uploaded by the University on the "Diavgeia" portal. Features of algorithms and techniques used for natural language processing and feature extraction, as well as, machine learning and deep learning classification methods will be explored, making use of the Python programming language and its various libraries, so that an automated classification/categorization tool of the University's documents is created.

Keywords: machine learning, deep learning, text classification, natural language processing, devmax.df, chi square

Πίνακας Περιεχομένων

Πίνακας Σχημάτων

Πίνακας Πινάκων

Κεφάλαιο 1. Εισαγωγή

1.1 Πλαίσιο, σκοπός και στόχοι της διπλωματικής εργασίας

Η εργασία αφορά στη διερεύνηση της εφαρμογής της ταξινόμησης των εγγράφων του Πανεπιστημίου Δυτικής Αττικής που σχετίζονται με το σύνολο των δραστηριοτήτων του. Για την δημιουργία του απαιτούμενου Dataset θα αντληθούν δεδομένα από το σύνολο των εγγράφων που έχουν αναρτηθεί από το Πανεπιστήμιο στην υπηρεσία της «Διαύγειας». Θα διερευνηθούν τα χαρακτηριστικά των αλγορίθμων και τεχνικών που χρησιμοποιούνται για την επεξεργασία φυσικής γλώσσας και την εξαγωγή σημαντικών χαρακτηριστικών καθώς και μέθοδοι ταξινόμησης μηχανικής και βαθιάς μάθησης, κάνοντας χρήση της γλώσσας προγραμματισμού Python και διαφόρων βιβλιοθηκών της, έτσι ώστε να προκύψει ένα εργαλείο αυτοματοποιημένης ταξινόμησης/κατηγοριοποίησης των εγγράφων του Πανεπιστημίου.

Οι κύριοι στόχοι της εργασίας είναι:

α) να διερευνήσει την εφαρμογή της ταξινόμησης εγγράφων στο σύνολο των εγγράφων που αφορούν στο σύνολο των δραστηριοτήτων του Πανεπιστημίου Δυτικής Αττικής και τη παροχή ενός εργαλείου αυτοματοποιημένης ταξινόμησης/κατηγοριοποίησης τους.

β) να διερευνήσει τα χαρακτηριστικά των αλγορίθμων και τεχνικών που χρησιμοποιούνται για την επεξεργασία φυσικής γλώσσας,

γ) να διερευνήσει τα χαρακτηριστικά των μεθόδων μηχανικής μάθησης που χρησιμοποιούνται για την ταξινόμηση κειμένου,

δ) η παροχή ενός εργαλείου αυτοματοποιημένης ταξινόμησης των εγγράφων που αφορούν τις διαφορετικές δραστηριότητες του Πανεπιστημίου.

1.2 Οργάνωση Κεφαλαίων

Η παρούσα εργασία παρουσιάζει το σύνολο των ενεργειών που πραγματοποιήθηκαν για την ταξινόμηση/κατηγοριοποίηση του ενεργού αρχείου του ΠΑΔΑ στη «Διαύγεια», χρησιμοποιώντας αλγορίθμους στη γλώσσα προγραμματισμού Python και τη βιβλιοθήκη Scikit-learn. Επίσης συγκρίνει και αναλύει τα αποτελέσματά τους.

Πιο συγκεκριμένα παρουσιάζονται τα ακόλουθα κεφάλαια:

Στο Κεφάλαιο 2 «Θεωρητικό μέρος – Σχετικές προσπάθειες» παρουσιάζονται: α) μια ανασκόπηση της βασικής βιβλιογραφίας για την κατηγοριοποίηση κειμένου και β) σχετικές προσπάθειες στην κατηγοριοποίηση κειμένου.

Στο Κεφάλαιο 3 «Μεθοδολογία» παρουσιάζονται: τα βασικά βήματα που ακολουθήθηκαν και τις τεχνικές που χρησιμοποιήθηκαν κατά τη διαδικασία κατηγοριοποίησης (προεπεξεργασία, διανυσματοποίηση).

Στο Κεφάλαιο 4 «Αποτελέσματα» παρουσιάζονται: η ανάλυση των αποτελεσμάτων της θεματικής κατηγοριοποίησης του κειμένου των αρχείων του ΠΑΔΑ στη «Διαύγεια» χρησιμοποιώντας τα προαναφερθέντα βήματα και τεχνικές.

Τέλος, Κεφάλαιο 5 «Συμπεράσματα»: συνοψίζει τα βασικά βήματα και τα συμπεράσματα της εργασίας.

Επίσης, η εργασία περιλαμβάνει παραρτήματα με τον Python κώδικα που χρησιμοποιήθηκε σε κάθε βήμα.

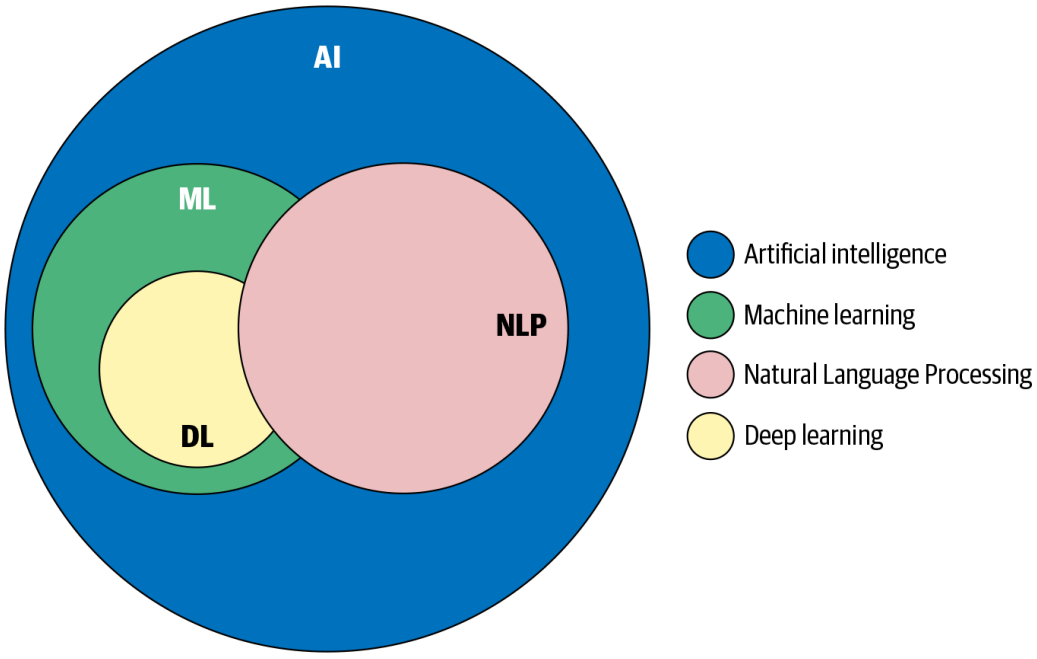
Κεφάλαιο 2. Θεωρητικό μέρος – Σχετικές προσπάθειες

Στο παρόν κεφάλαιο θα δοθεί μια συνοπτική αναφορά σε έννοιες της μηχανικής μάθησης (Machine Learning - ML) και βαθιάς μάθησης (Deep Learning - DL) καθώς και της επεξεργασίας φυσικής γλώσσας (Natural Language Processing – NLP). Επίσης θα γίνει μια σύντομη αναφορά σε σχετικές, με την παρούσα εργασία, προσπάθειες και ερευνητικές εργασίες.

2.1 Θεωρητικό μέρος

2.1.1 Τεχνητή νοημοσύνη, μηχανική μάθηση και βαθιά μάθηση

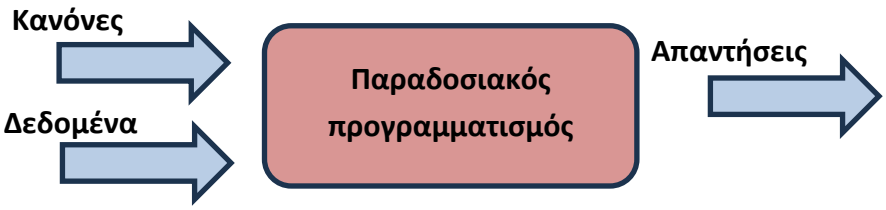
Ο όρος «τεχνητή νοημοσύνη» αναφέρεται στον τομέα της επιστήμης υπολογιστών που ασχολείται με την ανάπτυξη υπολογιστικών συστημάτων ικανών να εκδηλώσουν ικανότητες που συνδέονται με την ανθρώπινη νοημοσύνη (Vajjala et al., 2020).



Εικόνα 1: Η σχέση μεταξύ AI, ML, DL και NLP ((Vajjala et al., 2020, p. 15)

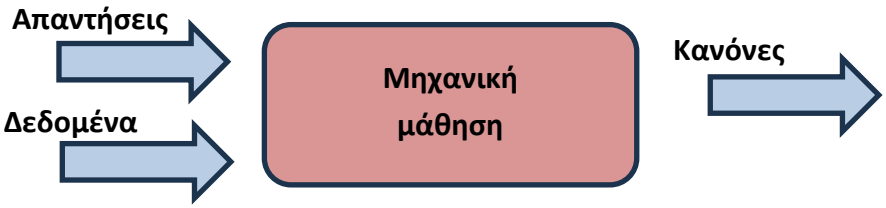
Μια προσέγγιση της τεχνητής νοημοσύνης αποτελεί η Μηχανική Μάθηση. Σκοπός των αλγόριθμων μηχανικής μάθησης είναι να επιτρέψουν στα συστήματα να μάθουν από τα δεδομένα και να βελτιώσουν την απόδοσή τους με την εμπειρία (Mitchell, 1997) χωρίς να υπάρχει ανάγκη να προγραμματιστούν εκ νέου (Γεωργούλη, 2015).

Στον παραδοσιακό προγραμματισμό, οι απαντήσεις στο προς επίλυση πρόβλημα προέρχονται μετά από την εφαρμογή κανόνων, οι οποίοι εκφράζονται σε μια γλώσσα προγραμματισμού, πάνω στα δεδομένα (Εικόνα 2).



Εικόνα 2: Παραδοσιακός προγραμματισμός

Αντίθετα με τον παραδοσιακό προγραμματισμό, στη Μηχανική Μάθηση το σύστημα δέχεται ένα σύνολο δεδομένων στο οποίο περιλαμβάνονται και είναι γνωστές οι απαντήσεις και εξάγει τους κανόνες ώστε να προβλέψει αυτόματα τις απαντήσεις σε δεδομένα που θα του δοθούν μελλοντικά (Εικόνα 3).



Εικόνα 3: Μηχανική Μάθηση

Η βαθιά μάθηση (Deep Learning) αποτελεί υποκατηγορία της μηχανικής μάθησης. Οι μέθοδοι βαθιάς μάθησης είναι μέθοδοι εκμάθησης αναπαράστασης με πολλαπλά επίπεδα αναπαράστασης, που λαμβάνονται με τη σύνθεση απλών αλλά μη γραμμικών ενοτήτων που καθεμία μετατρέπει την αναπαράσταση σε ένα επίπεδο (ξεκινώντας από την ακατέργαστη είσοδο) σε μια αναπαράσταση σε υψηλότερο, ελαφρώς πιο αφηρημένο επίπεδο. Σε προβλήματα ταξινόμησης, κάθε επίπεδο αναπαράστασης ενισχύει τις πτυχές της εισόδου που είναι σημαντικές για τη διάκριση μεταξύ των κατηγοριών ταξινόμησης και καταστέλλει όσες δεν είναι (LeCun et al., 2015).

2.1.1.1 Κατηγορίες Μηχανικής μάθησης

Τα συστήματα Μηχανικής Μάθησης μπορεί να διακριθούν σε κατηγορίες, ανάλογα με την έκταση αλλά και τον τρόπο επίβλεψης που δέχονται κατά την διάρκεια της εκπαίδευσης των μοντέλων τους (Geron, 2019, p. 8).

Οι τέσσερις κύριες κατηγορίες είναι: της επιβλεπόμενης μάθησης (supervised learning), της μάθησης χωρίς επίβλεψη (unsupervised learning), της ημι-επιβλεπόμενης μάθησης (semi-supervised learning) και της ενισχυτικής μάθησης (reinforcement learning).

Supervised learning

Η πιο διαδεδομένη κατηγορία μηχανικής μάθησης είναι η επιβλεπόμενη μάθηση κατά την οποία ο αλγόριθμος τροφοδοτείται με δεδομένα που περιέχουν την επιθυμητή λύση, το μοντέλο δημιουργεί μια συνάρτηση που συνδέει τα δεδομένα εισόδου με τις λύσεις, έτσι ώστε να

προβλέψει σωστά την κατάλληλη λύση σε άγνωστα δεδομένα εισόδου που θα του δοθούν μελλοντικά (Chollet, 2020).

Η επιβλεπόμενη μάθηση χρησιμοποιείται κατά κύριο λόγο σε προβλήματα ταξινόμησης (classification).

2.1.1.2 Κατηγορίες προβλημάτων ταξινόμησης

- Binary classification: διαδικασία κατά την οποία κάθε δείγμα εισόδου θα μπορούσε να ταξινομηθεί σε μια από δυο διακριτές κλάσεις.
- Multiclass classification: διαδικασία στην οποία κάθε δείγμα εισόδου θα μπορούσε να ταξινομηθεί σε μια κλάση ανάμεσα σε περισσότερες από δυο κλάσεις π.χ. ταξινόμηση χειρόγραφων χαρακτήρων.
- Multilabel classification: διαδικασία κατά την οποία σε κάθε δείγμα εισόδου μπορούν να ανήκει σε περισσότερες από μια κλάσεις π.χ. σε μια εικόνα που περιέχει ένα αυτοκίνητο και ένα ποδήλατο θα χαρακτηριστεί ότι ανήκει στις κλάσεις «αυτοκίνητο» και «ποδήλατο».
- Multi-output classification: αποτελεί μια γενίκευση της multilabel classification όπου κάθε label μπορεί να είναι multiclass.

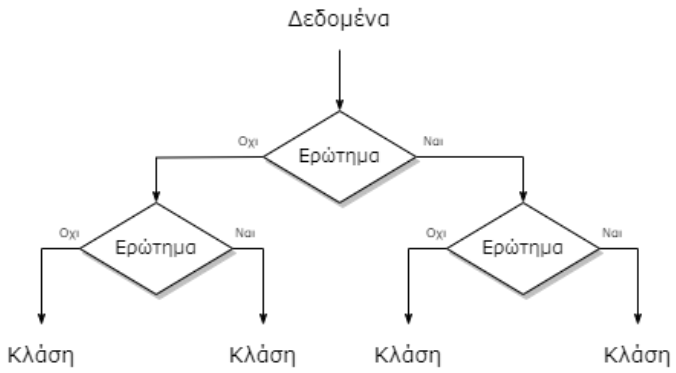
2.1.1.3 One-vs-the-rest

Πρόκειται για την πιο συνηθισμένη στρατηγική που χρησιμοποιείται τόσο σε multiclass όσο και multilabel προβλήματα. Η τεχνική που χρησιμοποιείται είναι αυτή της εκπαίδευσης ενός δυαδικού ταξινομητή ανά κλάση. Σε κάθε δυαδικό ταξινομητή που εκπαιδεύεται, τα δείγματα που ανήκουν στη κλάση αντιμετωπίζονται ως θετικά ενώ τα δείγματα των υπόλοιπων κλάσεων ως αρνητικά. Η στρατηγική διακρίνεται για την υπολογιστική της αποτελεσματικότητα (απαιτούνται τόσοι ταξινομητές όσες και οι κλάσεις).

2.1.1.4 Αλγόριθμοι Μηχανικής Μάθησης

2.1.1.4.1 Decision Tree

Τα Δέντρα Απόφασης (Decision Tree) είναι αλγόριθμοι ταξινόμησης οι οποίοι μέσω μιας σειράς ερωτημάτων αποδομούν τα δεδομένα εισόδου και καταλήγουν σε κάποιο αποτέλεσμα. Τα ερωτήματα αυτά μπορεί να αφορούν είτε κατηγορικές είτε αριθμητικές μεταβλητές (Raschka & Mirjalili, 2019, p. 91).

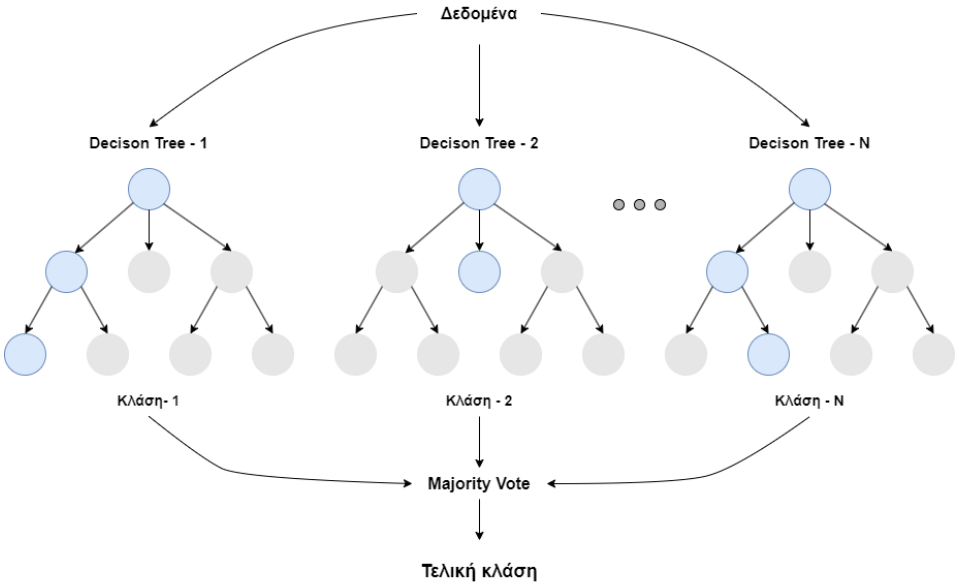


Εικόνα 4: Παράδειγμα Decision Tree

Η γραφική αναπαράσταση ενός αλγορίθμου Δέντρου Απόφασης έχει τη μορφή ανάποδου δέντρου (Εικόνα 4) στο οποίο για να καταλήξουμε από τα δεδομένα εισόδου (ρίζα του δέντρου) σε κάποιο αποτέλεσμα (φύλλα του δέντρου), που για ένα πρόβλημα ταξινόμησης είναι η κλάση κάθε δεδομένου στην είσοδο, ακολουθούμε μια μοναδική διαδρομή. Ο αριθμός των ενδιάμεσων ερωτημάτων (κόμβων) μεταξύ εισόδου και εξόδου ονομάζεται βάθος (depth) και αποτελεί ένα από τα χαρακτηριστικά ενός Δέντρου Απόφασης.

2.1.1.4.2 Random Forest

Ο αλγόριθμος Random Forest ανήκει στην κατηγορία των μεθόδων *ensemble*, οι οποίες βασίζονται στο συνδυασμό πολλαπλών μοντέλων και προτιμώνται τόσο για την απόδοση τους σε εφαρμογές ταξινόμησης όσο και για την ανοχή τους σε υπερεκπαίδευση (*overfitting)* (Raschka & Mirjalili, 2019, p. 100). Επομένως, ένα Random Forest μπορεί να θεωρηθεί ως ένα σύνολο από Decision Trees.

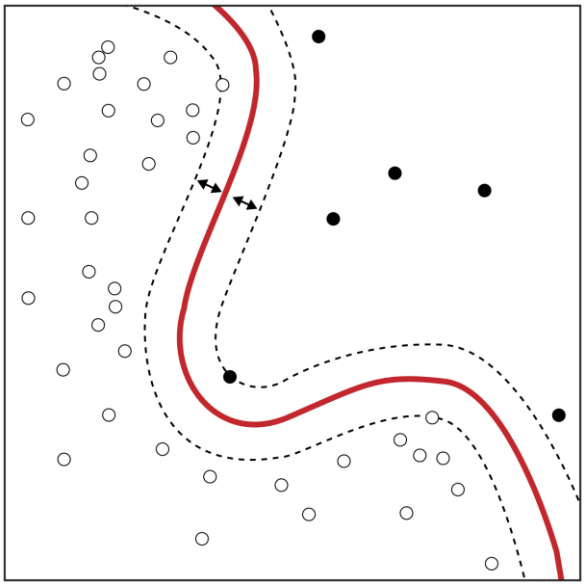


Εικόνα 5: Παράδειγμα Random Forest

Μια παράμετρος ενός Random Forest είναι ο αριθμός των δέντρων (Decision Trees) που αποτελείται. Κάθε ένα από τα δέντρα δέχεται στην είσοδο του τα ίδια δεδομένα με τα υπόλοιπα και καταλήγει σε μια προβλεπόμενη κλάση για τα δεδομένα αυτά. Η τελική προβλεπόμενη κλάση προκύπτει βάσει της πλειοψηφούσας κλάσης στο σύνολο των προβλεπόμενων κλάσεων (Εικόνα 5).

2.1.1.4.3 SVM

Ο αλγόριθμος SVM (support vector machine) είναι ένας δημοφιλής αλγόριθμος, ο οποίος μεγιστοποιεί το περιθώριο του ορίου απόφασης μεταξύ δειγμάτων που ανήκουν σε διαφορετικές κλάσεις (Εικόνα 6). Είναι ανθεκτικός στη διακύμανση και τον θόρυβο στα δεδομένα. Στα αρνητικά του σημεία περιλαμβάνεται η υψηλή υπολογιστική πολυπλοκότητα καθώς μπορεί να απαιτεί πολλούς υπολογιστικούς πόρους και χρόνο εκπαίδευσης όταν υπάρχουν μεγάλοι όγκοι δεδομένων εκπαίδευσης (Vajjala et al., 2020, p. 21).



Εικόνα 6: Απεικόνιση του περιθωρίου ορίου απόφασης ενός SVM (Vajjala et al., 2020, p. 21)

2.1.1.5 Artificial Neural Networks – Deep Learning

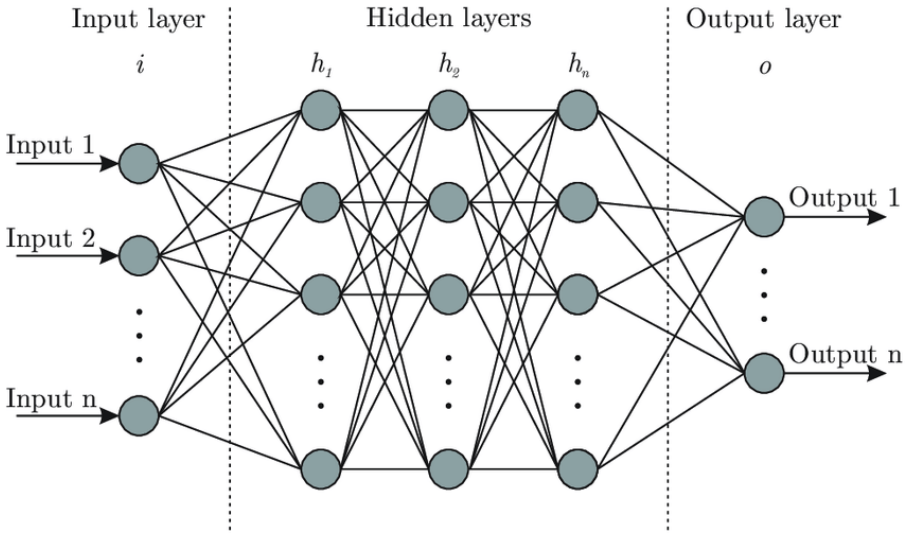
Τα νευρωνικά δίκτυα, εμπνευσμένα από τη δομή των νευρώνων στον ανθρώπινο εγκέφαλο, αποτελούνται από ένα πλήθος κόμβων, που επίσης καλούνται νευρώνες, οι οποίοι είναι τοποθετημένοι σε διαφορετικά επίπεδα (layers) και είναι συνδεδεμένοι μεταξύ τους σε όλη τη διαδρομή από την είσοδο των δεδομένων, μέχρι την έξοδο της πληροφορίας από αυτά. Τα επίπεδα που βρίσκονται ενδιάμεσα των επιπέδων εισόδου και εξόδου ονομάζονται κρυφά

επίπεδα (hidden layers), ενώ το συνολικό πλήθος των επιπέδων ονομάζεται βάθος (depth) του δικτύου. Δίκτυα με μικρό πλήθος επιπέδων ονομάζονται ρηχά (shallow).

Το πλήθος των κόμβων (νευρώνων) που βρίσκεται σε κάθε επίπεδο εκφράζει την διάσταση (width) του επιπέδου (Goodfellow et al., 2016). Οι συνδέσεις μεταξύ των κόμβων που βρίσκονται σε διαφορετικά επίπεδα και κατ' επέκταση οι συνδέσεις μεταξύ επιπέδων, έχουν διαφορετική βαρύτητα, η οποία μπορεί να επανακαθοριστεί κατά την διαδικασία εκπαίδευσης του δικτύου.

Ένα ακόμα σημαντικό στοιχείο της δομής ενός νευρωνικού δικτύου είναι οι συναρτήσεις ενεργοποίησης (activation functions) που θα παρουσιαστούν στη συνέχεια.

Ένα παράδειγμα απεικόνισης των κόμβων και των επιπέδων ενός νευρωνικού δικτύου παρουσιάζεται στην παρακάτω εικόνα:



Εικόνα 7: Νευρωνικό δίκτυο (Sharma et al., 2020)

Μπορούμε να διακρίνουμε δυο βασικές κατηγορίες νευρωνικών δικτύων (Singh & Chauhan, 2009, p. 38):

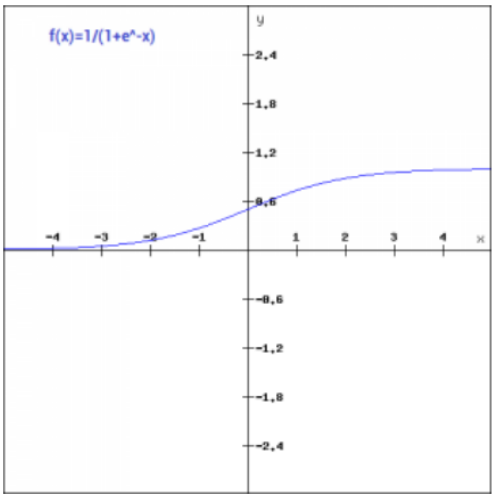
- **Feedforward Networks:** πρόκειται για δίκτυα στα οποία η ροή διεργασιών γίνεται προς μια κατεύθυνση, από τους κόμβους εισόδου προς τους κόμβους εξόδου μέσω κάποιου πλήθους (από 0 μέχρι n) ενδιάμεσων κρυφών κόμβων, χωρίς να υπάρχει ανατροφοδότηση μεταξύ των κόμβων.
- **Recurrent Networks:** δίκτυα στα οποία υπάρχει ανατροφοδότηση μεταξύ κόμβων, δίνοντας την δυνατότητα αξιοποίησης δεδομένων που προέρχονται από μεταγενέστερα στάδια ώστε να βελτιωθεί η διαδικασία εκπαίδευσης προγενέστερων σταδίων.

Οι συναρτήσεις ενεργοποίησης (activation functions) παίζουν σημαντικό ρόλο στα νευρωνικά δίκτυα. Σκοπός αυτών των προκαθορισμένων συναρτήσεων είναι ο μετασχηματισμός των

δεδομένων που δέχονται στην είσοδο τους (από ένα προηγούμενο επίπεδο) σε μια μορφή κατάλληλη για την περαιτέρω επεξεργασία τους από το επόμενο επίπεδο.

Οι συναρτήσεις που χρησιμοποιήθηκαν σε αυτή την εργασία παρουσιάζονται στη συνέχεια.

Sigmoid

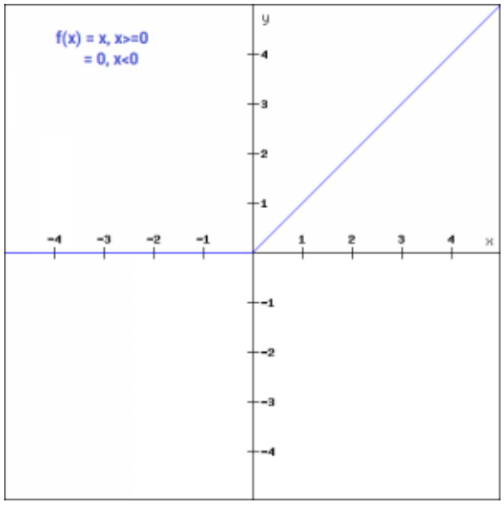


Εικόνα 8: Γραφική παράσταση της συνάρτησης sigmoid (Sharma et al., 2020)

Είναι παλαιότερη συνάρτηση ενεργοποίησης που χρησιμοποιούνταν σε πρώιμα νευρωνικά δίκτυα. Εξακολουθεί να χρησιμοποιείται σε ορισμένες περιπτώσεις, όπως σε προβλήματα δυαδικής ταξινόμησης ή στο τελευταίο επίπεδο για την εξαγωγή πιθανοτήτων. Είναι μη γραμμική, με σημείο συμμετρίας διάφορο του μηδενός, μετασχηματίζοντας τις τιμές εισόδου στην κλίμακα από 0 μέχρι 1. Ορίζεται από τον ακόλουθο τύπο:

$$f(x) = \frac{1}{1 + e^{-x}}$$

ReLU



Εικόνα 9: Γραφική παράσταση της συνάρτησης ReLU (Sharma et al., 2020)

Η ReLU (rectified liner unit) είναι μια πιο σύγχρονη και δημοφιλής συνάρτηση ενεργοποίησης που χρησιμοποιείται σε νευρωνικά δίκτυα. Είναι μη γραμμική και ορίζεται από τον ακόλουθο τύπο:

$$f(x) = max(0, x)$$

Η συνάρτηση ReLU είναι υπολογιστικά αποδοτική και εύκολη στην εφαρμογή, καθώς απλά επιλέγει την μέγιστη τιμή μεταξύ του 0 και της εισόδου της. Αυτό συνεπάγεται γρήγορους υπολογισμούς κατά την εκπαίδευση, καθιστώντας την ως πρώτη επιλογή σε πολλές αρχιτεκτονικές νευρωνικών δικτύων. Επίσης, η αυξημένη απόδοση της (αποτελεσματικότητα, οικονομία υπολογιστικών πόρων) συγκριτικά με άλλες συναρτήσεις έγκειται στο γεγονός ότι η ReLU δεν ενεργοποιεί όλους τους νευρώνες ταυτόχρονα (Sharma et al., 2020).

Dropout Layer

Το συγκεκριμένο επίπεδο αποτελεί μια τεχνική που χρησιμοποιείται συνήθως σε νευρωνικά δίκτυα, ιδιαίτερα σε μοντέλα βαθιάς μάθησης, για την αποφυγή υπερπροσαρμογής (overfitting). Η υπερπροσαρμογή συμβαίνει όταν ένα νευρωνικό δίκτυο μαθαίνει να απομνημονεύει τα δεδομένα εκπαίδευσης αντί να γενικεύει από αυτά, οδηγώντας σε χαμηλή απόδοση σε άγνωστα δεδομένα.

Το dropout layer λειτουργεί με τυχαίο μηδενισμό ενός συγκεκριμένου κλάσματος των νευρώνων ενός επίπεδου σε κάθε βήμα κατά τη διάρκεια της εκπαίδευσης. Αυτό το κλάσμα καθορίζεται από μια υπερπαράμετρο που ονομάζεται αναλογία απόρριψης (dropout rate), η οποία παίρνει τιμές μεταξύ 0 και 1. Μια συνήθης τιμή της είναι το 0,5, που σημαίνει ότι, κατά μέσο όρο, οι μισοί νευρώνες μηδενίζονται σε κάθε χρήση του επιπέδου.

2.1.2 Επεξεργασία φυσικής γλώσσας (NLP)

Η επεξεργασία φυσικής γλώσσα αποτελεί ένα πεδίο της επιστήμης των υπολογιστών και της τεχνητής νοημοσύνης, που έχει βάσεις στην υπολογιστική γλωσσολογία. Ασχολείται με τη μελέτη, σχεδιασμό και κατασκευή συστημάτων που μπορούν να κατανοήσουν τη φυσική ανθρώπινη γλώσσα ώστε να παρέχουν στη συνέχεια χρήσιμα αποτελέσματα και είναι ικανά να επιτρέπουν την αλληλεπίδραση μεταξύ μηχανών και φυσικών γλωσσών (Sarkar, 2016).

Η επεξεργασία φυσικής γλώσσας έχει ποικίλες εφαρμογές σε πλήθος διαφορετικών τομέων (Γεωργούλη, 2015) όπως:

- **Ανάκτηση δεδομένων** (information retrieval)
- **Ανάκτηση εγγράφων** (document retrieval)
- **Εξαγωγή πληροφορίας** (information extraction)
- **Εξόρυξη δεδομένων** (data mining)
- **Αναγνώριση μερών του λόγου** (part-of-speech tagging)

- **Κατηγοριοποίηση κειμένων** (text categorization)

2.1.2.1 Ταξινόμηση / Κατηγοριοποίηση Κειμένου

Η ταξινόμηση κειμένου αποτελεί βασικό πεδίο της επεξεργασίας φυσικής γλώσσας. Έχοντας οποιοδήποτε κείμενο ως πρωταρχικό στοιχείο εισόδου, αυτό θα πρέπει να επεξεργαστεί με κατάλληλο τρόπο πριν οδηγηθεί σε κάποιο μοντέλο μηχανικής μάθησης.

Βασικά βήματα σε αυτή τη διαδικασία είναι τα ακόλουθα:

- **Text Tokenization:** Διάσπαση του κειμένου σε μικρότερα τμήματα. Αυτά μπορεί να είναι είτε μια πρόταση είτε μια λέξη. Αν η διάσπαση γίνει σε επίπεδο λέξης, η διαδικασία ονομάζεται **word tokenization**. Χωρίζοντας ένα μεγάλο κείμενο σε λέξεις, κάθε μια λέξη επεξεργάζεται ξεχωριστά.
- **Αφαίρεση των σημείων στίξης:** τα σημεία στίξης δεν παρέχουν κάποια πληροφορία, ούτε έχουν κάποια αξία.
- **Αφαίρεση των stop words:** αφαίρεση των πολύ συχνά χρησιμοποιούμενων λέξεων όπως άρθρα, σύνδεσμοι, επιρρήματα κλπ. Δεν παρέχουν κάποια πληροφορία, ούτε έχουν κάποια αξία.
- **Μετατροπή του κειμένου σε κεφαλαία:** συμβάλει στην ομογενοποίηση του κειμένου έτσι ώστε να αποφευχθούν διπλές εγγραφές στα επόμενα στάδια της διαδικασίας όπως αυτό της δημιουργίας του λεξικού.
- **Stemming:** διαδικασία κατά την οποία ελέγχεται κάθε λέξη του κειμένου και με κατάλληλες μεθόδους (π.χ. αφαίρεση της κατάληξη της) ανακαλύπτεται η βασική μορφή της (ρίζα).

2.1.2.2 Μετατροπή κειμένου σε διανύσματα (Bag-of-words)

Ένας κλασσικός τρόπος αναπαράστασης κείμενου ώστε να μπορεί να επεξεργαστεί από τους αλγόριθμους μηχανικής και βαθιάς μάθησης είναι να το μετατρέψουμε σε αριθμούς. Μια απλή και δημοφιλής μέθοδος είναι η Bag-of-Words η οποία αναπαριστά το κείμενο περιγράφοντας την εμφάνιση των λέξεων μέσα σε αυτό. Βασικά στάδια σε αυτή τη διαδικασία είναι:

- η δημιουργία ενός λεξικού που αποθηκεύει τη βασική μορφή κάθε λέξης ενώ οι υπόλοιπες μορφές μπορούν να προκύψουν με κανόνες μορφολογικής ανάλυσης (Γεωργούλη, 2015)
- ο υπολογισμός της σημαντικότητας κάθε λέξης μέσα στο κείμενο και η επιλογή των πιο σημαντικών από αυτές, εφαρμόζοντας κατάλληλες τεχνικές επιλογής χαρακτηριστικών (feature selection) στο σύνολο (corpus) των προς επεξεργασία κειμένων.

Η ονομασία Bag-of-Words (τσάντα λέξεων) προκύπτει από το γεγονός ότι δεν διατηρείται κάποια πληροφορία σχετικά με την σειρά ή την δομή των λέξεων μέσα στο κείμενο αλλά με το αν υπάρχουν γνωστές λέξεις (του λεξικού) μέσα στο κείμενο. Επομένως, σε αυτή την προσέγγιση, εξετάζεται το ιστόγραμμα των λέξεων μέσα στο κείμενο, δηλ. θεωρείται το πλήθος εμφάνισης κάθε λέξης ως χαρακτηριστικό (feature). (Goldberg, 2017, p. 69)

	Λέξη1	Λέξη2	Λέξη3	Λέξη4	Λέξη5
Έγγραφο1	1	1	0	0	0
Έγγραφο2	0	1	0	1	0
Έγγραφο3	0	0	1	0	0
Έγγραφο4	1	1	0	1	1
Έγγραφο5	1	0	1	0	1

Εικόνα 10: Παράδειγμα μετατροπής κειμένου σε διανύσματα

2.1.2.3 Τεχνικές επιλογής χαρακτηριστικών (feature selection)

2.1.2.3.1 CHI SQUARE (x^2)

Η συγκεκριμένη είναι μια δημοφιλής μετρική που χρησιμοποιείται για την επιλογή χαρακτηριστικών σε προβλήματα ταξινόμησης κειμένου. Υπολογίζεται βάσει της ακόλουθης εξίσωσης (Triantafyllou et al., 2020, p. 8):

$$x^2 = \sum_{i=1}^{c} \frac{(O_i - E_i)^2}{E_i}$$

2.1.2.3.2 DEVMAX.DF

Μια νέα μέθοδος που χρησιμοποιείται σε προβλήματα ταξινόμησης κειμένου είναι η DEVMAX.DF (Triantafyllou et al., 2020, p. 8). Η μέθοδος προωθεί λέξεις-όρους που έχουν τη μέγιστη απόκλιση στις εμφανίσεις ή τις ελάχιστες εμφανίσεις σε άλλες κλάσεις από τη βασική κλάση (max).

$$Devmax.DF = \frac{\sqrt{\frac{1}{c-1}\sum_{i=1}^{c}(\frac{DF_i}{D_i} - max)^2}}{max} * \log(DF)$$

Η τιμή του max προκύπτει από τη παρακάτω εξίσωση:

$$max = maximum_{i=1}^{c} \frac{DF_i}{D_i}$$

Το c είναι το σύνολο των κλάσεων, DF_i είναι το σύνολο των εγγράφων (δειγμάτων) που περιέχουν τον όρο F στην κλάση i και D το σύνολο των εγγράφων που ανήκουν στην κλάση i.

2.1.3 Μετρικές και Τεχνικές Αξιολόγησης

Οι μετρικές αξιολόγησης αποτελούν ένα σημαντικό εργαλείο στην τελική αποτίμηση της απόδοσης ενός μοντέλου μηχανικής μάθησης, αφενός γιατί επιτρέπουν να εκτιμήσουμε την ακρίβεια που έχει αυτό ως προς την ζητούμενη πρόβλεψη, αφετέρου γιατί επιτρέπουν τη σύγκριση μεταξύ μοντέλων και την επιλογή του βέλτιστου.

Οι παρακάτω ποσότητες αποτελούν βασικά στοιχεία στον υπολογισμό των μετρικών που θα παρουσιαστούν στη συνέχεια:

1. True Positive (TP): Το μοντέλο προβλέπει **ορθώς** ότι ένα αντικείμενο **ανήκει** σε μια κλάση-κατηγορία
2. True Negative (TN): Το μοντέλο προβλέπει **ορθώς** ότι ένα αντικείμενο **δεν ανήκει** σε μια κατηγορία
3. False Positive (FP): Το μοντέλο προβλέπει **λανθασμένα** ότι ένα αντικείμενο **ανήκει** σε μια κατηγορία
4. False Negative (FN): Το μοντέλο προβλέπει **λανθασμένα** ότι ένα αντικείμενο **δεν ανήκει** σε μια κατηγορία

Confusion Matrix

Οι παραπάνω ποσότητες μπορούν να οπτικοποιηθούν στο λεγόμενο πίνακα σύγχυσης (confusion matrix).

	Προβλεφθέντα 0	Προβλεφθέντα 1
Πραγματικά 0	TN	FP
Πραγματικά 1	FN	TP

Εικόνα 11: Confusion matrix

Accuracy

Η συνολική ακρίβεια ορίζεται ως ο λόγος των ορθών προβλέψεων προς το σύνολο των προβλέψεων (Lipton et al., 2014):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision

Η ακρίβεια είναι μια μετρική που αποτελεί μια εκτίμηση της ορθότητας των προβλέψεων ενός μοντέλου (Lipton et al,2014). Ορίζεται ως ο λόγος των ορθώς θετικών προβλέψεων προς το σύνολο των θετικών προβλέψεων:

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

Recall

Η ανάκληση είναι μια μετρική που αποτελεί μια εκτίμηση της πληρότητας (completeness) των προβλέψεων ενός μοντέλου(Lipton et al., 2014). Ορίζεται ως ο λόγος των ορθώς θετικών προβλέψεων προς το σύνολο των σωστών προβλέψεων που υπάρχουν:

$$Recall = \frac{TP}{TP + FN}$$

F_1 score

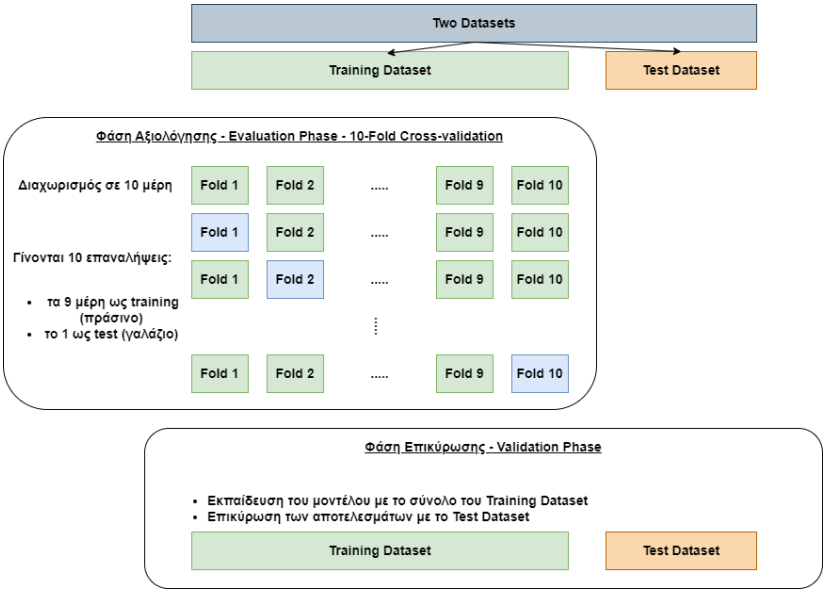
Η μετρική F-measure υπολογίζεται ως ο αρμονικός μέσος όρος (harmonic mean) των μετρικών Precision και Recall (Lipton et al., 2014) και ορίζεται από την εξίσωση:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Στην περίπτωση που αντιμετωπίζουμε ένα multilabel πρόβλημα ταξινόμησης μπορούμε να υπολογίσουμε τον μέσο όρο όλων των ετικετών, αν αποδώσουμε διαφορετική βαρύτητα σε κάθε ετικέτα σύμφωνα με το πλήθος εμφανίσεων της στο σύνολο των δεδομένων (Geron, 2019, p. 109).

Αξιολόγηση - Evaluation - 10-fold Cross-validation (Διασταυρωμένη επικύρωση)

Η πιο διαδεδομένη τεχνική για την εκτίμηση της απόδοσης ενός μοντέλου μηχανικής μάθησης είναι η 10-fold Cross-validation. Τα αρχικά δεδομένα (Initial Dataset) χωρίζονται σε δυο μέρη, στο Training Dataset που θα παίξει το ρόλο των δεδομένων εκπαίδευσης και στο Test Dataset που θα χρησιμοποιηθεί για την τελική επικύρωση (validation) των αποδόσεων των μοντέλων.



Εικόνα 12: Φάσεις Αξιολόγησης – Επικύρωσης (Evaluation – Validation)

Στη συνέχεια το Training Dataset χωρίζεται σε 10 μέρη με τέτοιο τρόπο ώστε κάθε κλάση να εμφανίζεται σε ίδιο ποσοστό (stratification) όπως και στο συνολικό Dataset.

Η διαδικασία εκπαίδευσης του μοντέλου εκτελείται 10 φορές. Σε κάθε μια, ένα εκ των δέκα μερών, διαφορετικό σε κάθε επανάληψη, έχει το ρόλο δεδομένων ελέγχου (Test Dataset) ενώ τα εναπομείναντα εννέα χρησιμοποιούνται ως δεδομένα εκπαίδευσης (Training Dataset) (Εικόνα 12: Φάσεις Αξιολόγησης – Επικύρωσης (Evaluation – Validation). Η διάσπαση του Dataset σε 10 μέρη έχει διαπιστωθεί ότι είναι η βέλτιστη επιλογή (Witten et al., 2017, p. 168).

Στην περίπτωση ενός multilabel προβλήματος μια μέθοδος που προτείνεται είναι της Iterative Stratification (Sechidis et al., 2011; Szymański & Kajdanowicz, 2017) η οποία δίνει λύση σε προβλήματα που μπορεί να προκύψουν κατά τον υπολογισμό κάποιων μετρικών όταν στον τυχαίο διαχωρισμό του Training Dataset κατά τη διαδικασία του 10-fold cross-validation παρουσιαστεί έλλειψη δειγμάτων μιας σπάνιας κλάσης σε κάποιο από τα υποσύνολα δεδομένων.

Επικύρωση - Validation: Αρχικά, το μοντέλο εκπαιδεύεται κάνοντας χρήση του συνόλου του Training Dataset. Στη συνέχεια, ελέγχεται ως προς την τελική του απόδοση στα άγνωστα δεδομένα εισόδου του Test Dataset (Εικόνα 12: Φάσεις Αξιολόγησης – Επικύρωσης (Evaluation – Validation).

2.2 Σχετικές προσπάθειες

Η αυτοματοποιημένη ταξινόμηση κειμένου αποτελεί ένα ταχέως αναπτυσσόμενο και με μεγάλο ορίζοντα ανάπτυξης πεδίο έρευνας λόγω της διαρκώς αυξανόμενης παραγωγής πολύπλοκων εγγραφών σε ψηφιακή μορφή.

Τα πεδία εφαρμογής πολλά και ποικίλα, σχετικές ερευνητικές εργασίες που έχουν γίνει αφορούν νομικά κείμενα (H. Chen et al., 2022), ιατρικά δεδομένα (Amin-Nejad et al., 2020), την βιοϊατρική (Kesiku et al., 2022), τις κοινωνικές επιστήμες (P.-F. Chen et al., 2021; Nobles et al., 2018), την επιχειρηματική στρατηγική (Triantafyllou et al., 2020), την βιβλιοθηκονομία (Vorgia et al., 2017) και άλλα.

Οι τρόποι που οι ερευνητές προσεγγίζουν το αντικείμενο ποικίλουν (Mirończuk & Protasiewicz, 2018) καθώς χρησιμοποιούν διάφορες τεχνικές προεπεξεργασίας του κειμένου και έχουν ως πυρήνα τους τόσο κλασικούς αλγόριθμους μηχανικής μάθησης όσο και βαθιάς μάθησης (Gasparetto et al., 2022; Li et al., 2022).

Κεφάλαιο 3. Μεθοδολογία

Στο παρόν κεφάλαιο παρουσιάζονται:

- τα χαρακτηριστικά του προγράμματος «Διαύγεια»,
- τα Dataset που χρησιμοποιήθηκαν,
- το λογισμικό και υλικό που χρησιμοποιήθηκε,
- η διαδικασία προεπεξεργασίας των δεδομένων,
- οι τεχνικές επιλογής χαρακτηριστικών (feature selection),
- η εφαρμογή των μοντέλων μηχανικής και βαθιάς μάθησης,
- η παραμετροποίηση των μοντέλων μηχανικής και βαθιάς μάθησης.

3.1 Πρόγραμμα Διαύγεια[1]

Η ανάγκη εισαγωγής των τεχνολογιών πληροφορικής και επικοινωνιών στη λειτουργία του δημόσιου τομέα στο πλαίσιο της ηλεκτρονικής διακυβέρνησης οδήγησε στη δημιουργία του προγράμματος «Διαύγεια». Σκοπός του προγράμματος είναι η δημοσίευση στο διαδίκτυο των αποφάσεων (θα αναφέρονται ως πράξεις στη συνέχεια) των κυβερνητικών οργάνων, των φορέων του στενού και του ευρύτερου δημόσιου τομέα, των Ανεξάρτητων Αρχών και των οργανισμών τοπικής αυτοδιοίκησης Α΄ και Β΄ βαθμού. Το πρόγραμμα «Διαύγεια» έχει στόχο να βελτιώσει την ποιότητα και την αποτελεσματικότητα των υπηρεσιών του δημόσιου τομέα και να διευκολύνει τους πολίτες και τις επιχειρήσεις στην αλληλεπίδρασή τους με τον δημόσιο τομέα.

3.1.1 OpenData API[2]

Το σύστημα της «Διαύγειας» προσφέρει, μέσω αποθετηρίου του στο Github[3], πρόσβαση σε δείγματα client κώδικα διεπαφής με το API του. Οι χρήστες, μέσω της αποστολής κατάλληλα διαμορφωμένων HTTP αιτημάτων, χρησιμοποιούν τις παρεχόμενες λειτουργίες του συστήματος όπως π.χ. η ανάρτηση, η ενημέρωση και η αναζήτηση πράξεων.
Το κύριο format που υποστηρίζεται για την υποβολή κλήσεων προς το API είναι το JSON[4].

[1] https://diavgeia.gov.gr/

[2] https://diavgeia.gov.gr/api/help

[3] https://github.com/diavgeia

[4] https://www.json.org/json-en.html

3.1.2 ΑΔΑ και μεταδεδομένα πράξεων

Όταν ολοκληρωθεί η διαδικασία ανάρτησης μιας πράξης στο σύστημα «Διαύγεια», αυτή αποκτά έναν αλγοριθμικά παραγόμενο μοναδικό Αριθμό Διαδικτυακής Ανάρτησης (ΑΔΑ), ο οποίος την πιστοποιεί. Εκτός από τον ΑΔΑ, κάθε πράξη συνοδεύεται από ένα σύνολο μεταδεδομένων τα οποία παρουσιάζονται παρακάτω.

Κάθε πράξη αποτελείται από τα εξής μέρη:

- Ένα σύνολο μεταδεδομένων τα οποία περιγράφουν το σκοπό και το περιεχόμενο της, καθώς και τον εκδότη της (φορέας, μονάδα, τελικός υπογράφων).
- Το έγγραφο της πράξης σε μορφή PDF (ψηφιακά υπογεγραμμένο από το σύστημα).
- Προαιρετικά, ένα σύνολο συνοδευτικών εγγράφων (συνημμένα)
- Έναν αριθμό έκδοσης (Version ID)

Μερικά από τα βασικά ή κοινά μεταδεδομένα που διαθέτουν όλες οι πράξεις είναι:

- Θέμα πράξης
- Ημερομηνία έκδοσης (Unix timestamp)
- Είδος πράξης
- Κωδικός της μονάδας του φορέα που εμπλέκεται στην έκδοση της πράξης
- Θεματικές κατηγορίες πράξης

Στη «Διαύγεια» υπάρχουν διαθέσιμες εικοσιπέντε (25) Θεματικές κατηγορίες και τριανταπέντε (35) Είδη πράξεων, ώστε να επιλεγούν για να χαρακτηρίσουν κάποια πράξη. Το ΠΑΔΑ χρησιμοποιεί δεκατέσσερις (14) Θεματικές κατηγορίες και δεκαεννέα (19) Είδη πράξεων στις πράξεις που έχει ανεβάσει στη «Διαύγεια».

Οι διαθέσιμες Θεματικές κατηγορίες στη σελίδα της «Διαύγειας» παρουσιάζονται στον ακόλουθο πίνακα:

Πίνακας 1: Διαθέσιμες Θεματικές κατηγορίες πράξεων της Διαύγειας

ΔΙΑΘΕΣΙΜΕΣ ΘΕΜΑΤΙΚΕΣ ΚΑΤΗΓΟΡΙΕΣ ΠΡΑΞΕΩΝ ΣΤΗ ΔΙΑΥΓΕΙΑ		
ΠΟΛΙΤΙΚΗ ΖΩΗ	ΕΠΙΣΤΗΜΕΣ	ΒΙΟΜΗΧΑΝΙΑ
ΔΙΕΘΝΕΙΣ ΣΧΕΣΕΙΣ	ΕΠΙΧΕΙΡΗΣΕΙΣ ΚΑΙ ΑΝΤΑΓΩΝΙΣΜΟΣ	ΓΕΩΓΡΑΦΙΑ
ΕΥΡΩΠΑΪΚΗ ΈΝΩΣΗ	ΑΠΑΣΧΟΛΗΣΗ ΚΑΙ ΕΡΓΑΣΙΑ	ΔΙΕΘΝΕΙΣ ΟΡΓΑΝΙΣΜΟΙ
ΔΙΚΑΙΟ	ΜΕΤΑΦΟΡΕΣ	ΔΗΜΟΣΙΑ ΔΙΟΙΚΗΣΗ
ΟΙΚΟΝΟΜΙΚΗ ΖΩΗ	ΠΕΡΙΒΑΛΛΟΝ	ΥΓΕΙΑ
ΟΙΚΟΝΟΜΙΚΕΣ ΚΑΙ ΕΜΠΟΡΙΚΕΣ ΣΥΝΑΛΛΑΓΕΣ	ΠΑΡΑΓΩΓΗ, ΤΕΧΝΟΛΟΓΙΑ ΚΑΙ ΕΡΕΥΝΑ	ΑΠΟΦΑΣΗ ΔΙΑΘΕΣΗΣ ΑΝΟΙΚΤΩΝ ΔΕΔΟΜΕΝΩΝ
ΔΗΜΟΣΙΟΝΟΜΙΚΑ	ΔΙΑΤΡΟΦΗ ΚΑΙ ΓΕΩΡΓΙΚΑ ΠΡΟΪΟΝΤΑ	ΔΑΠΑΝΕΣ ΕΠΙΧΟΡΗΓΟΥΜΕΝΩΝ ΦΟΡΕΩΝ ΑΡΘΡΟΥ 10Β Ν 3861/10

ΚΟΙΝΩΝΙΚΑ ΘΕΜΑΤΑ	ΓΕΩΡΓΙΑ, ΔΑΣΟΚΟΜΙΑ ΚΑΙ ΑΛΙΕΙΑ	ΕΝΕΡΓΕΙΑ
ΕΠΙΚΟΙΝΩΝΙΑ ΚΑΙ ΜΟΡΦΩΣΗ		

Τα διαθέσιμα Είδη πράξεων στη σελίδα της «Διαύγειας» παρουσιάζονται στον ακόλουθο πίνακα:

Πίνακας 2: Διαθέσιμα Είδη πράξεων της Διαύγειας

ΔΙΑΘΕΣΙΜΑ ΕΙΔΗ ΠΡΑΞΕΩΝ ΣΤΗ ΔΙΑΥΓΕΙΑ		
ΝΟΜΟΣ	ΕΓΚΥΚΛΙΟΣ	ΓΝΩΜΟΔΟΤΗΣΗ
ΠΡΑΞΗ ΝΟΜΟΘΕΤΙΚΟΥ ΠΕΡΙΕΧΟΜΕΝΟΥ (Σύνταγμα, άρθρο 44, παρ 1)	ΠΡΑΚΤΙΚΑ (Νομικού Συμβουλίου του Κράτους)	ΕΚΘΕΣΗ ΑΠΟΤΙΜΗΣΗΣ ΓΙΑ ΤΗΝ ΚΑΤΑΣΤΑΣΗ ΤΗΣ ΥΦΙΣΤΑΜΕΝΗΣ ΝΟΜΟΘΕΣΙΑΣ
ΕΚΘΕΣΗ ΤΗΣ ΚΕΝΤΡΙΚΗΣ ΕΠΙΤΡΟΠΗΣ ΚΩΔΙΚΟΠΟΙΗΣΗΣ	ΠΡΑΞΗ ΠΟΥ ΑΦΟΡΑ ΣΕ ΘΕΣΗ ΓΕΝΙΚΟΥ - ΕΙΔΙΚΟΥ ΓΡΑΜΜΑΤΕΑ - ΜΟΝΟΜΕΛΕΣ ΟΡΓΑΝΟ	ΠΡΑΞΗ ΠΟΥ ΑΦΟΡΑ ΣΕ ΣΥΛΛΟΓΙΚΟ ΟΡΓΑΝΟ - ΕΠΙΤΡΟΠΗ - ΟΜΑΔΑ ΕΡΓΑΣΙΑΣ - ΟΜΑΔΑ ΕΡΓΟΥ - ΜΕΛΗ ΣΥΛΛΟΓΙΚΟΥ ΟΡΓΑΝΟΥ
ΕΓΚΡΙΣΗ ΠΡΟΥΠΟΛΟΓΙΣΜΟΥ	ΕΠΙΤΡΟΠΙΚΟ ΕΝΤΑΛΜΑ	ΑΝΑΛΗΨΗ ΥΠΟΧΡΕΩΣΗΣ
ΔΙΟΡΙΣΜΟΣ	ΠΡΑΞΗ ΥΠΑΓΩΓΗΣ ΕΠΕΝΔΥΣΕΩΝ	ΠΡΑΞΕΙΣ ΧΩΡΟΤΑΞΙΚΟΥ - ΠΟΛΕΟΔΟΜΙΚΟΥ ΠΕΡΙΕΧΟΜΕΝΟΥ
ΛΟΙΠΕΣ ΑΤΟΜΙΚΕΣ ΔΙΟΙΚΗΤΙΚΕΣ ΠΡΑΞΕΙΣ	ΣΥΜΒΑΣΗ-ΠΡΑΞΕΙΣ ΑΝΑΠΤΥΞΙΑΚΩΝ ΝΟΜΩΝ	ΑΠΟΦΑΣΗ ΕΝΑΡΞΗΣ ΠΑΡΑΓΩΓΙΚΗΣ ΛΕΙΤΟΥΡΓΙΑΣ ΕΠΕΝΔΥΣΗΣ
ΑΛΛΗ ΠΡΑΞΗ ΑΝΑΠΤΥΞΙΑΚΟΥ ΝΟΜΟΥ	ΑΘΩΩΤΙΚΗ ΠΕΙΘΑΡΧΙΚΗ ΑΠΟΦΑΣΗ	ΔΗΜΟΣΙΑ ΠΡΟΤΥΠΑ ΕΓΓΡΑΦΑ
ΥΠΗΡΕΣΙΑΚΗ ΜΕΤΑΒΟΛΗ	ΠΕΡΙΛΗΨΗ ΔΙΑΚΗΡΥΞΗΣ	ΠΡΟΚΗΡΥΞΗ ΠΛΗΡΩΣΗΣ ΘΕΣΕΩΝ
ΠΙΝΑΚΕΣ ΕΠΙΤΥΧΟΝΤΩΝ, ΔΙΟΡΙΣΤΕΩΝ & ΕΠΙΛΑΧΟΝΤΩΝ	ΕΓΚΡΙΣΗ ΔΑΠΑΝΗΣ	ΑΝΑΘΕΣΗ ΕΡΓΩΝ / ΠΡΟΜΗΘΕΙΩΝ / ΥΠΗΡΕΣΙΩΝ / ΜΕΛΕΤΩΝ
ΚΑΤΑΚΥΡΩΣΗ	ΣΥΜΒΑΣΗ	ΔΩΡΕΑ - ΕΠΙΧΟΡΗΓΗΣΗ
ΠΑΡΑΧΩΡΗΣΗ ΧΡΗΣΗΣ ΠΕΡΙΟΥΣΙΑΚΩΝ ΣΤΟΙΧΕΙΩΝ	ΟΡΙΣΤΙΚΟΠΟΙΗΣΗ ΠΛΗΡΩΜΗΣ	ΠΡΟΓΡΑΜΜΑΤΙΚΗ ΣΥΜΦΩΝΙΑ ΟΙΚΟΝΟΜΙΚΗΣ ΥΠΟΣΤΗΡΙΞΗΣ
ΚΑΝΟΝΙΣΤΙΚΗ ΠΡΑΞΗ	ΙΣΟΛΟΓΙΣΜΟΣ – ΑΠΟΛΟΓΙΣΜΟΣ	

3.2 Training – Test Datasets

Το Training Dataset αποτελείται από 77396 εγγραφές και αντιστοιχούν σε ισάριθμες πράξεις που έχει ανεβάσει το ΠΑΔΑ στη «Διαύγεια» ενώ το Test Dataset αποτελείται από 5165 εγγραφές.

Η χρονική περίοδος που καλύπτεται, ξεκινάει από τις 22-03-2018 μέχρι και την 21-03-2023 για το Training Dataset ενώ το Test Dataset καλύπτει την περίοδο 22-03-2023 έως 23-06-2023.

Καθώς η παρούσα εργασία ασχολείται με την ταξινόμηση/κατηγοριοποίηση των πράξεων τόσο ως προς την Θεματική κατηγορία που αυτές ανήκουν όσο και ως προς το Είδος κάθε πράξης, θα

παρουσιαστούν στη συνέχεια κάποια στατιστικά στοιχεία σχετικά με τη σύνθεση των δυο Datasets.

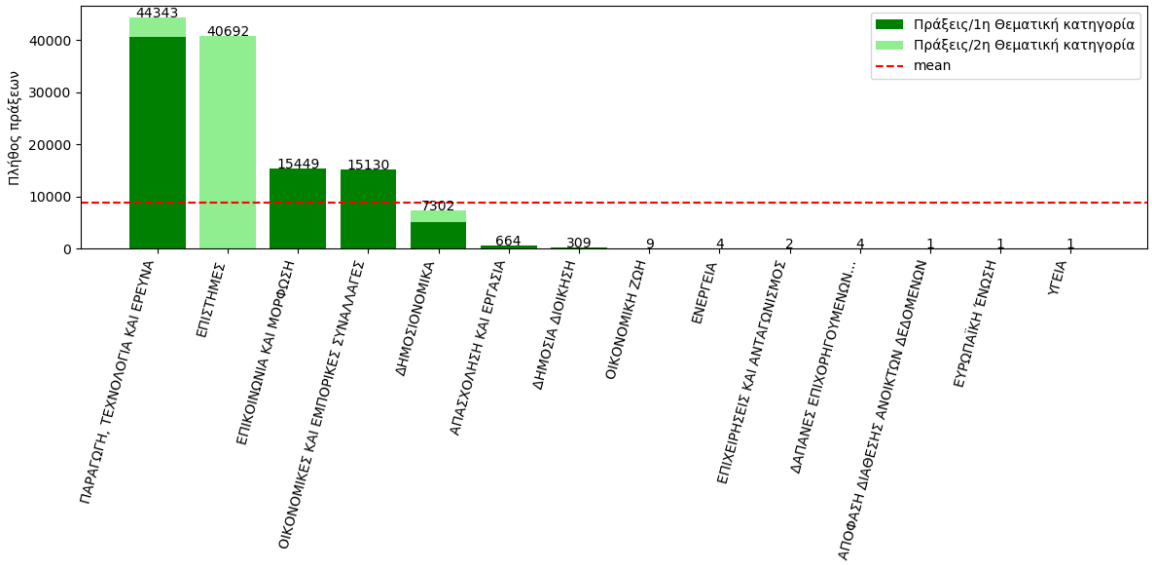
3.2.1 Θεματική κατηγορία πράξης

Κάθε πράξη ανήκει σε τουλάχιστον μια ή το μέγιστο σε δυο από τις δεκατέσσερις (14) θεματικές κατηγορίες που αναφέρονται στον Πίνακας 3. Επομένως, η συγκεκριμένη εργασία ταξινόμησης εντάσσεται στην κατηγορία multilabel .

Πίνακας 3: Θεματικές κατηγορίες πράξεων που χρησιμοποιεί το ΠΑΔΑ

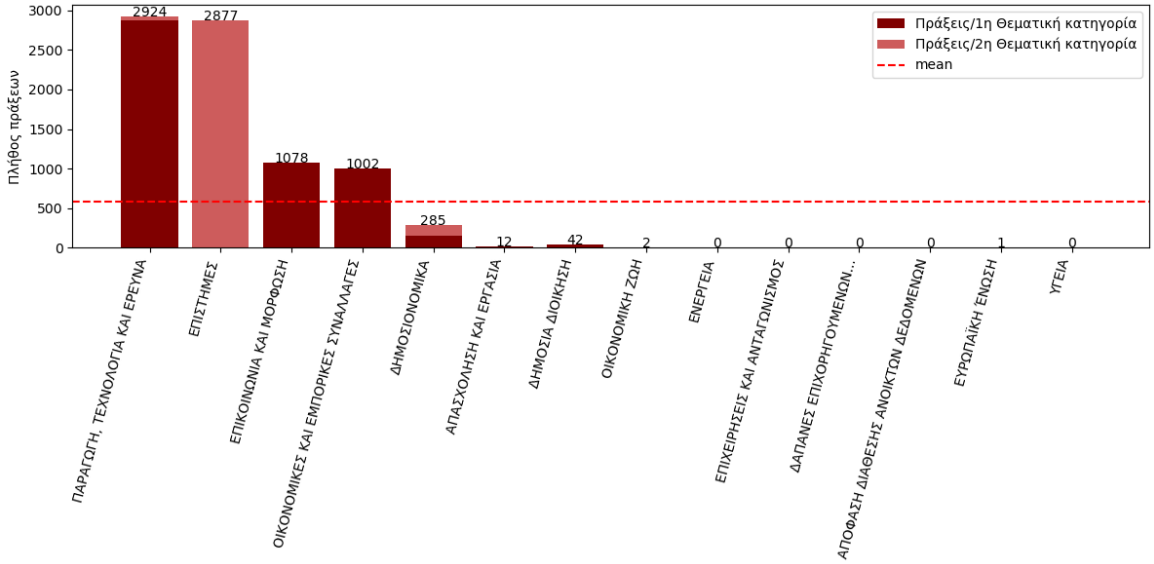
ΘΕΜΑΤΙΚΕΣ ΚΑΤΗΓΟΡΙΕΣ ΠΡΑΞΕΩΝ - ΠΑΔΑ	
ΠΑΡΑΓΩΓΗ, ΤΕΧΝΟΛΟΓΙΑ ΚΑΙ ΕΡΕΥΝΑ	ΟΙΚΟΝΟΜΙΚΗ ΖΩΗ
ΕΠΙΣΤΗΜΕΣ	ΕΝΕΡΓΕΙΑ
ΕΠΙΚΟΙΝΩΝΙΑ ΚΑΙ ΜΟΡΦΩΣΗ	ΕΠΙΧΕΙΡΗΣΕΙΣ ΚΑΙ ΑΝΤΑΓΩΝΙΣΜΟΣ
ΟΙΚΟΝΟΜΙΚΕΣ ΚΑΙ ΕΜΠΟΡΙΚΕΣ ΣΥΝΑΛΛΑΓΕΣ	ΔΑΠΑΝΕΣ ΕΠΙΧ/ΝΩΝ ΦΟΡΕΩΝ ΑΡΘΡΟΥ 10Β Ν 3861/10
ΔΗΜΟΣΙΟΝΟΜΙΚΑ	ΑΠΟΦΑΣΗ ΔΙΑΘΕΣΗΣ ΑΝΟΙΚΤΩΝ ΔΕΔΟΜΕΝΩΝ
ΑΠΑΣΧΟΛΗΣΗ ΚΑΙ ΕΡΓΑΣΙΑ	ΕΥΡΩΠΑΪΚΗ ΈΝΩΣΗ
ΔΗΜΟΣΙΑ ΔΙΟΙΚΗΣΗ	ΥΓΕΙΑ

Η κατανομή των πράξεων του Training Dataset ανά 1η ή 2η Θεματική κατηγορία φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 8850:



Εικόνα 13: Κατανομή θεματικών κατηγοριών στο Training Dataset (22/03/2018 – 21/03/2023)

Η κατανομή των πράξεων του Test Dataset ανά 1η ή 2η Θεματική κατηγορία φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 587:



Εικόνα 14: Κατανομή Θεματικής κατηγορίας στο Test Dataset (22/03/2023 – 23/06/2023)

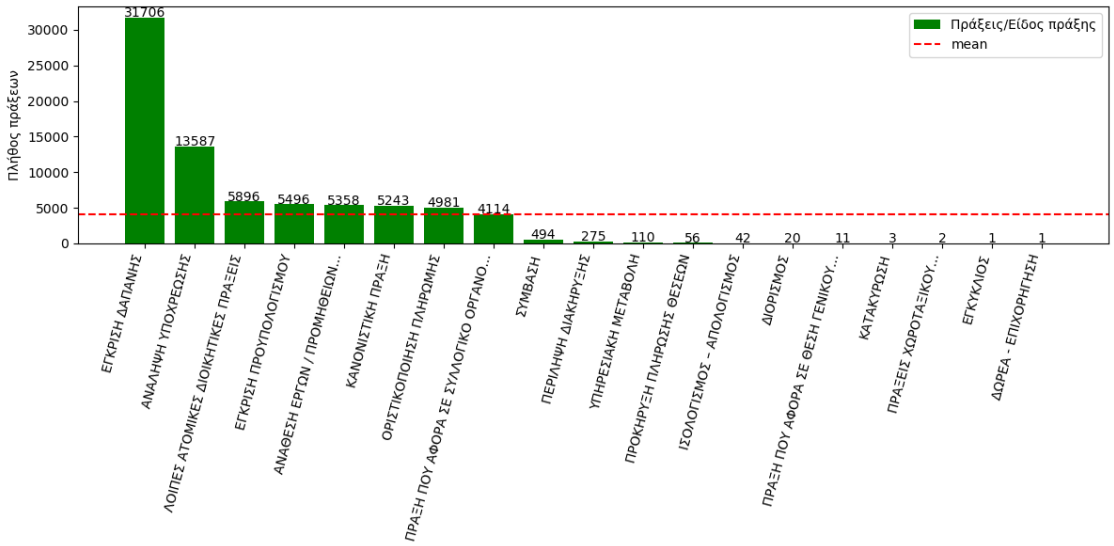
3.2.2 Είδος πράξης

Επιπρόσθετα, κάθε πράξη μπορεί να ανήκει τουλάχιστον σε ένα από τα δεκαεννέα (19) είδη πράξεων που αναφέρονται στον Πίνακα 2. Επομένως η συγκεκριμένη εργασία ταξινόμησης εντάσσεται στην κατηγορία multiclass.

Πίνακας 4: Είδη πράξεων που χρησιμοποιεί το ΠΑΔΑ

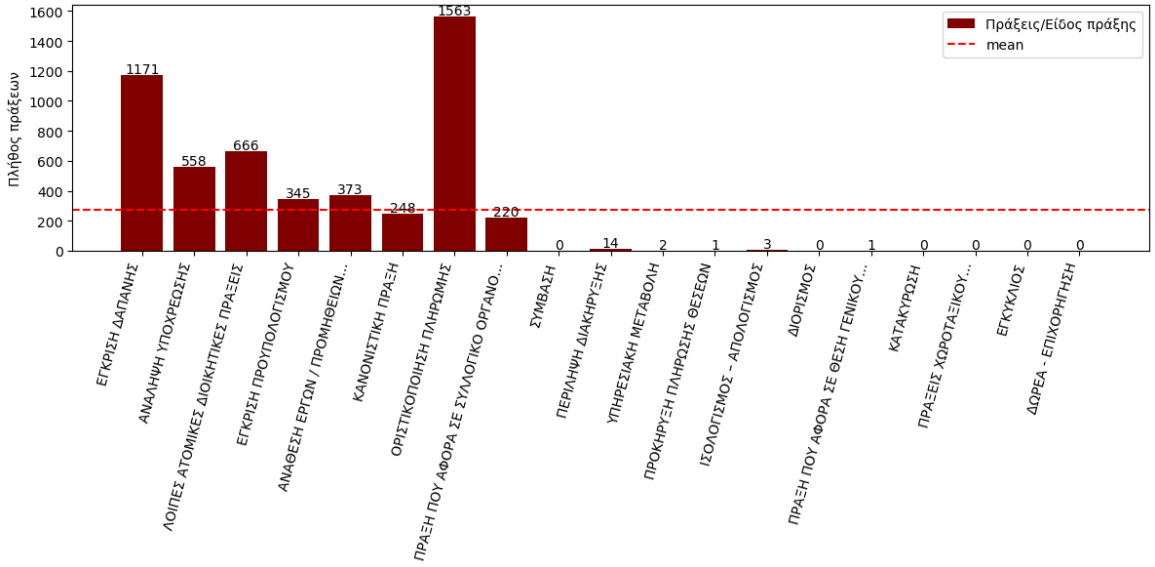
ΕΙΔΗ ΠΡΑΞΕΩΝ - ΠΑΔΑ	
ΕΓΚΡΙΣΗ ΔΑΠΑΝΗΣ	ΥΠΗΡΕΣΙΑΚΗ ΜΕΤΑΒΟΛΗ
ΑΝΑΛΗΨΗ ΥΠΟΧΡΕΩΣΗΣ	ΠΡΟΚΗΡΥΞΗ ΠΛΗΡΩΣΗΣ ΘΕΣΕΩΝ
ΛΟΙΠΕΣ ΑΤΟΜΙΚΕΣ ΔΙΟΙΚΗΤΙΚΕΣ ΠΡΑΞΕΙΣ	ΙΣΟΛΟΓΙΣΜΟΣ – ΑΠΟΛΟΓΙΣΜΟΣ
ΕΓΚΡΙΣΗ ΠΡΟΥΠΟΛΟΓΙΣΜΟΥ	ΔΙΟΡΙΣΜΟΣ
ΑΝΑΘΕΣΗ ΕΡΓΩΝ/ΠΡΟΜΗΘΕΙΩΝ/ΥΠΗΡΕΣΙΩΝ/ΜΕΛΕΤΩΝ	ΕΓΚΥΚΛΙΟΣ
ΚΑΝΟΝΙΣΤΙΚΗ ΠΡΑΞΗ	ΚΑΤΑΚΥΡΩΣΗ
ΟΡΙΣΤΙΚΟΠΟΙΗΣΗ ΠΛΗΡΩΜΗΣ	ΠΡΑΞΕΙΣ ΧΩΡΟΤΑΞΙΚΟΥ - ΠΟΛΕΟΔΟΜΙΚΟΥ ΠΕΡΙΕΧΟΜΕΝΟΥ
ΠΡΑΞΗ ΠΟΥ ΑΦΟΡΑ ΣΕ ΣΥΛΛΟΓΙΚΟ ΟΡΓΑΝΟ – ΕΠΙΤΡΟΠΗ - ΟΜΑΔΑ ΕΡΓΑΣΙΑΣ - ΟΜΑΔΑ ΕΡΓΟΥ - ΜΕΛΗ ΣΥΛΛΟΓΙΚΟΥ ΟΡΓΑΝΟΥ	ΠΡΑΞΗ ΠΟΥ ΑΦΟΡΑ ΣΕ ΘΕΣΗ ΓΕΝΙΚΟΥ - ΕΙΔΙΚΟΥ ΓΡΑΜΜΑΤΕΑ - ΜΟΝΟΜΕΛΕΣ ΟΡΓΑΝΟ
ΣΥΜΒΑΣΗ	ΔΩΡΕΑ - ΕΠΙΧΟΡΗΓΗΣΗ
ΠΕΡΙΛΗΨΗ ΔΙΑΚΗΡΥΞΗΣ	

Η κατανομή των πράξεων ανά Είδος πράξης στο Training Dataset φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 4073:



Εικόνα 15: Κατανομή Είδους πράξης στο Training Dataset (22/03/2018 – 21/03/2023)

Αντίστοιχα για το Test Dataset στο ακόλουθο διάγραμμα, η μέση τιμή είναι 272 :



Εικόνα 16: Κατανομή Είδους πράξης στο Test Dataset (22/03/2023 – 23/06/2023)

3.2.3 Επιπλέον ενέργειες – Βήματα

Στην προσπάθεια ερμηνείας και αντιμετώπισης των πολύ χαμηλών αποτελεσμάτων που πέτυχαν οι αλγόριθμοι στο στάδιο της επικύρωσης (validation) της ταξινόμησης του Είδους πράξης και εμφανίζονται στην παράγραφο 4.2.2, μελετήθηκε η δομή του Test Dataset και στη συνέχεια συγκρίθηκε με αυτή του Training Dataset.

Διαπιστώθηκε ότι υπάρχει ασυμφωνία των στατιστικών μεταξύ των δομών των δυο Datasets. Όπως είναι εμφανές από την Εικόνα 16: Κατανομή Είδους πράξης στο Test Dataset, της κατανομής των κλάσεων «Είδος πράξης» στο Test Dataset, εμφανίζεται μια κορυφή στην κλάση «Οριστικοποίηση πληρωμής». Επίσης, από την ανάλυση του Test Dataset φαίνεται ότι το μεγαλύτερο πλήθος πράξεων με την συγκεκριμένη κλάση εμφανίζεται τους μήνες Μάιο και Ιούνιο.

Ακολουθήθηκαν οι παρακάτω στρατηγικές ώστε να διαπιστωθεί αν τα αποτελέσματα επηρεάζονται από τη στατιστική δομή του Test Dataset ή αν παίζει ρόλο η αλγοριθμική υλοποίηση. Σκοπός των ενεργειών αυτών είναι η συμφωνία των στατιστικών μεγεθών στα δυο Datasets.

- Διατήρηση του αρχικού Training Dataset και μείωση του εύρους της χρονικής περιόδου που καλύπτει το Test Dataset κατά ένα μήνα: η χρονική περίοδος ορίστηκε από τις 22/03/2023 έως 23/05/2023.
- Διατήρηση του αρχικού Training Dataset και μείωση του εύρους της χρονικής περιόδου που καλύπτει το Test Dataset κατά δυο μήνες: η χρονική περίοδος ορίστηκε από τις 22/03/2023 έως 30/04/2023.
- Ενσωμάτωση του Test Dataset (22/03/2023 – 23/06/2023) στο Training Dataset (22/03/2018 – 21/03/2023). Το νέο Training Dataset καλύπτει την περίοδο 22/03/2018 έως 23/06/2023. Ορισμός νέου Test Dataset που καλύπτει τη χρονική περίοδο 24/06/2023 έως 23/08/2023. Τα νέα πλήθη των πράξεων είναι 82561 για το Training Dataset και 2890 για το Test Dataset.

Για τις δυο πρώτες από τις παραπάνω στρατηγικές επαναλήφθηκε η φάση επικύρωσης (validation) για τα διανύσματα που προέκυψαν από την φάση της αξιολόγησης (evaluation) καθώς πέτυχαν τα υψηλότερα αποτελέσματα ώστε να ελεγχθεί η συμπεριφορά των μοντέλων στα καινούργια Datasets. Για την τρίτη στρατηγική επαναλήφθηκαν οι φάσεις αξιολόγησης και επικύρωσης με τα νέα Datasets μόνο για τα προαναφερθέντα διανύσματα όπως και παραπάνω. Δεν έγινε εκ νέου διερεύνηση σε διανύσματα διαφορετικού μήκους, από την οποία ενδεχομένως θα προέκυπτε ως βέλτιστο κάποιο διαφορετικό διάνυσμα.

Οι προαναφερθείσες ενέργειες εφαρμόστηκαν τόσο για την ταξινόμηση των εγγράφων βάσει του είδους πράξης όπου είχε διαπιστώθηκε η στατιστική ασυμφωνία των Datasets όσο και στην

ταξινόμηση βάσει της Θεματικής κατηγορίας. Ο έλεγχος της συμπεριφοράς των αλγορίθμων στην ταξινόμηση της Θεματικής κατηγορίας με τα νέα Datasets κρίθηκε αναγκαίος, παρά τα πολύ καλά αποτελέσματα που είχαν επιτευχθεί με τα αρχικά Datasets, ώστε να αποκλειστεί η πιθανότητα τα αρχικά αποτελέσματα να προέκυψαν τυχαία και να επιβεβαιωθεί η υψηλή απόδοση των αλγορίθμων.

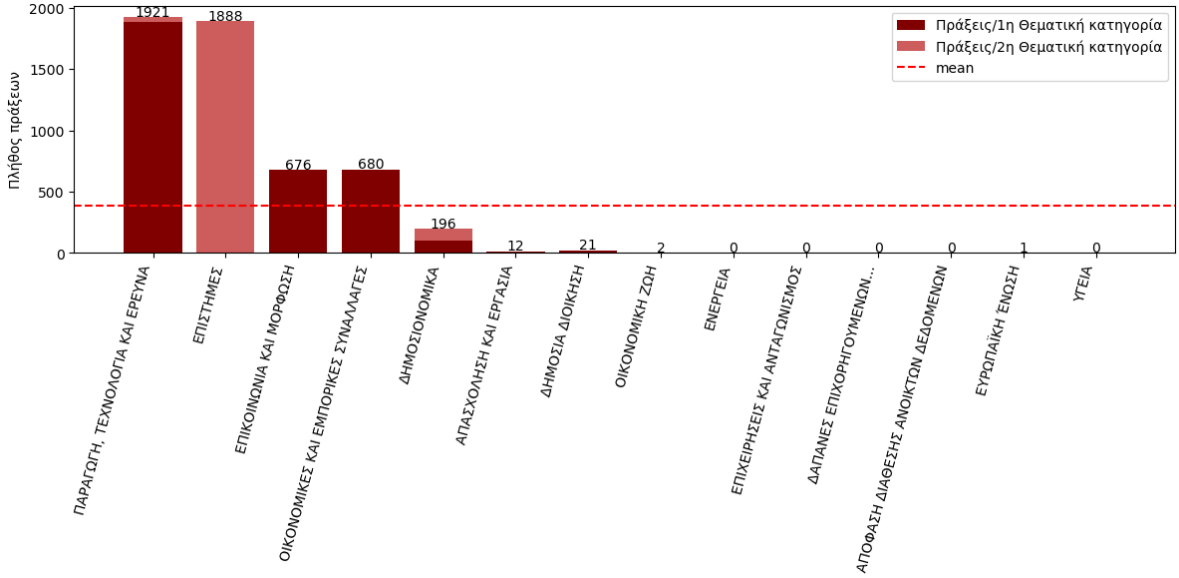
Τα αποτελέσματα των παραπάνω ενεργειών παρουσιάζονται για την Θεματική κατηγορία στην παράγραφο 0 ενώ για το Είδος πράξης στην παράγραφο 4.2.3.

Στη συνέχεια, παρουσιάζονται τα διαγράμματα κατανομής των πράξεων τόσο ανά Θεματική κατηγορία όσο και ανά Είδος πράξης για κάθε μια από τις παραπάνω στρατηγικές και τα νέα Test και Training Datasets που προέκυψαν βάσει αυτών.

- Μείωση του εύρους της χρονικής περιόδου που καλύπτει το Test Dataset κατά ένα μήνα: η χρονική περίοδος ορίστηκε από τις 22/03/2023 έως 23/05/2023.

Θεματική κατηγορία

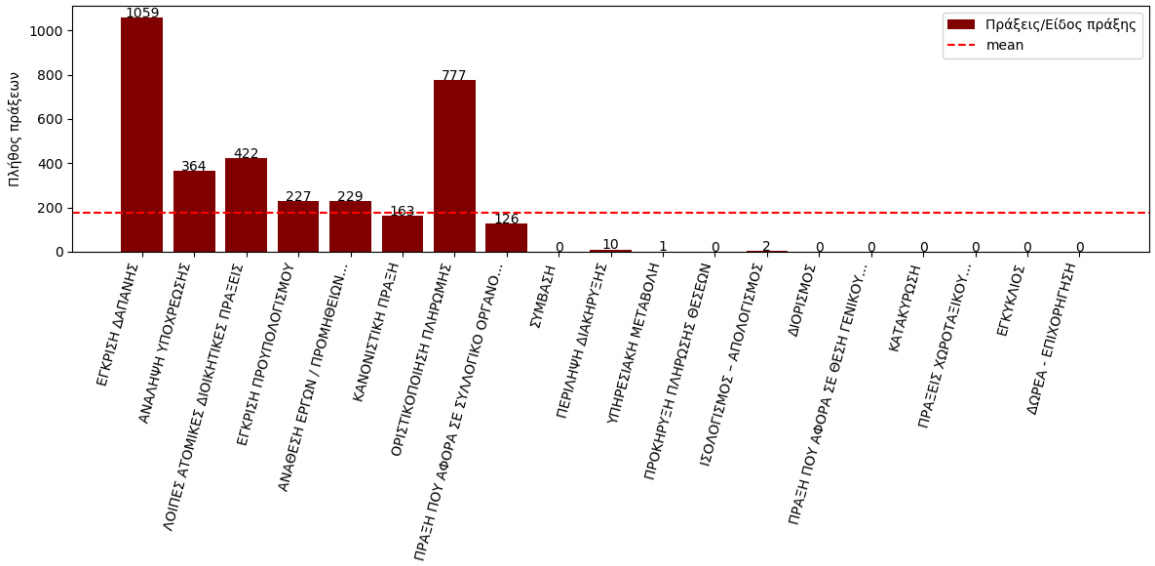
Η κατανομή των πράξεων του Test Dataset ανά 1η ή 2η Θεματική κατηγορία φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 385:



Εικόνα 17: Κατανομή Θεματικής κατηγορίας στο Test Dataset (22/03/2023 – 23/05/2023)

Είδος πράξης

Η κατανομή των πράξεων του Test Dataset ανά Είδος πράξης φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 178:

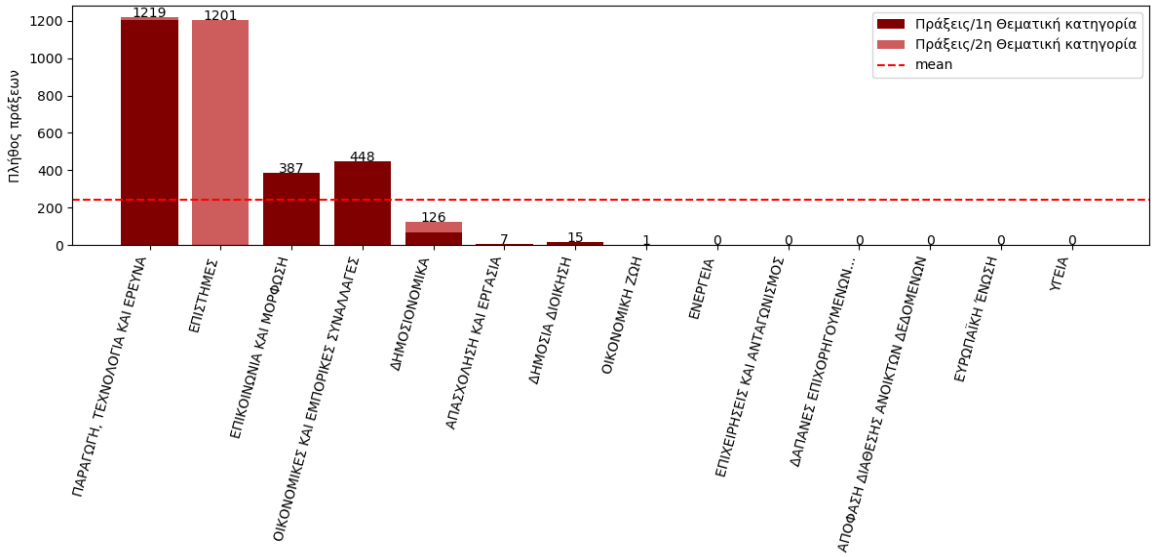


Εικόνα 18: Κατανομή Είδους πράξης στο Test Dataset (22/03/2023 - 23/05/2023)

- Μείωση του εύρους της χρονικής περιόδου που καλύπτει το Test Dataset κατά δυο μήνες: η χρονική περίοδος ορίστηκε από τις 22/03/2023 έως 30/04/2023.

Θεματική κατηγορία

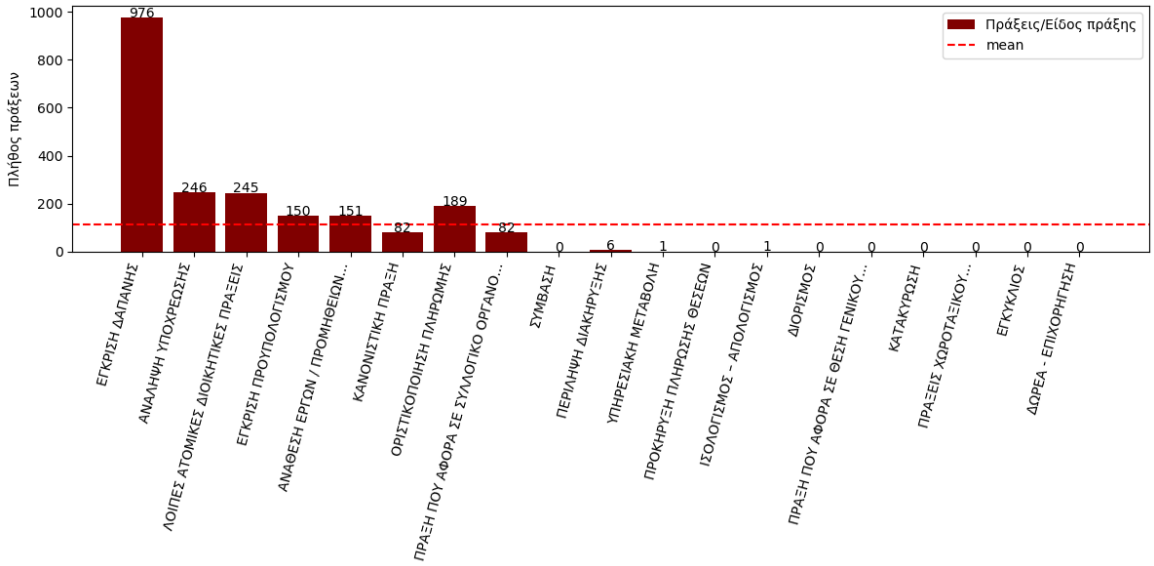
Η κατανομή των πράξεων του Test Dataset ανά 1η ή 2η Θεματική κατηγορία φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 243:



Εικόνα 19: Κατανομή Θεματικής κατηγορίας στο Test Dataset (22/03/2023 - 30/04/2023)

Είδος πράξης

Η κατανομή των πράξεων του Test Dataset ανά Είδος πράξης κατηγορία φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 112:

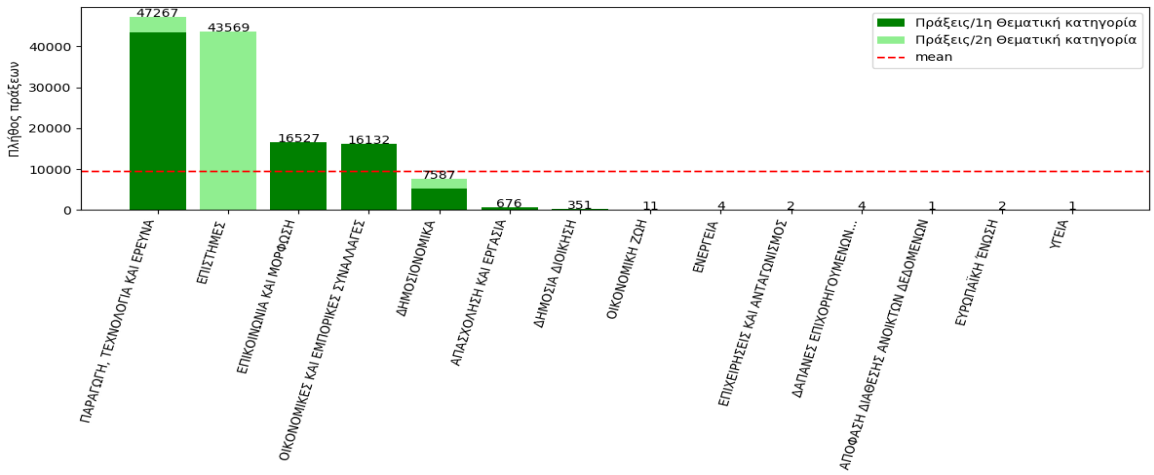


Εικόνα 20: Κατανομή Είδους πράξης στο Test Dataset (22/03/2023 – 30/04/2023)

- Ενσωμάτωση αρχικού Test Dataset στο Training Dataset. Το νέο Training Dataset καλύπτει την περίοδο 22/03/2018 έως 23/06/2023. Δημιουργία νέου Test Dataset που καλύπτει τη χρονική περίοδο 24/06/2023 έως 23/08/2023.

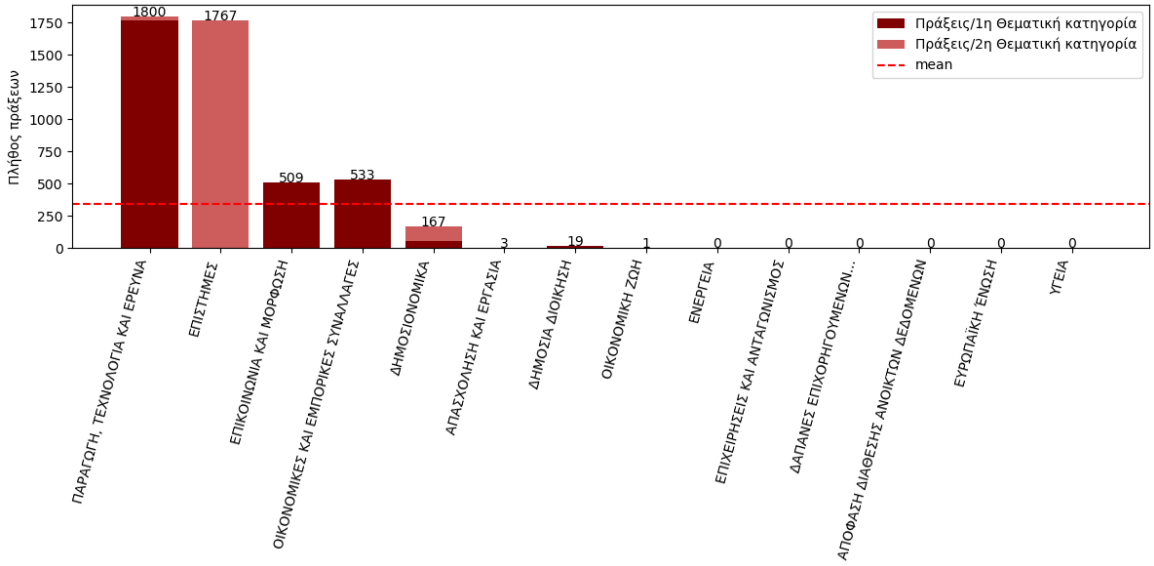
Θεματική κατηγορία

Η κατανομή των πράξεων του Training Dataset ανά 1η ή 2η Θεματική κατηγορία φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 9438:



Εικόνα 21: Κατανομή Θεματικής κατηγορίας στο νέο Training Dataset (22/03/2018 – 23/06/2023)

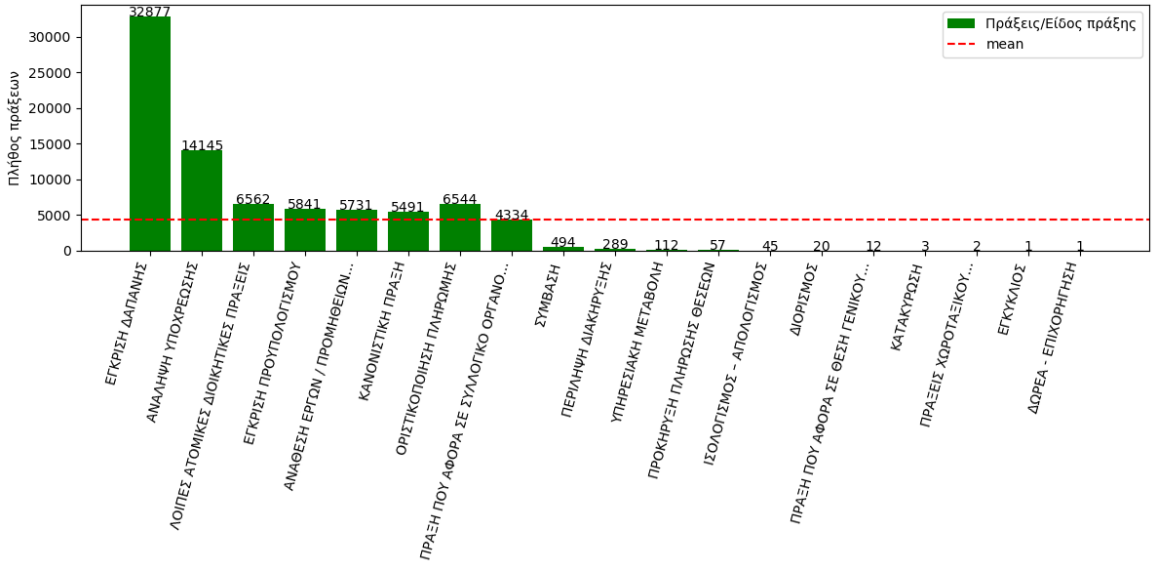
Η κατανομή των πράξεων του Test Dataset ανά 1η ή 2η Θεματική κατηγορία φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 343:



Εικόνα 22: Κατανομή Θεματικής κατηγορίας στο νέο Test Dataset (24-06/2023 – 23/08/2023

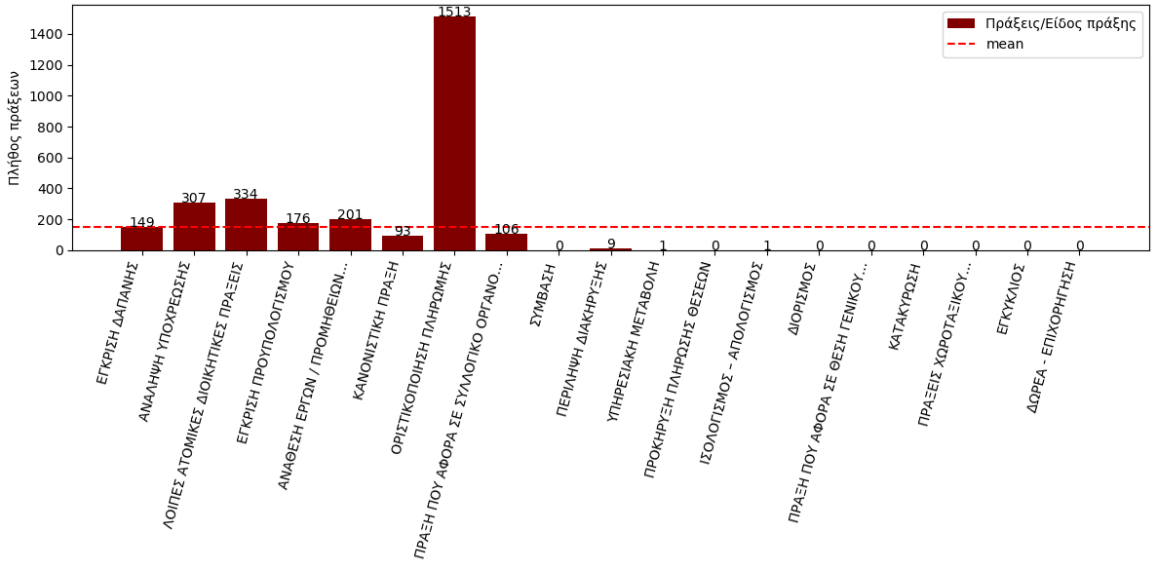
Είδος πράξης

Η κατανομή των πράξεων ανά Είδος πράξης στο νέο Training Dataset φαίνεται στο ακόλουθο διάγραμμα , η μέση τιμή είναι 4345:



Εικόνα 23: Κατανομή Είδους πράξης στο νέο Training Dataset (22/03/2018 - 23/06/2023)

Η κατανομή των πράξεων του νέου Test Dataset ανά Είδος πράξης κατηγορία φαίνεται στο ακόλουθο διάγραμμα, η μέση τιμή είναι 152:



Εικόνα 24: Κατανομή Είδους πράξης στο νέο Test Dataset (24/06/2023 – 23/08/2023)

Λόγω του πλήθους των Training και Test Datasets ακολουθείται η παρακάτω κωδικοποίηση στους τίτλους των παραγράφων και στις ετικέτες πινάκων και σχημάτων ανάλογα με την χρονική περίοδο που καλύπτουν και την διαδικασία που εμπλέκονται:

Training1: χρονική περίοδος από 22-03-2018 μέχρι και την 21-03-2023

Training2: χρονική περίοδος από 22-03-2018 μέχρι και την 23-06-2023

Test1: χρονική περίοδος από 22-03-2023 μέχρι και την 23-06-2023

Test2: χρονική περίοδος από 22-03-2023 μέχρι και την 23-05-2023

Test3: χρονική περίοδος από 22-03-2023 μέχρι και την 30-04-2023

Test4: χρονική περίοδος από 24-06-2023 μέχρι και την 23-08-2023

10f-: φάση αξιολόγησης, evaluation phase (10-fold cross-validation)

Val-: φάση επικύρωσης, validation phase

3.3 Λογισμικό

Στην παρούσα εργασία χρησιμοποιήθηκαν τόσο η εφαρμογή PyCharm[5] στα πρώτα στάδια της (ανάκτηση μεταδεδομένων, εξαγωγή κειμένου από PDF, προ-επεξεργασία κειμένου) όσο και η web-based πλατφόρμα Jupyter[6] για την εφαρμογή των αλγορίθμων μηχανικής και βαθιάς μάθησης. Η έκδοση της γλώσσας προγραμματισμού Python[7] ήταν η 3.11.

Στη συνέχεια αναφέρονται οι βιβλιοθήκες της Python που χρησιμοποιήθηκαν:

NumPy (Harris et al., 2020): είναι μια βιβλιοθήκη ιδιαίτερα χρήσιμη για την εκτέλεση αριθμητικών και επιστημονικών υπολογισμών. Χρησιμοποιείται για υπολογισμούς και εφαρμογές γραμμικής άλγεβρας, στατιστικής ανάλυσης, μοντελοποίηση δεδομένων και άλλα.

Pandas (McKinney, 2010): είναι μια βιβλιοθήκη ανοικτού κώδικα. Μια από τις βασικές δομές δεδομένων που παρέχει είναι το DataFrame, στην ουσία ένας δισδιάστατος πίνακας που περιλαμβάνει πολλές στήλες δεδομένων. Οι λειτουργίες της βιβλιοθήκης περιλαμβάνουν τη φόρτωση και την αποθήκευση δεδομένων από/προς διάφορες πηγές, την επεξεργασία και τον καθαρισμό των δεδομένων, τον υπολογισμό στατιστικών και άλλα.

Matplotlib (Hunter, 2007): είναι μια βιβλιοθήκη που χρησιμοποιείται για την οπτικοποίηση δεδομένων και τη δημιουργία γραφημάτων.

NLTK – Natural Language Toolkit (Bird et al., 2009), **spaCy** (Honnibal & Montani, 2017): είναι βιβλιοθήκες που παρέχουν εργαλεία και λειτουργίες για την επεξεργασία κειμένου και την ανάλυση γλώσσας, όπως το διαχωρισμό κειμένου σε λέξεις, την εντοπισμό οντοτήτων, την ανάλυση της σύνταξης, τον ποσοτικό προσδιορισμό λέξεων κλπ.

Scikit-Learn (Pedregosa et al., 2011): αποτελεί μία από τις πιο δημοφιλείς βιβλιοθήκες μηχανικής μάθησης στο περιβάλλον της Python. **Scikit-Multilearn** (Szymański & Kajdanowicz, 2018): βιβλιοθήκη, βασισμένη στην Scikit-Learn, προσανατολισμένη σε multilabel προβλήματα.

SciPy (Virtanen et al., 2020): αποτελεί μια συλλογή από διάφορες βιβλιοθήκες (NumPy, Pandas, Matplotlib, SciPy). Ως βιβλιοθήκη, μεταξύ των άλλων, παρέχει δυνατότητες διαχείρισης αραιών πινάκων (sparce matrix).

PyMuPDF (McKie & Liu, 2016): είναι μια ισχυρή βιβλιοθήκη για την επεξεργασία PDF αρχείων και μπορεί να χρησιμοποιηθεί για διάφορες εργασίες, όπως η ανάγνωση PDF αρχείων και η εξαγωγή κειμένου από αυτά. Βασίζεται στην πυρήνα της παλαιότερης βιβλιοθήκης MuPDF.

[5] https://www.jetbrains.com/pycharm/

[6] https://jupyter.org/

[7] https://www.python.org/

Tensorflow (Martín Abadi et al., 2015): είναι μια ανοικτού κώδικα βιβλιοθήκη μηχανικής μάθησης που αναπτύχθηκε από την ομάδα Google Brain της Google. Επιτρέπει την ανάπτυξη και εκτέλεση μοντέλων μηχανικής μάθησης, βαθιών νευρωνικών δικτύων και αλγορίθμων επιβλεπόμενης και μη επιβλεπόμενης μάθησης.

Keras (Chollet & others, 2015): είναι μια βιβλιοθήκη μηχανικής μάθησης που παρέχει ένα απλό και ευέλικτο περιβάλλον ανάπτυξης για τη δημιουργία μοντέλων νευρωνικών δικτύων.

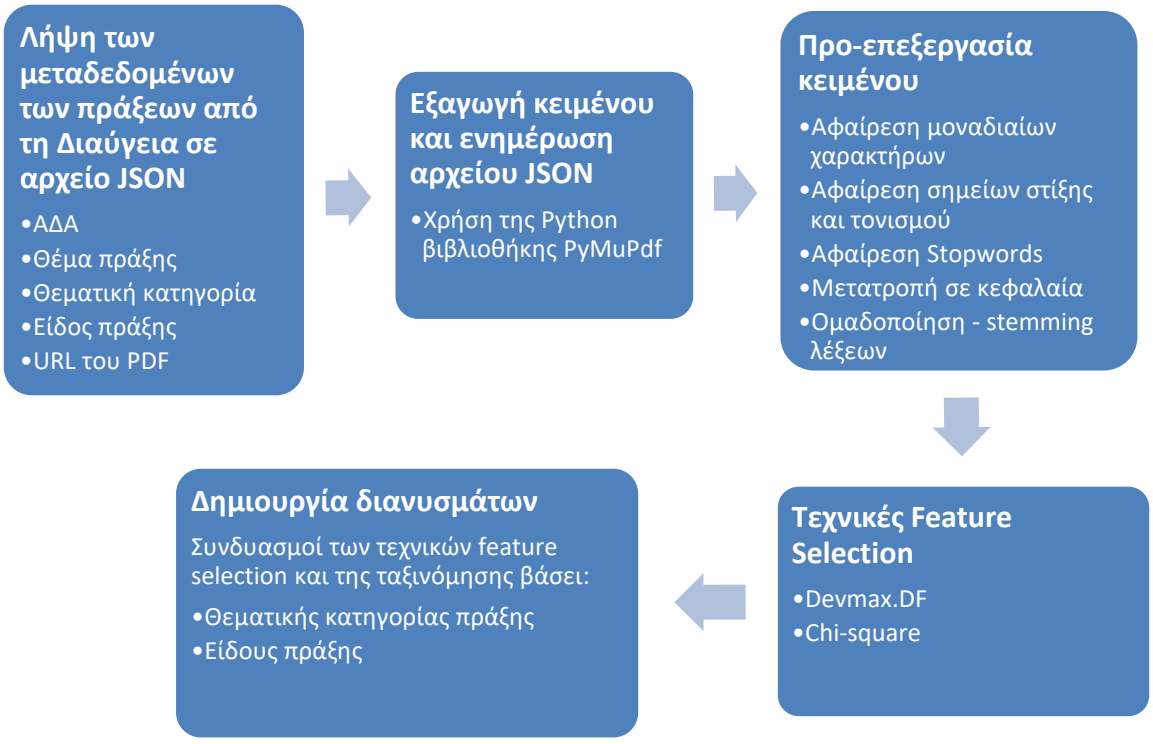
3.4 Υλικό

Η εκτέλεση του συνόλου του κώδικα πραγματοποιήθηκε σε ένα Toshiba Dynabook laptop με τα παρακάτω χαρακτηριστικά:

- CPU: AMD Ryzen 5 5600U @ 2.3GHz
- RAM: 24 GB
- Windows 11, 64-bit

3.5 Λήψη μεταδεδομένων, εξαγωγή κειμένου, προεπεξεργασία

Τα βήματα που ακολουθήθηκαν για την δημιουργία των αρχείων διανυσμάτων, που χρησιμοποιήθηκαν στη συνέχεια στους αλγορίθμους μηχανικής και βαθιάς μάθησης, φαίνονται στο ακόλουθο διάγραμμα ροής.



Εικόνα 25: Διάγραμμα ροής – δημιουργία διανυσμάτων

3.5.1 Λήψη μεταδεδομένων και εξαγωγή κειμένου

Για την λήψη των μεταδεδομένων των πράξεων του ΠΑΔΑ, έγινε χρήση του API που παρέχει η «Διαύγεια». Τα μεταδεδομένα αποθηκεύτηκαν σε JSON αρχείο, το οποίο χρησιμοποιήθηκε και στα επόμενα στάδια της διαδικασίας. Ο κώδικας παρουσιάζεται στο Παράρτημα Α.

Για την εξαγωγή κειμένου από τα αρχεία PDF χρησιμοποιήθηκε η βιβλιοθήκη PyMuPDF της Python. Κρίθηκε προτιμότερο, για την διευκόλυνση των επόμενων σταδίων της διαδικασίας, να γίνει λήψη του συνόλου των PDF αρχείων και επεξεργασία τους τοπικά. Ο κώδικας για την λήψη των αρχείων υπάρχει στο Παράρτημα Β ενώ για την εξαγωγή του κειμένου στο Παράρτημα Γ.

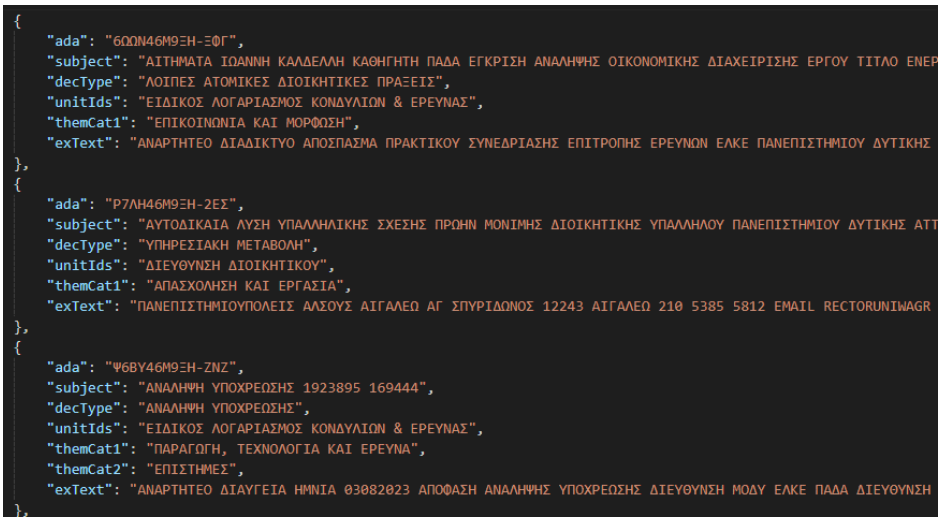
Δεν ήταν εφικτή η εξαγωγή κειμένου σε τρία αρχεία pdf μιας και το αρχείο περιείχε εικόνα (scan ή φωτογραφία) του εγγράφου της πράξης. Επίσης διαπιστώθηκε ότι η εξαγωγή του κεφαλαίου γράμματος **Δ** γινόταν σε encoding που αντιστοιχούσε σε μαθηματικό σύμβολο που ανήκε στο μπλοκ «Μαθηματικοί τελεστές» του Unicode (Unicode: u"\u2206). Έγινε αντικατάσταση με τον κατάλληλο Unicode χαρακτήρα και διορθώθηκε στο επόμενο στάδιο της προεπεξεργασίας του κειμένου.

3.5.2 Προ-επεξεργασία κειμένου

Σε αυτό το στάδιο το προς επεξεργασία κείμενο κάθε πράξης αποτελείται από την ένωση του θέματος της και του κυρίως κειμένου της (πεδία subject και exText του αρχείου JSON).

Ο κώδικας που χρησιμοποιήθηκε για τα πρώτα βήματα της προεπεξεργασίας δηλαδή την αφαίρεση των stopwords, των σημείων στίξης, του τονισμού παρουσιάζεται στο Παράρτημα Δ. Χρησιμοποιήθηκαν οι βιβλιοθήκες NLTK και SpaCy.

Στην παρακάτω εικόνα παρουσιάζεται τμήμα των περιεχομένων του JSON αρχείου στην τελική του μορφή.



Εικόνα 26: Μεταδεδομένα πράξεων στο αρχείο JSON

Διακρίνονται τα μεταδεδομένα που αποθηκεύτηκαν μετά την εξαγωγή κειμένου από το αρχείο PDF. Μεταξύ αυτών είναι τα πεδία που θα μας απασχολήσουν στη συνέχεια:

- **subject, exText:** το θέμα και το κυρίως κείμενο κάθε πράξης
- **themCat1, themCat2:** οι Θεματικές κατηγορίες κάθε πράξης
- **decType:** το Είδος κάθε πράξης

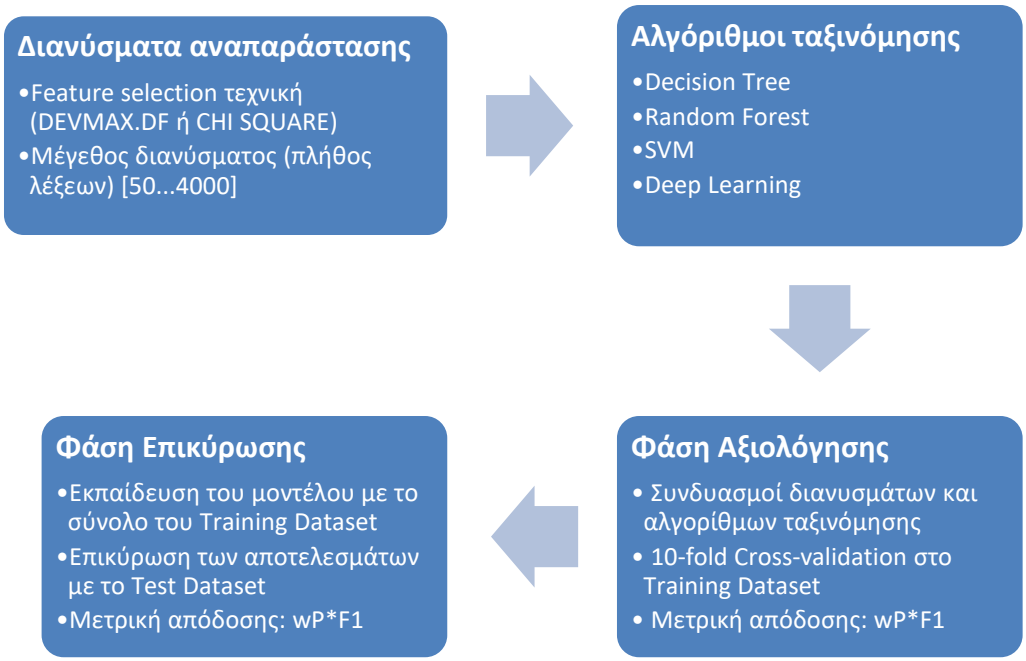
Το αρχικό Training Dataset περιείχε 28,7 εκατομμύρια tokens. Μετά τις ενέργειες της προεπεξεργασίας κειμένου παρέμειναν 252 χιλιάδες μοναδικές λέξεις (καταμετράται η πρώτη εμφάνιση κάθε λέξης και όχι το πλήθος εμφανίσεων της). Στη συνέχεια, με τη διαδικασία δημιουργίας ομάδων λέξεων που αναφέρονται στην επόμενη παράγραφο, καταλήξαμε σε 68 χιλιάδες βασικές λέξεις.

3.5.3 Δημιουργία λεξικών – μετατροπή κειμένου σε διανύσματα

Στη συνέχεια έγινε επεξεργασία και ανάλυση βάσει των τεχνικών επιλογής χαρακτηριστικών (feature selection) για την δημιουργία λεξικών και την δημιουργία διανυσμάτων κειμένου με την τεχνική του Bag-of-Words. Οι τεχνικές επιλογής χαρακτηριστικών που χρησιμοποιήθηκαν είναι οι CHI SQUARE (X^2) και DEVMAX.DF, ενώ δημιουργήθηκαν λεξικά τόσο για την κατηγοριοποίηση βάσει της Θεματικής κατηγορίας όσο και του Είδους πράξης των πράξεων. Ο κώδικας που χρησιμοποιήθηκε για την δημιουργία ομάδων (group) λέξεων (Triantafyllou et al., 2020) και για τη διαδικασία της δημιουργίας των λεξικών, ώστε να παραχθούν τα αρχεία διανυσμάτων ως αναπαράσταση του κειμένου, βασίστηκε σε προγενέστερο κώδικα του επιβλέποντα καθηγητή.

Ο κώδικας στο Παράρτημα Ε ελέγχει την ομοιότητα μεταξύ των λέξεων έτσι ώστε όμοιες λέξεις να τοποθετούνται στην ίδια ομάδα. Τέλος, τμήμα του κώδικα που αφορά στη δημιουργία των λεξικών παρουσιάζεται στο Παράρτημα ΣΤ.

3.6 Εφαρμογή μοντέλων ταξινόμησης - παραμετροποίηση



Εικόνα 27: Διάγραμμα ροής - στάδιο εφαρμογής αλγορίθμων ταξινόμησης

Οι αλγόριθμοι μηχανικής μάθησης που χρησιμοποιήθηκαν ήταν οι Decision Tree, Random Forest και SVM. Επιπλέον χρησιμοποιήθηκε ένα Feedforward Deep Learning μοντέλο.

Τα μεγέθη των διανυσμάτων αναπαράστασης κειμένου που χρησιμοποιήθηκαν καλύπτουν το εύρος από 50 όρους-λέξεις έως 4000 όρους-λέξεις, με διαστήματα ανά 25 (όρους-λέξεις) μέχρι τις 100, ανά 100 μέχρις τις 1000 και ανά 200 μέχρι τις 4000 (κατά περίπτωση τα διανύσματα έφτασαν τις 4400 όρους-λέξεις). Επιλέχθηκε η δυαδική αναπαράσταση του διανύσματος κειμένου προκειμένου να επιτευχθούν τα βέλτιστα δυνατά αποτελέσματα (Vorgia et al., 2017).

```
In [8]: df.head(20)
```

Out[8]:

	feature1-ΕΛΚΕ	feature2-ΤΙΤΛΟ	feature3-ΚΩΔΙΚΟΣ	feature4-ΚΑΤΗΓΟΡΙΑ	feature5-ΜΟΔΥ	feature6-ΥΠΕΥΘΥΝΟΥ	feature7-ΧΡΗΜΑΤΟΔΟΤΗΣΗ	feature8-ΕΡΓΟ	feature9-ΕΥΡΩ	feature10-LOCATION
Ω8ΚΧ46Μ9ΞΗ-1ΗΜ	1	1	1	1	1	0	1	1	1	1
6Σ9246Μ9ΞΗ-6ΥΩ	1	1	1	1	1	0	1	1	1	1
ΩΦΝ646Μ9ΞΗ-6ΧΙ	1	1	1	1	1	0	1	1	1	1
Ω1ΞΙ46Μ9ΞΗ-9ΜΥ	0	0	0	0	0	0	0	0	0	1
69ΒΠ46Μ9ΞΗ-ΨΚ6	1	1	1	0	1	0	1	1	1	1
72ΙΨ46Μ9ΞΗ-ΡΙΩ	1	1	1	1	1	0	1	1	1	1
Ψ2ΟΗ46Μ9ΞΗ-ΓΓ7	1	1	1	0	1	0	1	1	1	1
ΨΜ6Μ46Μ9ΞΗ-ΜΟΚ	0	0	0	0	0	0	1	1	1	1
9ΣΓ946Μ9ΞΗ-ΒΣΤ	1	1	1	0	1	0	1	1	1	1
7ΝΑ746Μ9ΞΗ-Β1Μ	1	0	1	0	1	0	0	1	1	1
7Χ3Ι46Μ9ΞΗ-ΦΧ7	1	1	1	0	1	0	1	1	1	1
Ω28Χ46Μ9ΞΗ-606	1	1	1	0	1	0	1	1	1	1

Εικόνα 28: Τμήμα αρχείου διανύσματος κειμένου - χαρακτηριστικά

Η δυαδική αναπαράσταση χρησιμοποιήθηκε και για το διάνυσμα των κλάσεων (y - prediction target) του Dataset.

Out[21]:

	label1	label2	label3	label4	label5	label6	label7	label8	label9	label10	label11	label12	label13	label14
Ω8KX46M9ΞH-1HM	1	1	0	0	0	0	0	0	0	0	0	0	0	0
6Σ9246M9ΞH-6YΩ	1	1	0	0	0	0	0	0	0	0	0	0	0	0
ΩΦN646M9ΞH-6XI	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Ω1ΞI46M9ΞH-9MY	0	0	1	0	0	0	0	0	0	0	0	0	0	0
69ΒΠ46M9ΞH-ΨK6	1	1	0	0	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
ΩΔΣ346M9ΞH-PBΓ	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Ψ74H46M9ΞH-399	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Εικόνα 29: Τμήμα αρχείου διανύσματος κειμένου - κωδικοποίηση κλάσεων

Για την αξιολόγηση των μοντέλων χρησιμοποιήθηκε η τεχνική 10-fold Cross-validation. Για κάθε μοντέλο, με βάση την απόδοση κάθε κλάσης υπολογίστηκαν μια σειρά μετρικών απόδοσης, Precision, Recall, F_1 καθώς και το γινόμενο Precision επί F_1. Στους υπολογισμούς ελήφθη υπόψιν το πλήθος εμφανίσεων της συγκεκριμένης κλάσης στο δείγμα (weighted).

Για την επιλογή του βέλτιστου συνδυασμού μοντέλου ταξινόμησης, τεχνικής επιλογής χαρακτηριστικών και μεγέθους διανύσματος επιλέχθηκε το σταθμισμένο γινόμενο Precision*F_1 (wP*F_1) καθώς προσφέρει αυξημένη ευαισθησία στην σωστή εκτίμηση των ορθά ταξινομημένων δειγμάτων (Triantafyllou et al., 2020). Επίσης, ελήφθη υπόψιν η απόδοση των αλγορίθμων σε κλάσεις με χαμηλό πλήθος εμφανίσεων στο σύνολο δεδομένων.

Στη συνέχεια τα μοντέλα, διατηρώντας τις ίδιες παραμέτρους (παρουσιάζονται παρακάτω), εκπαιδευτήκαν στο σύνολο του Training Dataset (χωρίς να γίνει χρήση 10-fold Cross-validation) και ελέγχθηκε η απόδοση τους στην ταξινόμηση του άγνωστου συνόλου δεδομένων του Test Dataset (validation). Για κάθε αλγόριθμο επιλέχθηκε ως βασικό (κεντρικό) μέγεθος διανύσματος αυτό που στο προηγούμενο στάδιο είχε τη βέλτιστη απόδοση και πάρθηκαν αποτελέσματα για διανύσματα εύρους $\pm$ 600 λέξεις από το βασικό (ανά 200 λέξεις).

Τμήμα του κώδικα που εκτελεί τις παραπάνω ενέργειες βρίσκεται στο Παράρτημα Ζ.

Παράμετροι αλγορίθμων

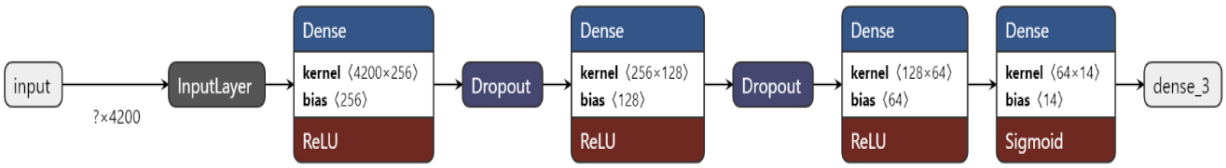
Decision Tree, Random Forest, SVM: χρησιμοποιήθηκαν οι ακόλουθες συναρτήσεις της βιβλιοθήκης Scikit-Learn:

- DecisionTreeClassifier()
- RandomForestClassifier()
- SVC()

Οι παραπάνω συναρτήσεις είχαν τις προεπιλεγμένες παραμέτρους της βιβλιοθήκης.

Deep Learning: επιλέχθηκε η συνάρτηση ενεργοποίησης ReLU στα ενδιάμεσα layer και η συνάρτηση Sigmoid στην έξοδο. Στα dropout layers η τιμή της αναλογίας απόρριψης (dropout rate) είχε τιμή 0,5. Στον compiler επιλέχθηκε ο αλγόριθμος Adam (optimization algorithm).

Στο παρακάτω διάγραμμα παρουσιάζεται η τοπολογία του δικτύου, θεωρώντας ότι στην είσοδο δέχεται διάνυσμα 4200 λέξεων:



Εικόνα 30: Τοπολογία Deep Learning Δικτύου

Κεφάλαιο 4. Αποτελέσματα

4.1 Αποτελέσματα στην ταξινόμηση των πράξεων ως προς την «Θεματική κατηγορία»

4.1.1 [10f-Training1] Evaluation - Training Dataset (22/03/2018 – 21/03/2023)

Στον παρακάτω πίνακα παρουσιάζονται τα καλύτερα αποτελέσματα των ανά ταξινομητή:

Πίνακας 5: Συγκεντρωτικά αποτελέσματα Evaluation (Θεματική κατηγορία) [10f-Training1]

Ταξινομητής	Μέθοδος	Λέξεις (διάνυσμα)	wF_1	wP	$wP*F_1$
Decision Tree	chi square	3200	98,72	98,70	97,50
Random Forest	chi square	3000	99,02	99,12	98,18
SVM	devmax.df	1400	98,79	98,87	97,72
Deep Learning	**devmax.df**	**4200**	**99,48**	**99,47**	**98,98**

Όπως φαίνεται στον παραπάνω πίνακα, η μετρική CHI SQUARE (x2) παρήγαγε τα καλύτερα αποτελέσματα στους ταξινομητές Decision Tree, Random Forest ενώ η μετρική DEVMAX.DF στους ταξινομητές SVM και Deep Learning. Επίσης, η μετρική DEVMAX.DF παρήγαγε τα καλύτερα αποτελέσματα στο σύνολο των ταξινομητών, με την μέγιστη τιμή για την $wP*F_1$ να επιτυγχάνεται για τον Deep Learning ταξινομητή με ποσοστό 98,98%.

Στην πλειοψηφία τους οι ταξινομητές είχαν πρακτικά μηδενική διακριτική ικανότητα στις κλάσεις με τον μικρότερο πληθυσμό.

Στις παραγράφους που ακολουθούν παρουσιάζονται αναλυτικά τα αποτελέσματα για κάθε ένα από τους ταξινομητές δομημένα σε πίνακες ανά μετρική και μέγεθος διανύσματος. Επίσης παρουσιάζονται, ανά ταξινομητή, γραφικές παραστάσεις:

- των τιμών $wP*F_1$,
- των τιμών wF_1 κάθε μιας κλάσης,

για κάθε μετρική και για διαφορετικά μεγέθη διανύσματος.

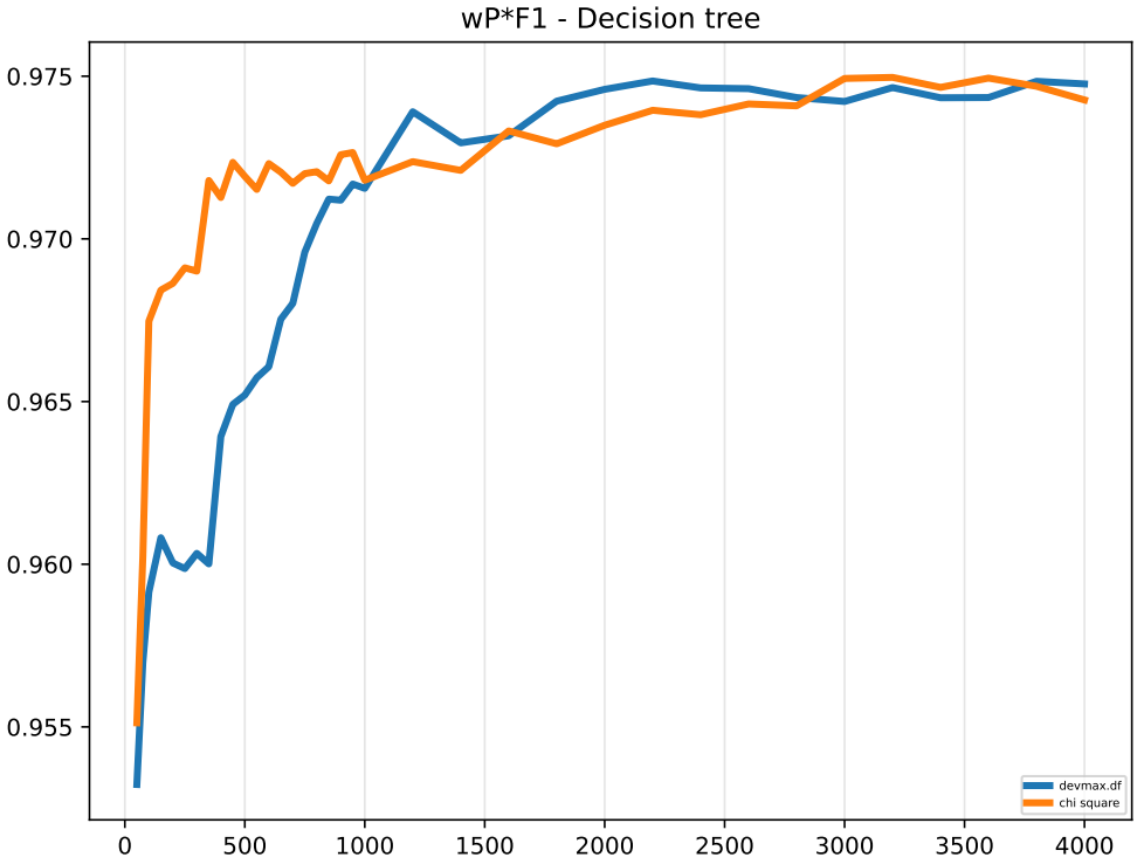
4.1.1.1 Decision Tree [10f-Training1]

Η βέλτιστη απόδοση επιτυγχάνεται για διάνυσμα μεγέθους 3200 και την μετρική CHI SQUARE όπως φαίνεται στον ακόλουθο πίνακα.

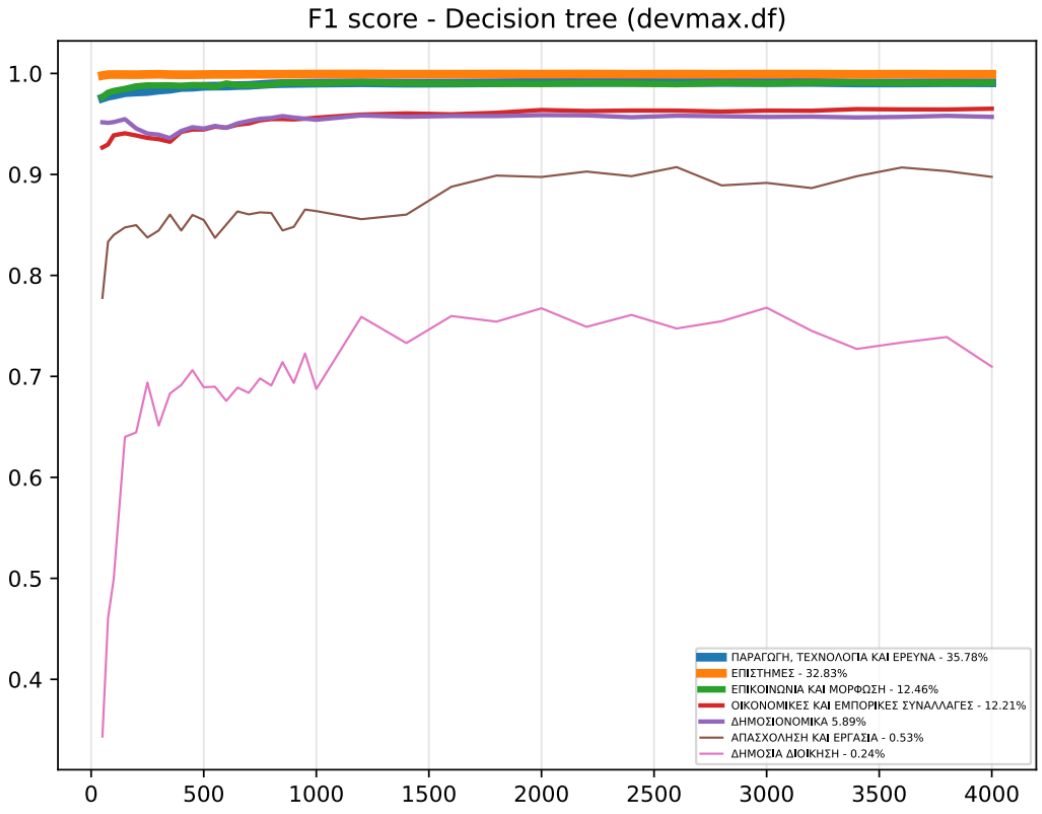
Πίνακας 6. Αποτελέσματα Decision Tree (Θεματική κατηγορία) [10f-Training1]

Λέξεις	DEVMAX.DF				CHI SQUARE			
	wF1	wP	wR	wP*F1	wF1	wP	wR	wP*F1
50	97,29	97,83	96,81	95,32	97,18	98,15	96,35	95,51
75	97,53	98,01	97,11	95,70	97,95	97,92	98,00	96,02
100	97,71	98,05	97,42	95,91	98,24	98,40	98,10	96,75
150	97,91	98,04	97,80	96,08	98,35	98,39	98,32	96,84
200	97,88	97,98	97,80	96,00	98,31	98,45	98,18	96,86
250	97,87	97,98	97,76	95,99	98,38	98,44	98,33	96,91
300	97,90	97,99	97,80	96,03	98,39	98,42	98,36	96,90
350	97,88	97,98	97,78	96,00	98,53	98,56	98,49	97,18
400	98,09	98,18	98,01	96,39	98,51	98,53	98,49	97,13
450	98,17	98,21	98,13	96,49	98,55	98,59	98,52	97,24
500	98,19	98,21	98,18	96,52	98,55	98,57	98,53	97,19
550	98,22	98,23	98,22	96,57	98,51	98,55	98,47	97,15
600	98,24	98,24	98,25	96,61	98,57	98,58	98,55	97,23
650	98,32	98,33	98,31	96,75	98,56	98,56	98,56	97,21
700	98,36	98,34	98,38	96,80	98,56	98,53	98,59	97,17
750	98,43	98,43	98,44	96,96	98,56	98,56	98,57	97,20
800	98,49	98,46	98,51	97,05	98,58	98,54	98,62	97,21
850	98,53	98,50	98,56	97,12	98,57	98,52	98,61	97,18
900	98,52	98,50	98,54	97,12	98,59	98,59	98,59	97,26
950	98,54	98,54	98,55	97,17	98,58	98,60	98,57	97,27
1000	98,54	98,52	98,56	97,16	98,58	98,52	98,64	97,18
1200	98,64	98,66	98,63	97,39	98,58	98,57	98,59	97,24
1400	98,61	98,60	98,62	97,30	98,57	98,55	98,60	97,21
1600	98,63	98,61	98,65	97,32	98,64	98,61	98,68	97,33
1800	98,67	98,68	98,65	97,42	98,62	98,59	98,64	97,29
2000	98,72	98,67	98,77	97,46	98,64	98,63	98,65	97,35
2200	98,71	98,71	98,71	97,49	98,67	98,65	98,68	97,40
2400	98,70	98,70	98,70	97,46	98,65	98,66	98,64	97,38
2600	98,70	98,70	98,70	97,46	98,68	98,66	98,72	97,41
2800	98,69	98,67	98,71	97,43	98,68	98,65	98,71	97,41
3000	98,69	98,65	98,74	97,42	98,72	98,70	98,75	97,49
3200	98,70	98,69	98,71	97,47	**98,72**	**98,70**	98,75	**97,50**
3400	98,69	98,66	98,73	97,43	98,70	98,70	98,70	97,47
3600	98,70	98,66	98,74	97,43	98,71	98,71	98,71	97,49
3800	98,71	98,70	98,72	97,48	98,70	98,69	98,71	97,47
4000	98,70	98,70	98,71	97,48	98,69	98,66	98,72	97,43

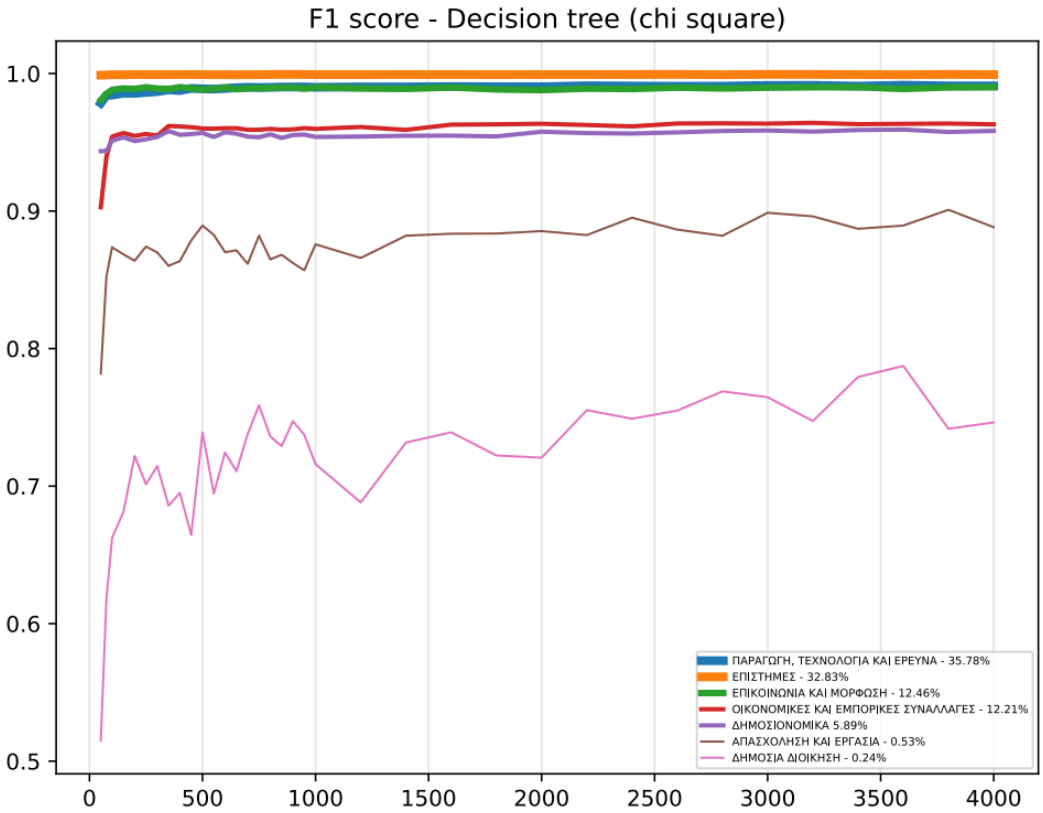
Στις παρακάτω εικόνες εμφανίζονται το συγκριτικό διάγραμμα wP*F_1 για τις δυο μετρικές και τα διαγράμματα του F_1 score για κάθε μια Θεματική κατηγορία (κλάση) και κάθε μια από τις δυο μετρικές.



Εικόνα 31: Διάγραμμα wP*F_1 - Decision Tree (Θεματική κατηγορία) [10f-Training1]



Εικόνα 32: : F_1 score ανά Θεματική κατηγορία (DEVMAX.DF - Decision Tree) [10f-Training1]



Εικόνα 33: F_1 score ανά Θεματική κατηγορία (Chi square - Decision Tree) [10f-Training1]

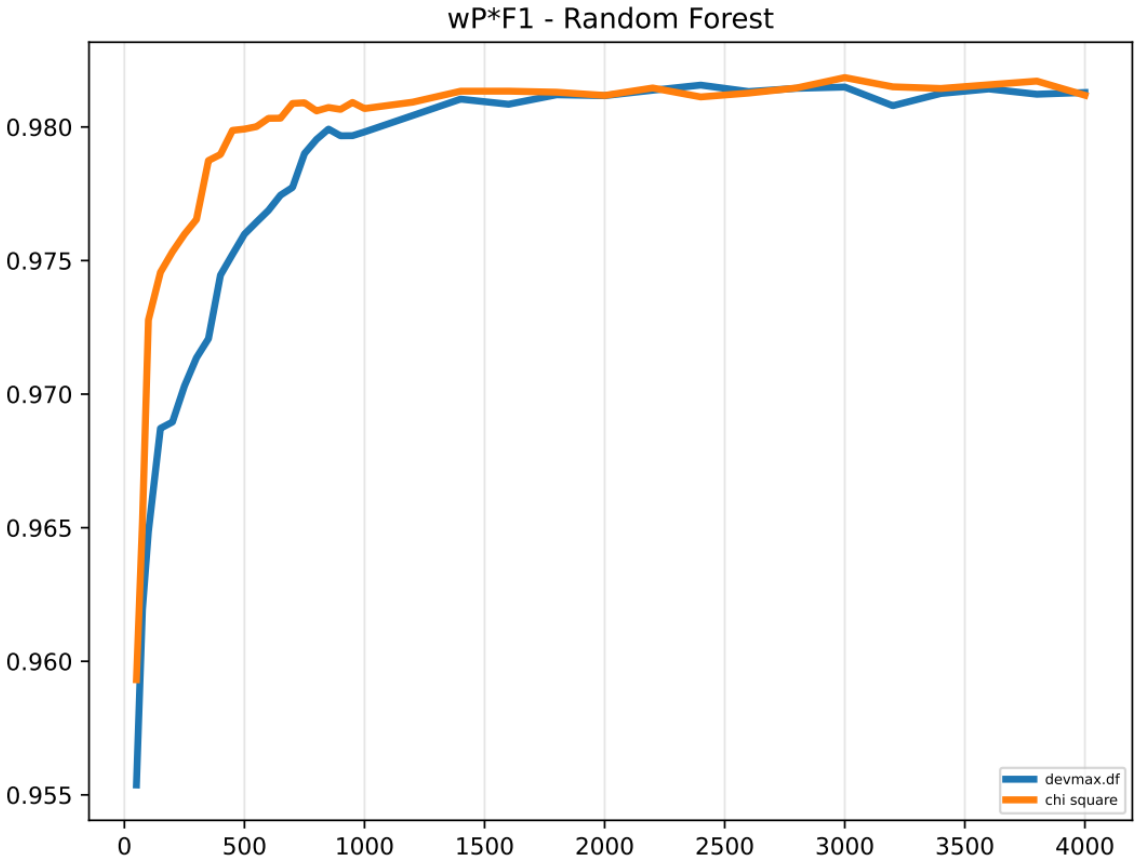
4.1.1.2 Random Forest [10f-Training1]

Η βέλτιστη απόδοση επιτυγχάνεται για διάνυσμα μεγέθους 3000 και την μετρική CHI SQUARE όπως φαίνεται στον ακόλουθο πίνακα.

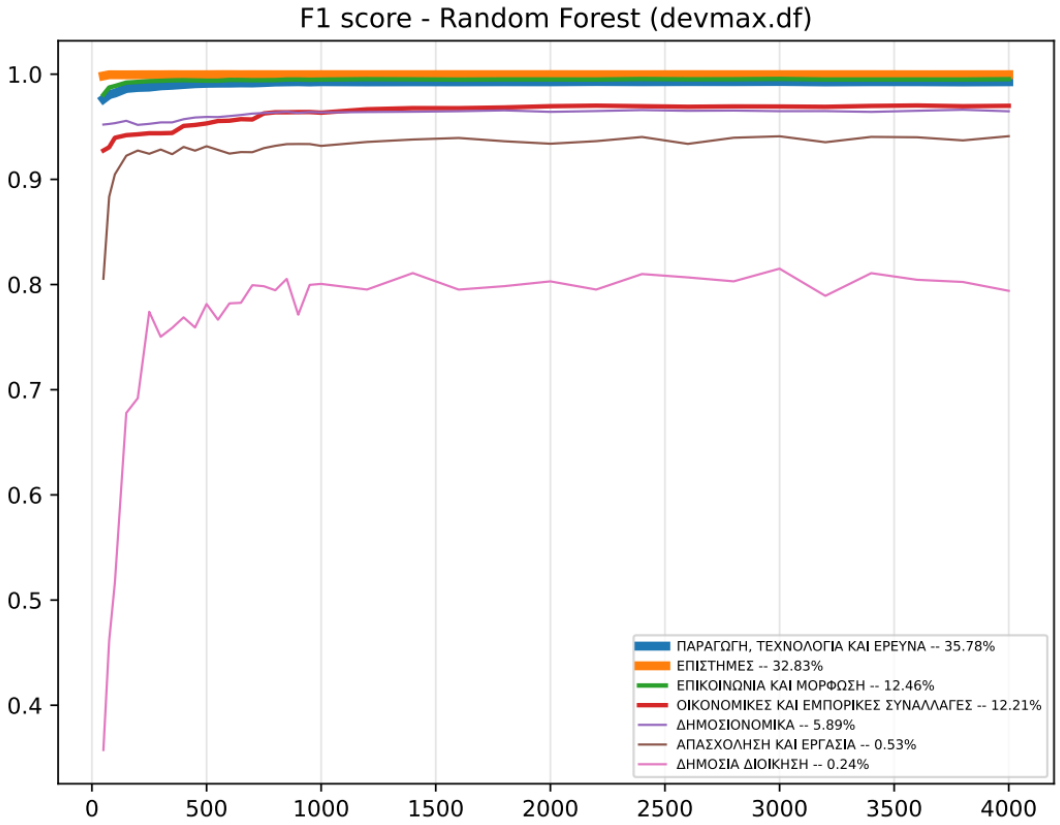
Πίνακας 7: Αποτελέσματα Random Forest (Θεματική κατηγορία) [10f-Training1]

Λέξεις	DEVMAX.DF				CHI SQUARE			
	wF1	wP	wR	wP*F1	wF1	wP	wR	wP*F1
50	97,41	97,98	96,96	95,54	97,41	98,39	96,59	95,93
75	97,78	98,29	97,38	96,19	98,21	98,19	98,27	96,52
100	98,00	98,40	97,68	96,49	98,53	98,68	98,40	97,28
150	98,28	98,50	98,10	96,87	98,66	98,73	98,60	97,46
200	98,32	98,48	98,18	96,90	98,68	98,79	98,59	97,53
250	98,38	98,57	98,21	97,03	98,75	98,79	98,72	97,60
300	98,43	98,62	98,25	97,14	98,78	98,82	98,74	97,65
350	98,46	98,67	98,27	97,21	98,89	98,94	98,84	97,87
400	98,60	98,78	98,42	97,45	98,89	98,96	98,82	97,90
450	98,64	98,82	98,47	97,52	98,94	99,00	98,89	97,99
500	98,67	98,86	98,50	97,60	98,94	99,01	98,88	97,99
550	98,71	98,88	98,55	97,64	98,94	99,02	98,87	98,00
600	98,73	98,90	98,57	97,69	98,95	99,03	98,88	98,03
650	98,76	98,93	98,60	97,74	98,96	99,02	98,92	98,03
700	98,76	98,96	98,58	97,77	98,99	99,05	98,93	98,09
750	98,85	99,00	98,72	97,90	98,98	99,07	98,90	98,09
800	98,89	99,02	98,77	97,95	98,97	99,04	98,91	98,06
850	98,91	99,04	98,78	97,99	98,98	99,05	98,91	98,07
900	98,90	99,02	98,79	97,97	98,97	99,05	98,90	98,07
950	98,90	99,02	98,79	97,97	98,98	99,06	98,91	98,09
1000	98,90	99,03	98,78	97,98	98,97	99,05	98,90	98,07
1200	98,94	99,05	98,84	98,04	98,98	99,07	98,91	98,09
1400	98,96	99,10	98,84	98,10	99,00	99,09	98,92	98,13
1600	98,96	99,08	98,84	98,09	99,00	99,09	98,92	98,13
1800	98,98	99,10	98,86	98,12	99,00	99,09	98,92	98,13
2000	98,98	99,09	98,88	98,12	98,99	99,08	98,91	98,12
2200	99,00	99,09	98,91	98,14	99,00	99,10	98,92	98,15
2400	99,00	99,11	98,90	98,16	98,98	99,08	98,89	98,11
2600	98,99	99,10	98,89	98,13	98,99	99,09	98,90	98,13
2800	98,99	99,11	98,89	98,14	99,00	99,10	98,91	98,15
3000	99,00	99,10	98,91	98,15	**99,02**	**99,12**	98,93	**98,18**
3200	98,97	99,07	98,88	98,08	99,01	99,10	98,92	98,15
3400	98,99	99,09	98,90	98,13	99,00	99,10	98,91	98,14
3600	99,00	99,10	98,90	98,14	99,00	99,11	98,91	98,16
3800	98,98	99,09	98,89	98,12	99,01	99,12	98,91	98,17
4000	98,99	99,09	98,90	98,13	98,98	99,09	98,89	98,12

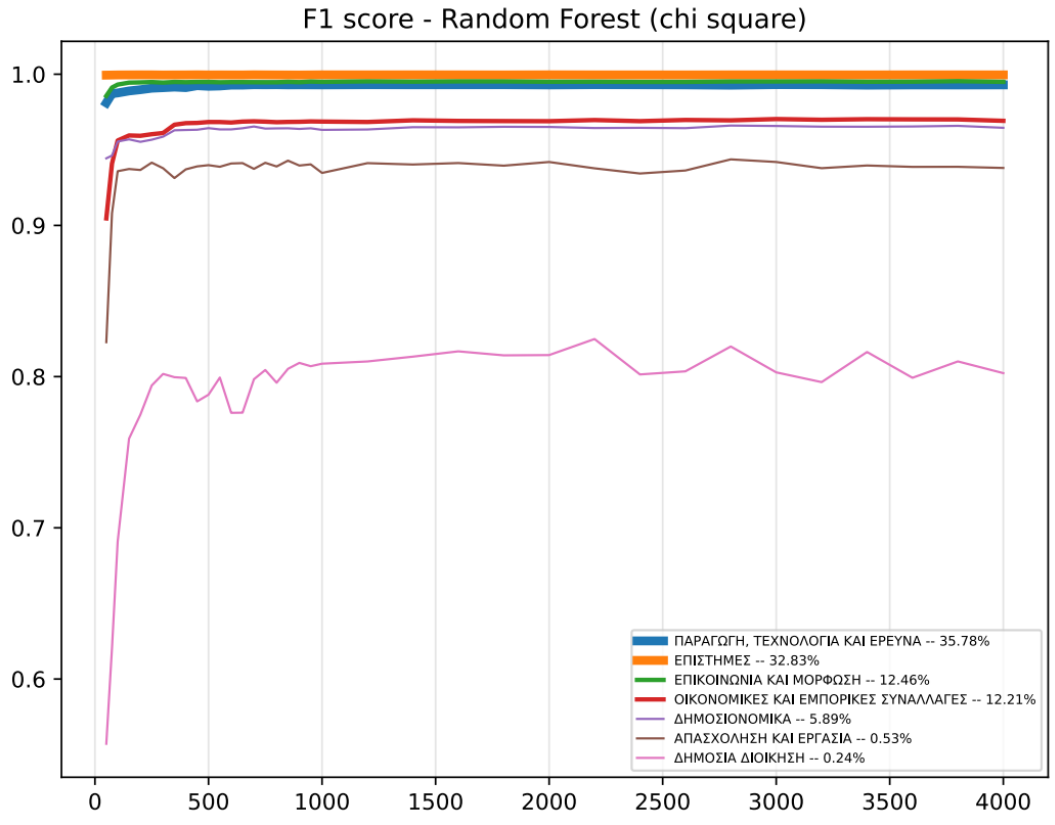
Στις παρακάτω εικόνες εμφανίζονται το συγκριτικό διάγραμμα wP*F_1 για τις δυο μετρικές και τα διαγράμματα του F_1 score για κάθε μια Θεματική κατηγορία (κλάση) και κάθε μια από τις δυο μετρικές.



Εικόνα 34: Διάγραμμα wP*F_1 - Random Forest (Θεματική κατηγορία) [10f-Training1]



Εικόνα 35: F_1 score ανά Θεματική κατηγορία (DEVMAX.DF – Random Forest) [10f-Training1]



Εικόνα 36: F_1 score ανά Θεματική κατηγορία (Chi square – Random Forest) [10f-Training1]

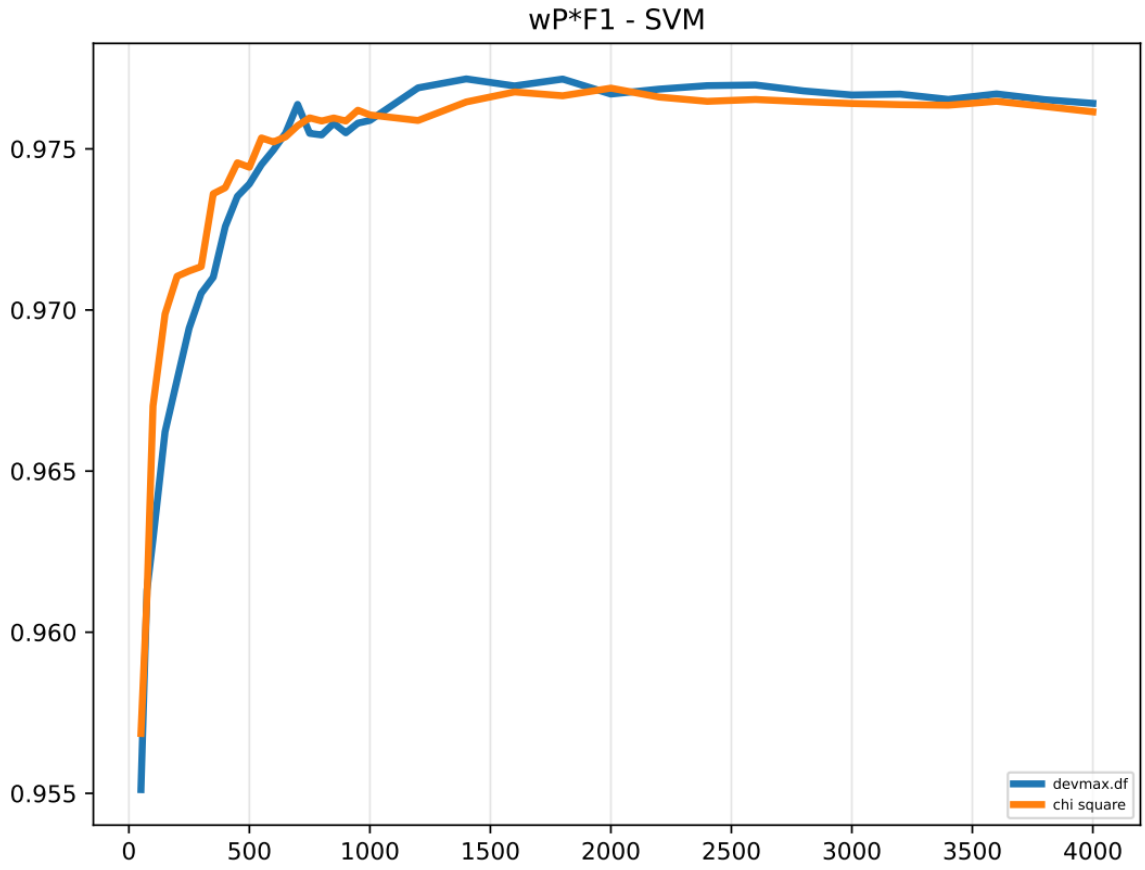
4.1.1.3 SVM [10f-Training1]

Η βέλτιστη απόδοση επιτυγχάνεται για διάνυσμα μεγέθους 1400 και την μετρική DEVMAX.DF όπως φαίνεται στον ακόλουθο πίνακα.

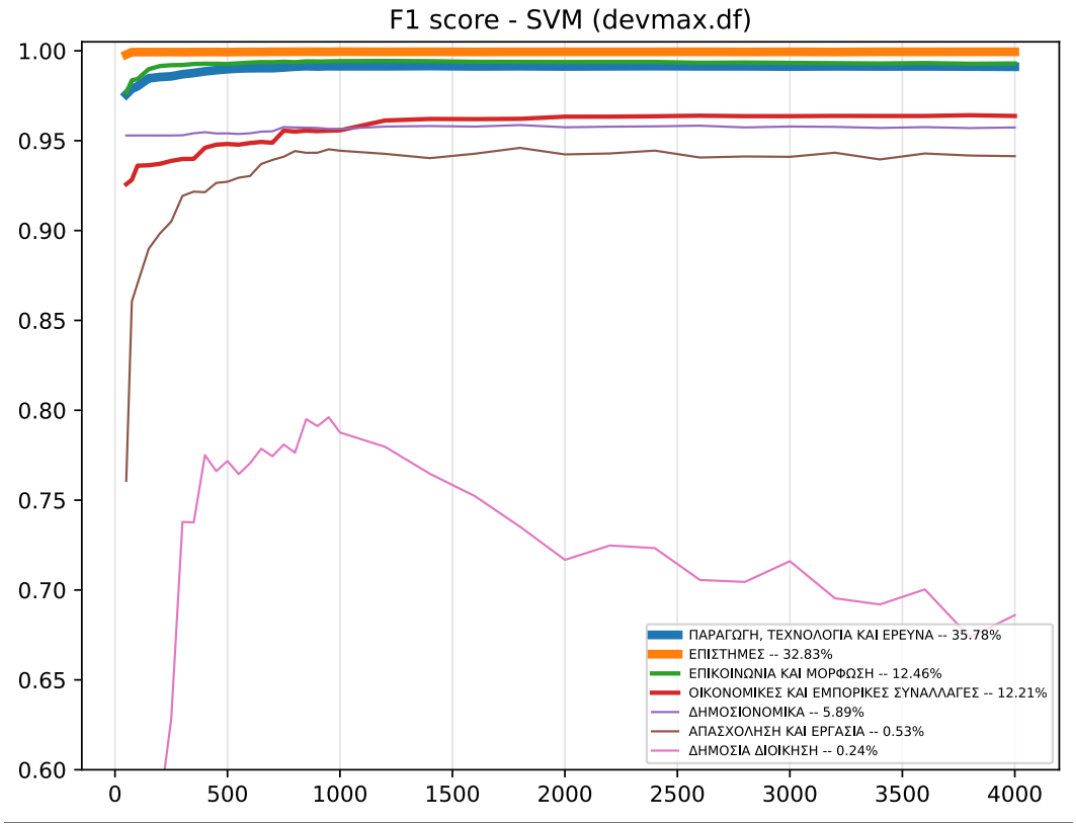
Πίνακας 8: Αποτελέσματα SVM (Θεματική κατηγορία) [10f-Training1]

Λέξεις	DEVMAX.DF				CHI SQUARE			
	wF1	wP	wR	wP*F1	wF1	wP	wR	wP*F1
50	97,21	98,14	96,54	95,51	97,03	98,38	95,96	95,69
75	97,58	98,45	96,93	96,13	97,97	98,02	98,04	96,11
100	97,75	98,43	97,25	96,29	98,26	98,35	98,24	96,70
150	98,04	98,49	97,70	96,62	98,34	98,56	98,16	96,99
200	98,13	98,57	97,77	96,78	98,45	98,57	98,36	97,10
250	98,18	98,68	97,76	96,94	98,53	98,51	98,56	97,12
300	98,27	98,70	97,90	97,05	98,55	98,51	98,60	97,13
350	98,32	98,71	97,97	97,10	98,63	98,66	98,61	97,36
400	98,44	98,75	98,17	97,26	98,63	98,68	98,59	97,38
450	98,48	98,81	98,19	97,35	98,69	98,71	98,68	97,46
500	98,50	98,83	98,22	97,39	98,68	98,69	98,68	97,44
550	98,52	98,87	98,21	97,45	98,71	98,76	98,68	97,53
600	98,54	98,89	98,24	97,50	98,71	98,75	98,67	97,52
650	98,57	98,92	98,25	97,55	98,71	98,76	98,68	97,54
700	98,56	99,02	98,16	97,64	98,73	98,78	98,69	97,57
750	98,68	98,81	98,57	97,55	98,74	98,79	98,70	97,60
800	98,68	98,79	98,60	97,54	98,73	98,79	98,69	97,59
850	98,71	98,81	98,64	97,58	98,74	98,80	98,69	97,60
900	98,70	98,79	98,63	97,55	98,73	98,79	98,69	97,59
950	98,71	98,81	98,62	97,58	98,75	98,81	98,70	97,62
1000	98,71	98,82	98,62	97,59	98,74	98,80	98,71	97,61
1200	98,78	98,85	98,72	97,69	98,74	98,79	98,70	97,59
1400	**98,79**	**98,87**	98,73	**97,72**	98,77	98,81	98,76	97,65
1600	98,77	98,87	98,70	97,70	98,78	98,84	98,75	97,68
1800	98,78	98,88	98,70	97,72	98,77	98,83	98,74	97,66
2000	98,77	98,84	98,74	97,67	98,79	98,83	98,77	97,69
2200	98,78	98,84	98,75	97,69	98,78	98,82	98,76	97,66
2400	98,79	98,85	98,76	97,70	98,77	98,82	98,75	97,65
2600	98,78	98,86	98,74	97,70	98,78	98,81	98,77	97,65
2800	98,77	98,85	98,72	97,68	98,77	98,81	98,76	97,65
3000	98,77	98,84	98,74	97,67	98,77	98,81	98,75	97,64
3200	98,77	98,84	98,73	97,67	98,77	98,81	98,75	97,64
3400	98,76	98,83	98,72	97,65	98,77	98,81	98,75	97,64
3600	98,77	98,84	98,73	97,67	98,77	98,81	98,76	97,65
3800	98,76	98,83	98,72	97,65	98,77	98,81	98,75	97,63
4000	98,76	98,83	98,72	97,64	98,76	98,80	98,74	97,62

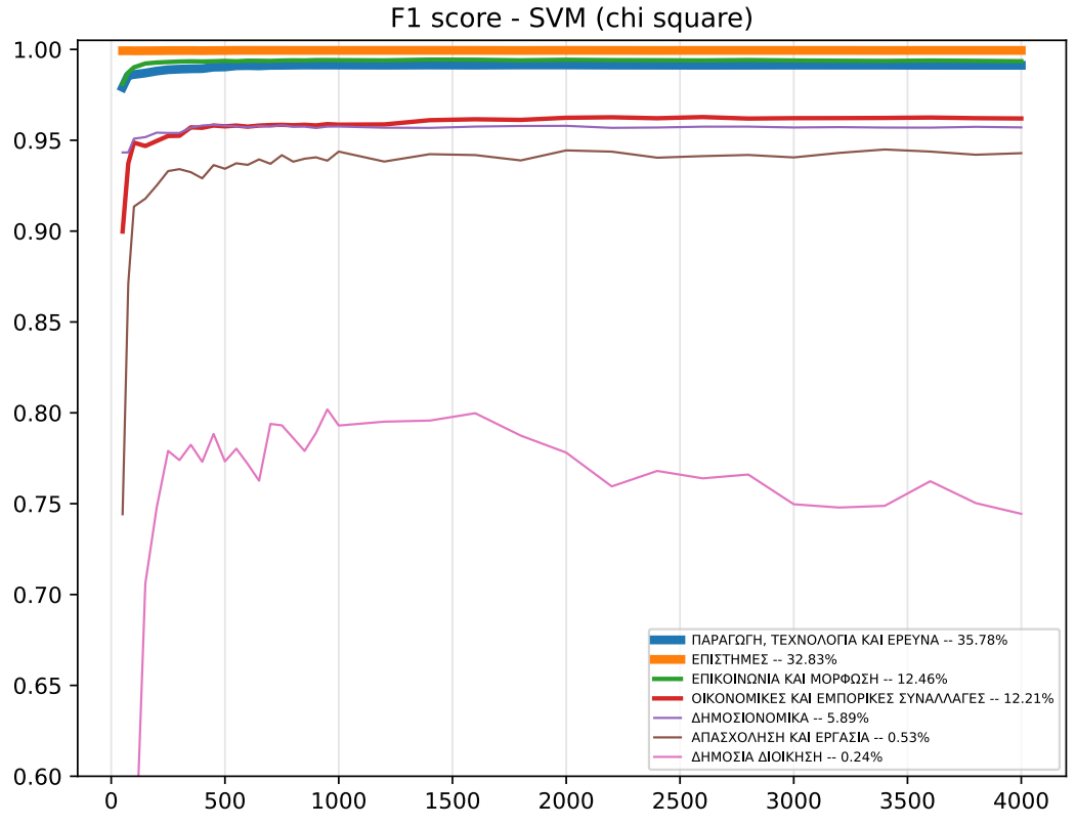
Στις παρακάτω εικόνες εμφανίζονται το συγκριτικό διάγραμμα wP*F1 για τις δυο μετρικές και τα διαγράμματα του F_1 score για κάθε μια Θεματική κατηγορία (κλάση) και κάθε μια από τις δυο μετρικές.



Εικόνα 37: Διάγραμμα wP*F_1 – SVM (Θεματική κατηγορία) [10f-Training1]



Εικόνα 38: F_1 score ανά Θεματική κατηγορία (DEVMAX.DF - SVM) [10f-Training1]



Εικόνα 39: F_1 score ανά Θεματική κατηγορία (Chi square – SVM) [10f-Training1]

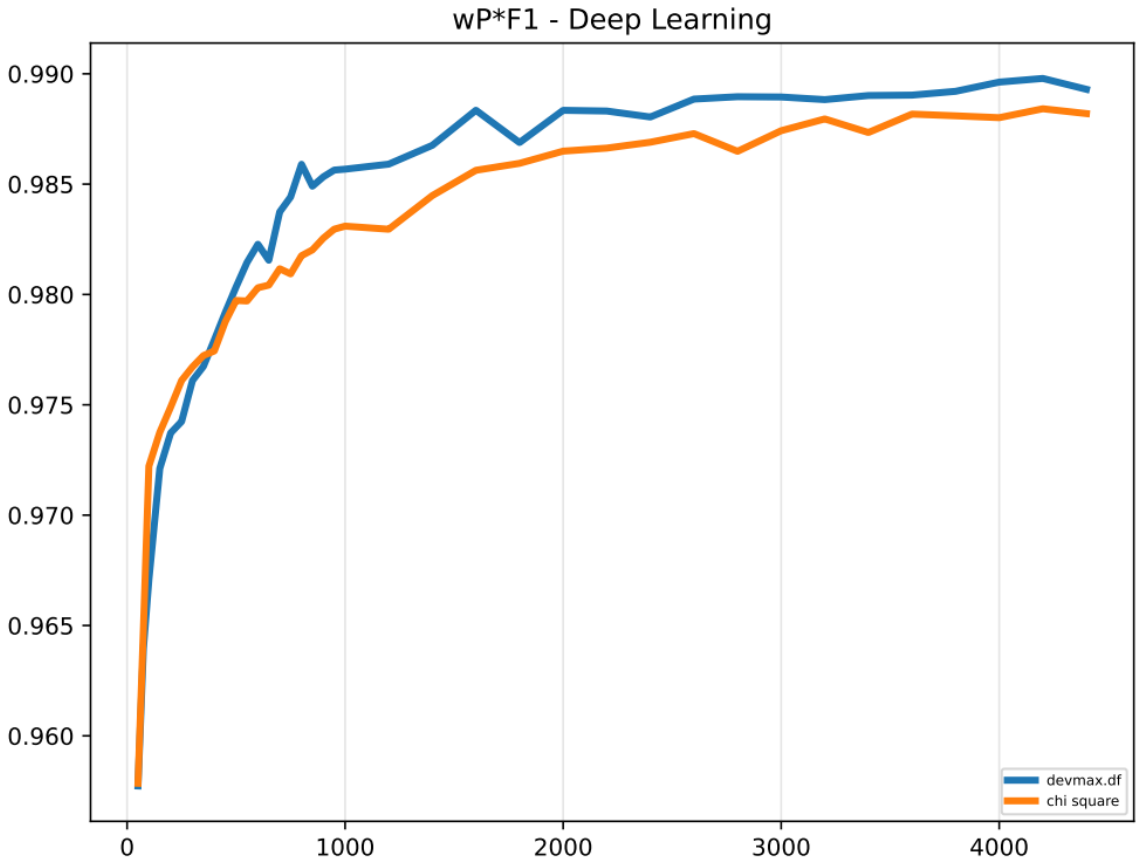
4.1.1.4 Deep Learning [10f-Training1]

Η βέλτιστη απόδοση επιτυγχάνεται για διάνυσμα μεγέθους 4200 και την μετρική DEVMAX.DF όπως φαίνεται στον ακόλουθο πίνακα.

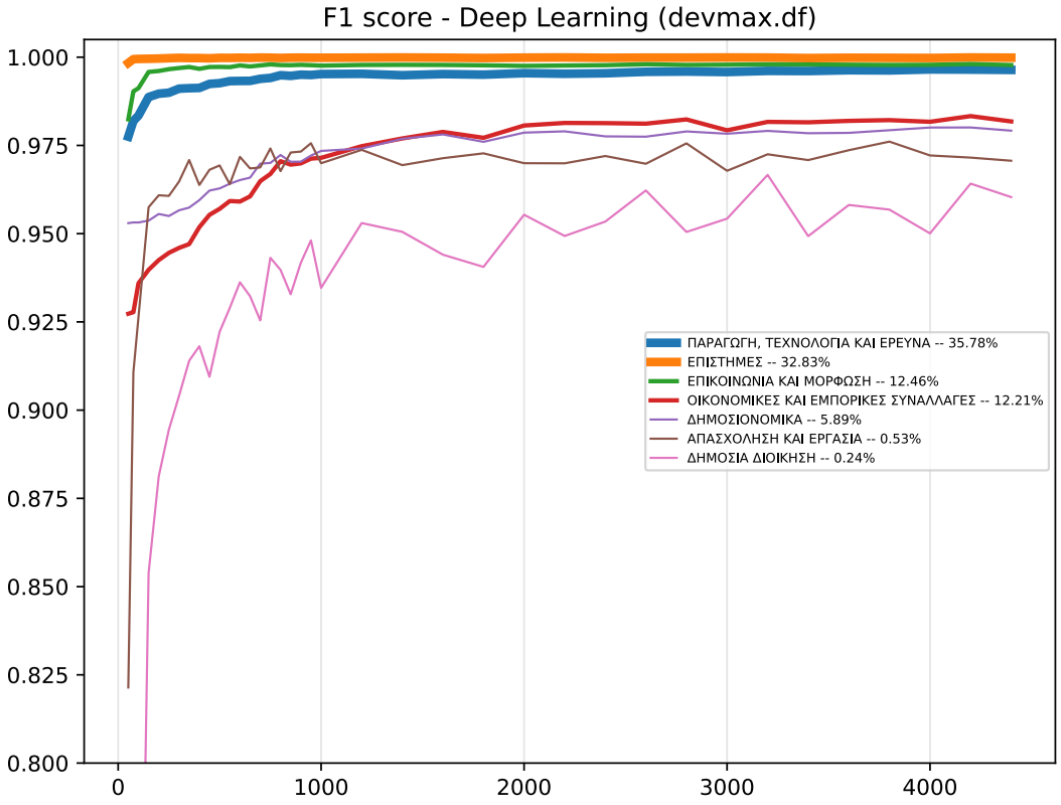
Πίνακας 9: Αποτελέσματα Deep Learning (Θεματική κατηγορία) [10f-Training1]

Λέξεις	DEVMAX.DF				CHI SQUARE			
	wF1	wP	wR	wP*F1	wF1	wP	wR	wP*F1
50	97,50	98,14	96,98	95,77	97,31	98,32	96,52	95,78
75	97,90	98,41	97,48	96,40	98,20	98,17	98,27	96,49
100	98,09	98,54	97,71	96,71	98,51	98,64	98,39	97,22
150	98,45	98,69	98,24	97,21	98,61	98,69	98,55	97,38
200	98,54	98,77	98,34	97,37	98,68	98,74	98,64	97,49
250	98,58	98,77	98,42	97,42	98,77	98,78	98,77	97,61
300	98,66	98,89	98,46	97,61	98,79	98,81	98,79	97,67
350	98,69	98,92	98,48	97,67	98,85	98,81	98,91	97,72
400	98,75	98,99	98,54	97,79	98,89	98,80	98,98	97,74
450	98,86	99,01	98,73	97,92	98,94	98,88	99,01	97,88
500	98,90	99,09	98,73	98,03	98,96	98,97	98,96	97,97
550	98,95	99,15	98,77	98,14	98,99	98,93	99,05	97,97
600	98,97	99,21	98,74	98,23	99,01	98,97	99,05	98,03
650	98,99	99,12	98,86	98,16	99,00	98,99	99,02	98,04
700	99,09	99,25	98,94	98,37	99,05	99,02	99,09	98,12
750	99,13	99,27	99,01	98,44	99,03	99,01	99,07	98,09
800	99,21	99,35	99,08	98,59	99,07	99,07	99,08	98,18
850	99,18	99,27	99,10	98,49	99,09	99,07	99,11	98,20
900	99,20	99,29	99,12	98,53	99,09	99,12	99,07	98,25
950	99,23	99,30	99,16	98,56	99,09	99,16	99,03	98,30
1000	99,24	99,29	99,19	98,57	99,13	99,14	99,12	98,31
1200	99,30	99,26	99,34	98,59	99,17	99,09	99,25	98,30
1400	99,32	99,32	99,33	98,68	99,23	99,18	99,29	98,45
1600	99,36	99,44	99,28	98,83	99,26	99,27	99,25	98,56
1800	99,32	99,34	99,31	98,69	99,26	99,30	99,23	98,59
2000	99,40	99,41	99,39	98,83	99,31	99,31	99,31	98,65
2200	99,40	99,40	99,41	98,83	99,32	99,31	99,35	98,66
2400	99,40	99,38	99,42	98,80	99,34	99,32	99,36	98,69
2600	99,42	99,44	99,40	98,89	99,37	99,33	99,41	98,73
2800	99,44	99,43	99,46	98,90	99,34	99,28	99,40	98,65
3000	99,39	99,47	99,32	98,89	99,36	99,35	99,38	98,74
3200	99,45	99,41	99,48	98,88	99,36	99,40	99,33	98,80
3400	99,43	99,45	99,41	98,90	99,37	99,33	99,41	98,73
3600	99,45	99,43	99,47	98,90	99,38	99,41	99,35	98,82
3800	99,45	99,44	99,46	98,92	99,39	99,39	99,38	98,81
4000	99,46	99,48	99,44	98,96	99,40	99,37	99,45	98,80
4200	**99,48**	**99,47**	99,50	**98,98**	99,41	99,40	99,43	98,84
4400	99,45	99,45	99,46	98,93	99,40	99,39	99,42	98,82

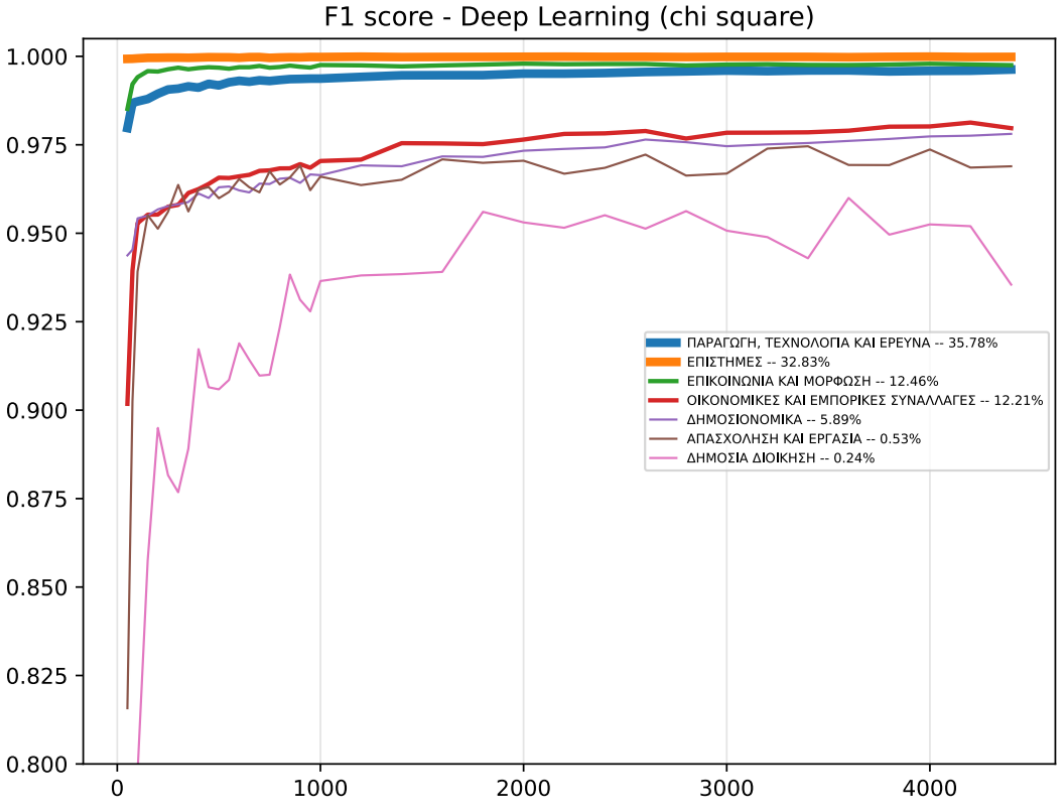
Στις παρακάτω εικόνες εμφανίζονται το συγκριτικό διάγραμμα wP*F_1 για τις δυο μετρικές και τα διαγράμματα του F_1 score για κάθε μια Θεματική κατηγορία (κλάση) και κάθε μια από τις δυο μετρικές.



Εικόνα 40: Διάγραμμα wP*F_1 – Deep Learning (Θεματική κατηγορία) [10f-Training1]



Εικόνα 41: F_1 score ανά Θεματική κατηγορία (DEVMAX.DF - Deep Learning) [10f-Training1]



Εικόνα 42: F_1 score ανά Θεματική κατηγορία (Chi square - Deep Learning) [10f-Training1]

4.1.2 [Val-Training1Test1] Validation: Training Dataset (22/03/2018 – 21/03/2023) => Test Dataset (22/03/2023 – 23/06/2023)

Στον παρακάτω πίνακα εμφανίζονται τα αποτελέσματα που πέτυχε κάθε ταξινομητής στις μετρικές wF_1, wP και $wP*F_1$ κατά τη φάση της επικύρωσης (validation) στην κατηγοριοποίηση των άγνωστων δεδομένων του Test Dataset. Τα αποτελέσματα είναι πολύ κοντά σε αυτά που παρήχθησαν κατά την διαδικασία αξιολόγησης (evaluation phase).

Η μετρική DEVMAX.DF στον Deep Learning ταξινομητή πέτυχε την υψηλότερη απόδοση με $wP*F_1$ = 97,66% για διάνυσμα 4200.

Πίνακας 10: Συγκριτικά αποτελέσματα φάσεων ταξινόμησης (Θεματική κατηγορία) [Val-Training1Test1]

			Evaluation Phase Training Dataset (22/03/2018 – 21/03/2023)			**Validation Phase Training => Test Dataset (22/03/2023 – 23/06/2023)**		
Ταξινομητής	**Μέθοδος**	**Λέξεις (διάνυσμα)**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**
Decision Tree	chi square	3200	98,72	98,70	**97,50**	98,02	98,23	**96,43**
Random Forest	chi square	3000	99,02	99,12	**98,18**	98,52	98,54	**97,58**
SVM	devmax.df	1400	98,79	98,87	**97,72**	98,63	98,59	**97,32**
Deep Learning	devmax.df	4200	99,48	99,47	**98,98**	98,66	98,87	**97,66**

Οι ταξινομητές SVM και Random Forest κατά τη διαδικασία του evaluation πέτυχαν υψηλότερες αποδόσεις σε διαφορετικά διανύσματα από αυτές που αναγράφονται στον παραπάνω πίνακα. Συγκεκριμένα, ο ταξινομητής SVM με $wP*F_1$ = 97,43% για διάνυσμα 2000 ενώ ο ταξινομητής Random Forest με $wP*F_1$ = 97,59% για διάνυσμα 2800.

Στις επόμενες ενότητες παρουσιάζονται αναλυτικά τα αποτελέσματα ανά μέθοδο.

4.1.2.1 Decision Tree [Val- Training1Test1]

Λέξεις	**CHI SQUARE**			
	wF1	**wP**	**wR**	**wP*F1**
2600	97,73	97,76	97,74	95,69
2800	97,50	97,05	98,08	94,96
3000	97,79	97,77	97,94	95,89
3200	98,02	98,23	97,85	96,43
3400	97,74	97,68	97,96	95,77
3600	97,96	98,24	97,75	96,39
3800	97,93	98,20	97,73	96,32

4.1.2.2 Random Forest [Val- Training1Test1]

Λέξεις	CHI SQUARE			
	wF1	wP	wR	wP*F1
2400	98,46	98,92	98,19	97,49
2600	98,41	98,77	98,18	97,37
2800	98,56	98,95	98,31	97,59
3000	98,52	98,94	98,26	97,58
3200	98,43	98,95	98,14	97,49
3400	98,39	98,80	98,13	97,34
3600	98,48	98,93	98,24	97,52

4.1.2.3 SVM [Val- Training1Test1]

Λέξεις	DEVMAX.DF			
	wF1	wP	wR	wP*F1
800	98,55	98,57	98,61	97,25
1000	98,59	98,61	98,67	97,32
1200	98,62	98,62	98,71	97,34
1400	98,63	98,59	98,76	97,32
1600	98,63	98,62	98,74	97,36
1800	98,67	98,63	98,80	97,39
2000	98,69	98,64	98,81	97,43

4.1.2.4 Deep Learning [Val- Training1Test1]

Λέξεις	DEVMAX.DF			
	wF1	wP	wR	wP*F1
3600	98,69	98,72	98,71	97,53
3800	98,64	98,87	98,47	97,62
4000	98,72	98,68	98,83	97,49
4200	98,66	98,87	98,57	97,66
4400	98,76	98,62	98,95	97,46

4.1.3 Επιπλέον ενέργειες – Βήματα

Στη συνέχεια παρουσιάζονται τα αποτελέσματα που προέκυψαν μετά τις ενέργειες που περιγράφονται στην παράγραφο 0.

4.1.3.1 Μείωση του εύρους της χρονικής περιόδου κατά ένα μήνα [Val- Training1Test2]

Πίνακας 11: Συγκριτικά αποτελέσματα - μείωση του εύρους της χρονικής περιόδου του Test Dataset κατά ένα μήνα (Θεματική κατηγορία) [Val- Training1Test2]

			Evaluation Phase			Validation Phase					
			Training Dataset (22/03/2018 – 21/03/2023)			Training => Test Dataset 22/03/2023 – 23/06/2023			Training => Test Dataset 22/03/2023 – 23/05/2023		
Ταξινομητής	**Μέθοδος**	**Λέξεις (διάνυσμα)**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**
Decision Tree	chi square	3200	98,72	98,70	**97,50**	98,02	98,23	**96,43**	93,30	90,81	**85,89**
Random Forest	chi square	3000	99,02	99,12	**98,18**	98,52	98,54	**97,58**	98,17	98,73	**97,01**
SVM	devmax.df	1400	98,79	98,87	**97,72**	98,63	98,59	**97,32**	97,70	98,43	**96,24**
Deep Learning	devmax.df	4200	99,48	99,47	**98,98**	98,66	98,87	**97,66**	97,30	98,32	**95,73**

4.1.3.2 Μείωση του εύρους της χρονικής περιόδου κατά δυο μήνες [Val-Training1Test3]

Πίνακας 12: Συγκριτικά αποτελέσματα - μείωση του εύρους της χρονικής περιόδου του Test Dataset κατά δυο μήνες (Θεματική κατηγορία) [Val- Training1Test3]

			Evaluation Phase			Validation Phase					
			Training Dataset (22/03/2018 – 21/03/2023)			Training => Test Dataset 22/03/2023 – 23/06/2023			Training => Test Dataset 22/03/2023 – 30/04/2023		
Ταξινομητής	**Μέθοδος**	**Λέξεις (διάνυσμα)**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**
Decision Tree	chi square	3200	98,72	98,70	**97,50**	98,02	98,23	**96,43**	93,26	90,20	**85,27**
Random Forest	chi square	3000	99,02	99,12	**98,18**	98,52	98,54	**97,58**	98,09	98,39	**96,69**
SVM	devmax.df	1400	98,79	98,87	**97,72**	98,63	98,59	**97,32**	97,42	98,37	**95,90**
Deep Learning	devmax.df	4200	99,48	99,47	**98,98**	98,66	98,87	**97,66**	97,48	98,79	**96,34**

4.1.3.3 Νέα Training και Test Datasets [10f-Training2] [Val- Training2Test4]

Πίνακας 13: Συγκριτικά αποτελέσματα - νέα Training και Test Datasets (Θεματική κατηγορία) [10f-Training2] [Val- Training2Test4]

			Evaluation Phase Training Dataset (22/03/2018 – 23/06/2023)			Validation Phase Training => Test Dataset (24/06/2023 – 23/08/2023)		
Ταξινομητής	**Μέθοδος**	**Λέξεις (διάνυσμα)**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**
Decision Tree	chi square	3200	98,70	98,66	**97,44**	97,70	98,31	**96,30**
Random Forest	chi square	3000	99,02	99,11	**98,18**	97,17	99,21	**96,42**
SVM	devmax.df	1400	98,79	98,86	**97,71**	96,88	98,42	**95,39**
Deep Learning	devmax.df	4200	99,44	99,42	**98,89**	98,74	98,50	**97,32**

4.2 Αποτελέσματα στην ταξινόμηση των πράξεων ως προς το «Είδος πράξης»

4.2.1 [10f-Training1] Evaluation - Training Dataset (22/03/2018 – 21/03/2023)

Στον παρακάτω πίνακα παρουσιάζονται τα καλύτερα αποτελέσματα των ανά ταξινομητή:

Πίνακας 14: Συγκεντρωτικά αποτελέσματα Evaluation phase (Είδος πράξης) [10f-Training1]

Ταξινομητής	Μέθοδος	Λέξεις (διάνυσμα)	wF_1	wP	$wP*F_1$
Decision Tree	chi square	1800	98,39	98,39	96,92
Random Forest	chi square	1600	98,94	99,15	98,16
SVM	devmax.df	2600	98,70	98,72	97,71
Deep Learning	**devmax.df**	**3400**	**99,54**	**99,55**	**99,12**

Όπως φαίνεται στον παραπάνω πίνακα, η μετρική CHI SQUARE (x^2) παρήγαγε τα καλύτερα αποτελέσματα στους ταξινομητές Decision Tree, Random Forest ενώ η μετρική DEVMAX.DF στους ταξινομητές SVM και Deep Learning. Επίσης, η μετρική DEVMAX.DF παρήγαγε τα καλύτερα αποτελέσματα μεταξύ των ταξινομητών, με την μέγιστη τιμή για την $wP*F_1$ να επιτυγχάνεται για τον Deep Learning ταξινομητή με ποσοστό 99,12% και διάνυσμα 3400.

Στην πλειοψηφία τους οι αλγόριθμοι είχαν πρακτικά μηδενική διακριτική ικανότητα στις κλάσεις με τον μικρότερο πληθυσμό.

Στις παραγράφους που ακολουθούν παρουσιάζονται αναλυτικά τα αποτελέσματα για κάθε ένα από τους ταξινομητές δομημένα σε πίνακες ανά μετρική και μέγεθος διανύσματος. Επίσης παρουσιάζονται, ανά ταξινομητή, γραφικές παραστάσεις:

- των τιμών $wP*F_1$,
- των τιμών wF_1 κάθε μιας κλάσης,

για κάθε μετρική και για διαφορετικά μεγέθη διανύσματος.

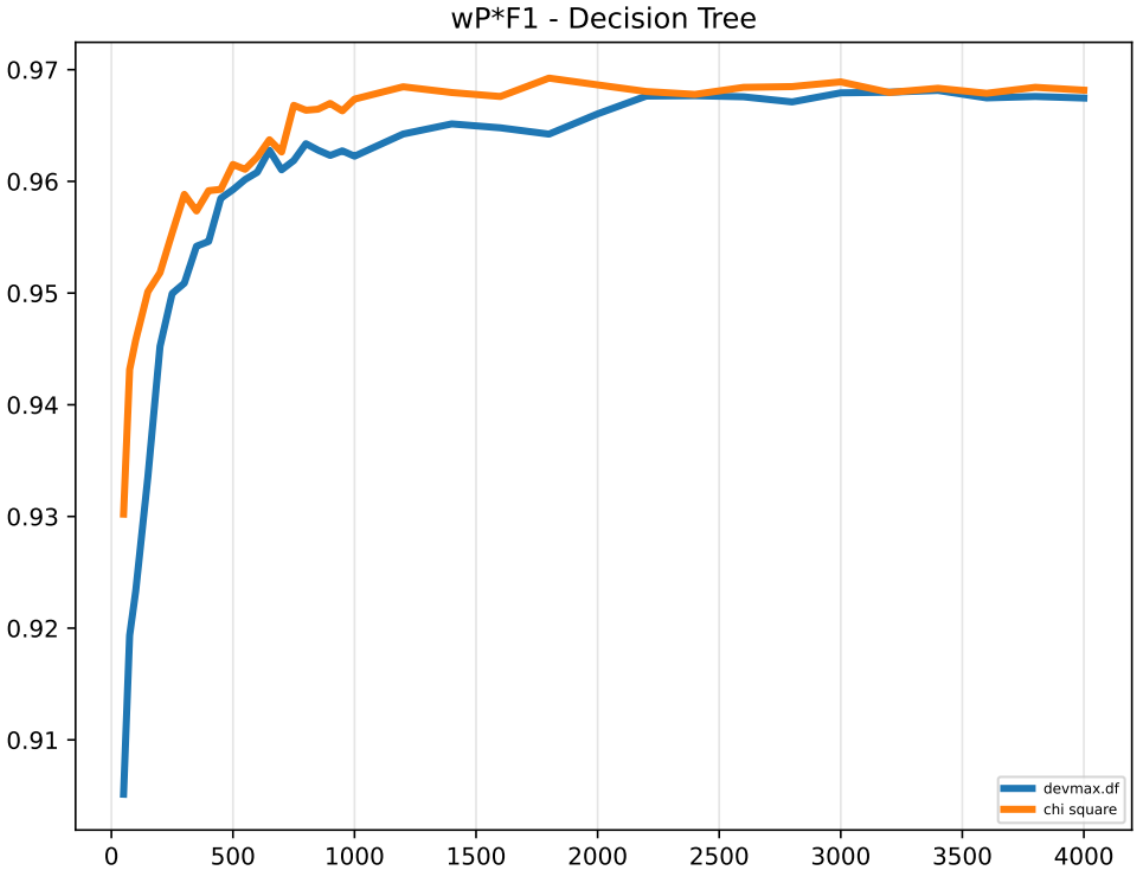
4.2.1.1 Decision Tree [10f-Training1]

Η βέλτιστη απόδοση επιτυγχάνεται για διάνυσμα μεγέθους 1800 και την μετρική Chi square όπως φαίνεται στον ακόλουθο πίνακα.

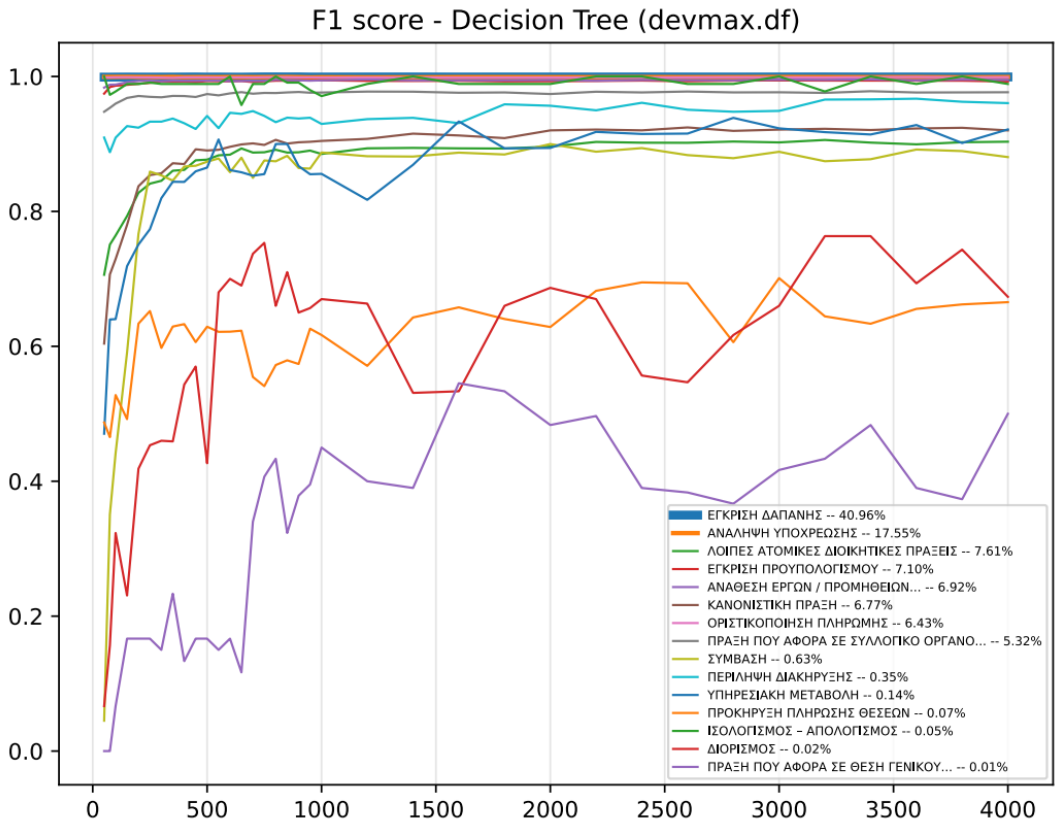
Πίνακας 15: Αποτελέσματα Decision Tree (Είδος πράξης) [10f-Training1]

Λέξεις	DEVMAX.DF				CHI SQUARE			
	wF1	wP	wR	wP*F1	wF1	wP	wR	wP*F1
50	93,64	95,04	92,97	90,51	96,02	96,22	95,85	93,02
75	95,02	95,74	94,56	91,93	96,88	96,94	96,85	94,32
100	95,41	95,87	95,08	92,33	97,03	97,10	96,98	94,58
150	96,16	96,42	95,93	93,35	97,28	97,36	97,21	95,01
200	96,97	97,07	96,90	94,52	97,38	97,44	97,34	95,18
250	97,26	97,34	97,20	95,00	97,61	97,62	97,62	95,54
300	97,31	97,41	97,24	95,09	97,80	97,82	97,79	95,88
350	97,55	97,55	97,56	95,42	97,70	97,75	97,65	95,73
400	97,56	97,58	97,55	95,46	97,81	97,85	97,77	95,92
450	97,81	97,78	97,85	95,85	97,82	97,85	97,79	95,93
500	97,84	97,82	97,88	95,93	97,95	97,97	97,94	96,15
550	97,90	97,88	97,93	96,02	97,95	97,93	97,97	96,11
600	97,93	97,91	97,96	96,08	98,00	97,99	98,03	96,22
650	98,05	98,02	98,09	96,28	98,09	98,08	98,11	96,37
700	97,98	97,90	98,07	96,10	98,03	98,02	98,05	96,26
750	98,00	97,96	98,06	96,19	98,27	98,23	98,32	96,68
800	98,10	98,02	98,20	96,34	98,24	98,23	98,26	96,64
850	98,04	98,02	98,08	96,28	98,24	98,24	98,25	96,65
900	98,04	97,96	98,14	96,23	98,26	98,26	98,28	96,70
950	98,04	98,02	98,08	96,27	98,26	98,19	98,33	96,63
1000	98,04	97,97	98,11	96,23	98,32	98,26	98,38	96,74
1200	98,12	98,10	98,16	96,42	98,37	98,32	98,43	96,85
1400	98,18	98,15	98,23	96,51	98,33	98,31	98,36	96,80
1600	98,16	98,13	98,21	96,48	98,31	98,28	98,36	96,76
1800	98,14	98,09	98,20	96,42	**98,39**	**98,39**	98,39	**96,92**
2000	98,24	98,19	98,29	96,60	98,37	98,35	98,39	96,86
2200	98,31	98,30	98,33	96,76	98,33	98,31	98,36	96,80
2400	98,31	98,30	98,33	96,77	98,32	98,30	98,34	96,78
2600	98,32	98,28	98,37	96,76	98,36	98,33	98,40	96,84
2800	98,29	98,25	98,34	96,71	98,37	98,33	98,41	96,85
3000	98,32	98,31	98,35	96,79	98,37	98,37	98,38	96,89
3200	98,34	98,30	98,40	96,80	98,33	98,31	98,36	96,80
3400	98,32	98,34	98,30	96,81	98,35	98,33	98,39	96,83
3600	98,31	98,28	98,35	96,75	98,32	98,32	98,32	96,79
3800	98,33	98,28	98,39	96,76	98,35	98,34	98,38	96,84
4000	98,31	98,28	98,35	96,75	98,34	98,32	98,36	96,82

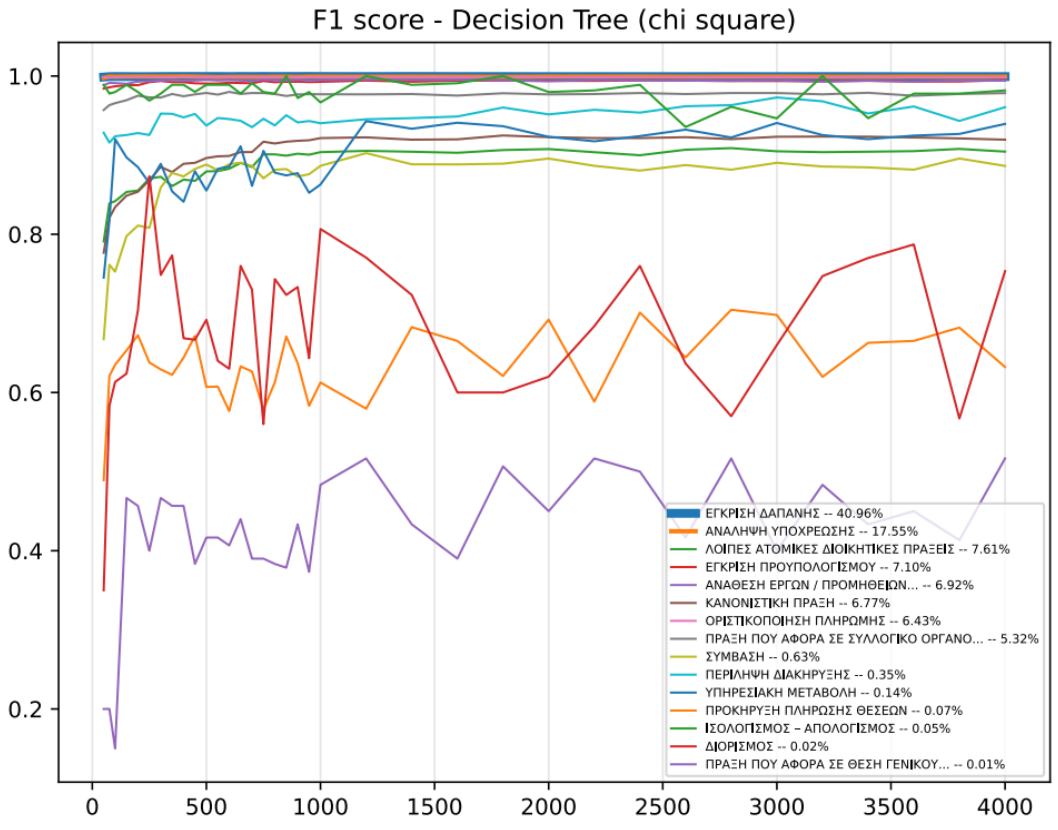
Στις παρακάτω εικόνες εμφανίζονται το συγκριτικό διάγραμμα wP*F_1 για τις δυο μετρικές και τα διαγράμματα του F_1 score για κάθε μια Θεματική κατηγορία (κλάση) και κάθε μια από τις δυο μετρικές.



Εικόνα 43: Διάγραμμα wP*F_1 - Decision Tree (Είδος πράξης) [10f-Training1]



Εικόνα 44: F_1 score ανά Είδος πράξης (DEVMAX.DF - Decision Tree) [10f-Training1]



Εικόνα 45: F_1 score ανά Είδος πράξης (Chi square – Decision Tree) [10f-Training1]

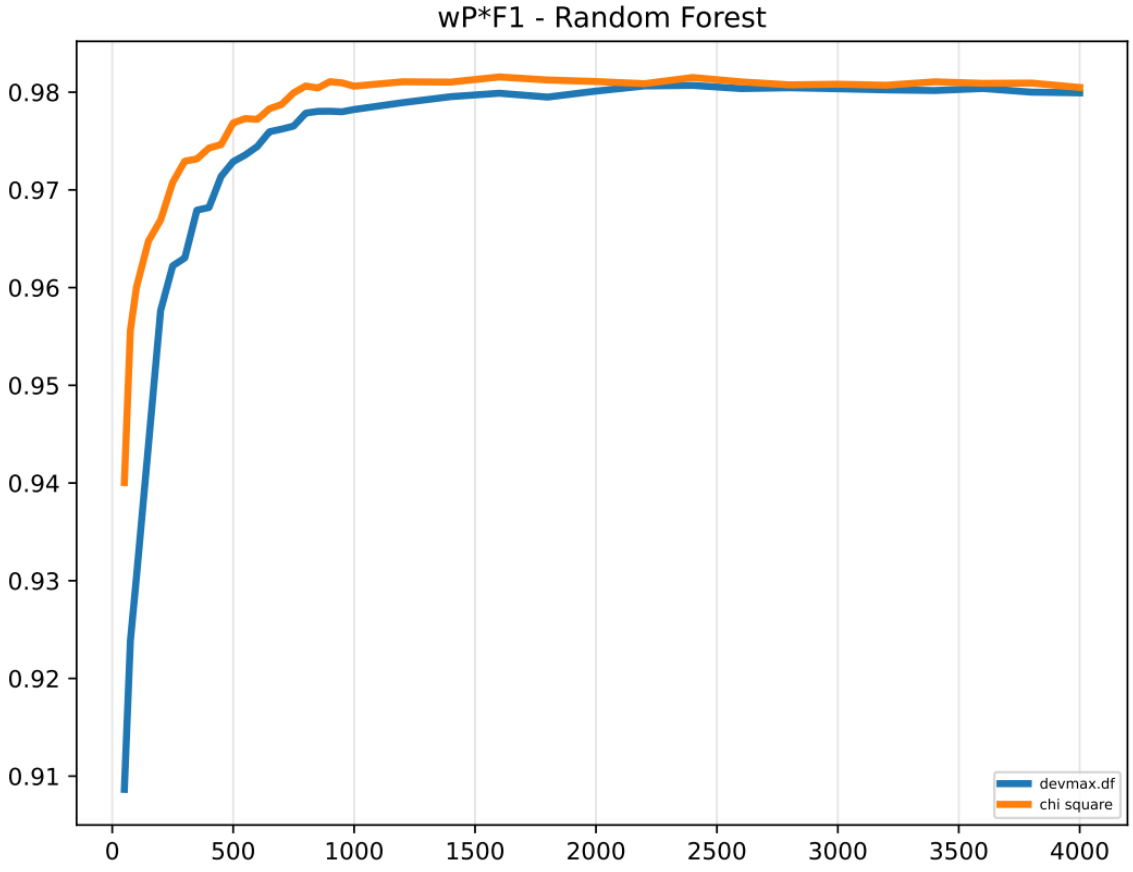
4.2.1.2 Random Forest [10f-Training1]

Η βέλτιστη απόδοση επιτυγχάνεται για διάνυσμα μεγέθους 1600 και την μετρική Chi square όπως φαίνεται στον ακόλουθο πίνακα.

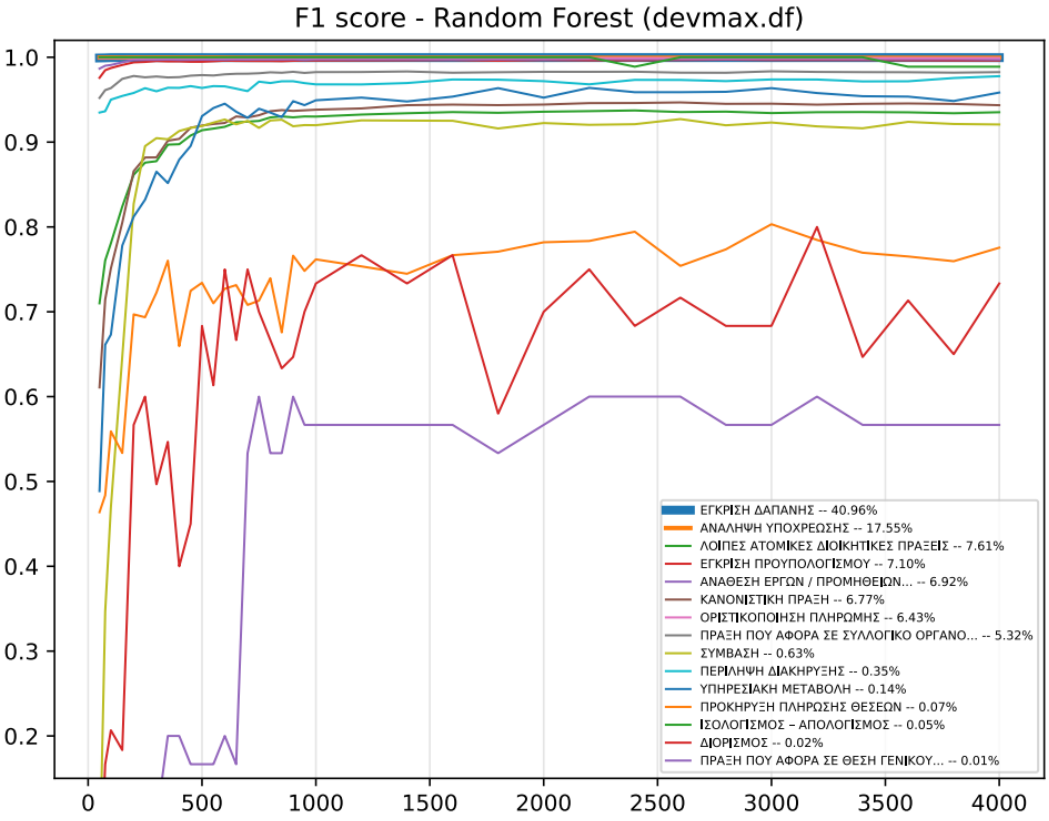
Πίνακας 16: Αποτελέσματα Random Forest (Είδος πράξης) [10f-Training1]

Λέξεις	DEVMAX.DF				CHI SQUARE			
	wF1	wP	wR	wP*F1	wF1	wP	wR	wP*F1
50	93,79	95,30	93,03	90,86	96,52	96,89	96,18	94,00
75	95,25	96,05	94,75	92,38	97,46	97,79	97,16	95,56
100	95,79	96,37	95,43	93,03	97,73	98,01	97,48	96,00
150	96,72	97,12	96,40	94,40	97,98	98,30	97,69	96,48
200	97,62	97,84	97,42	95,76	98,12	98,40	97,86	96,69
250	97,88	98,11	97,68	96,22	98,33	98,61	98,07	97,08
300	97,92	98,15	97,71	96,30	98,47	98,70	98,27	97,29
350	98,20	98,42	98,00	96,79	98,48	98,71	98,26	97,32
400	98,21	98,44	98,01	96,82	98,56	98,76	98,38	97,43
450	98,40	98,60	98,21	97,14	98,59	98,77	98,42	97,46
500	98,48	98,69	98,29	97,29	98,70	98,89	98,53	97,69
550	98,51	98,73	98,30	97,36	98,72	98,92	98,53	97,73
600	98,55	98,78	98,35	97,44	98,72	98,91	98,54	97,72
650	98,64	98,85	98,44	97,60	98,78	98,97	98,60	97,83
700	98,64	98,89	98,41	97,62	98,79	99,01	98,58	97,87
750	98,67	98,89	98,47	97,65	98,86	99,06	98,68	97,99
800	98,74	98,96	98,55	97,79	98,90	99,10	98,71	98,06
850	98,75	98,97	98,55	97,80	98,89	99,09	98,70	98,04
900	98,75	98,97	98,56	97,80	98,92	99,13	98,72	98,11
950	98,76	98,96	98,57	97,80	98,91	99,11	98,72	98,10
1000	98,77	98,97	98,59	97,82	98,89	99,10	98,69	98,06
1200	98,81	99,01	98,61	97,89	98,92	99,12	98,74	98,11
1400	98,85	99,03	98,68	97,95	98,91	99,13	98,72	98,10
1600	98,86	99,06	98,67	97,99	**98,94**	**99,15**	98,75	**98,16**
1800	98,84	99,03	98,65	97,95	98,92	99,13	98,73	98,12
2000	98,87	99,07	98,67	98,01	98,92	99,12	98,73	98,11
2200	98,89	99,11	98,68	98,06	98,90	99,12	98,70	98,09
2400	98,89	99,11	98,69	98,07	98,93	99,15	98,73	98,15
2600	98,88	99,09	98,69	98,04	98,91	99,14	98,69	98,11
2800	98,87	99,10	98,65	98,05	98,90	99,11	98,71	98,07
3000	98,87	99,09	98,66	98,04	98,89	99,13	98,67	98,08
3200	98,87	99,09	98,66	98,02	98,89	99,11	98,69	98,07
3400	98,86	99,08	98,66	98,02	98,91	99,13	98,69	98,11
3600	98,87	99,10	98,65	98,04	98,90	99,13	98,68	98,09
3800	98,85	99,08	98,63	98,00	98,91	99,12	98,71	98,09
4000	98,85	99,07	98,64	97,99	98,87	99,11	98,66	98,05

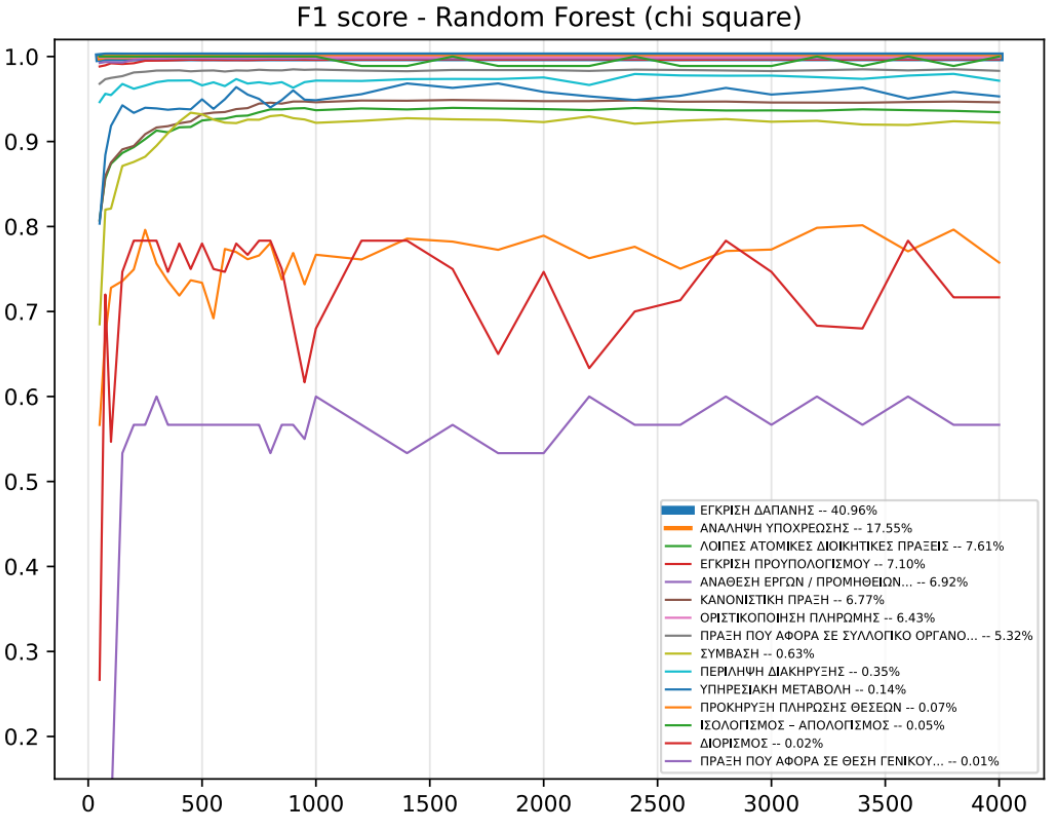
Στις παρακάτω εικόνες εμφανίζονται το συγκριτικό διάγραμμα wP*F_1 για τις δυο μετρικές και τα διαγράμματα του F_1 score για κάθε μια Θεματική κατηγορία (κλάση) και κάθε μια από τις δυο μετρικές.



Εικόνα 46: Διάγραμμα wP*F_1 - Random Forest (Είδος πράξης) [10f-Training1]



Εικόνα 47: F_1 score ανά Είδος πράξης (DEVMAX.DF – Random Forest) [10f-Training1]



Εικόνα 48: F_1 score ανά Είδος πράξης (Chi square – Random Forest) [10f-Training1]

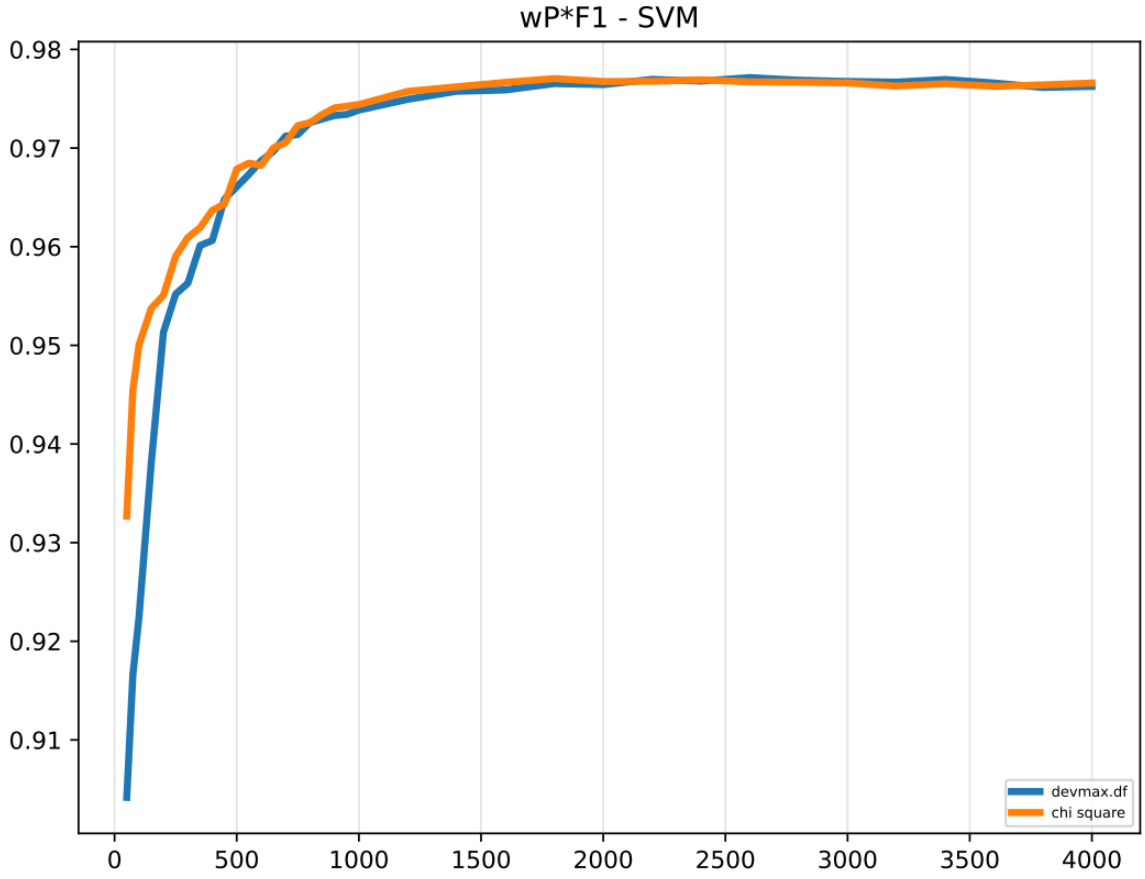
4.2.1.3 SVM [10f-Training1]

Η βέλτιστη απόδοση επιτυγχάνεται για διάνυσμα μεγέθους 2600 και την μετρική DEVMAX.DF όπως φαίνεται στον ακόλουθο πίνακα.

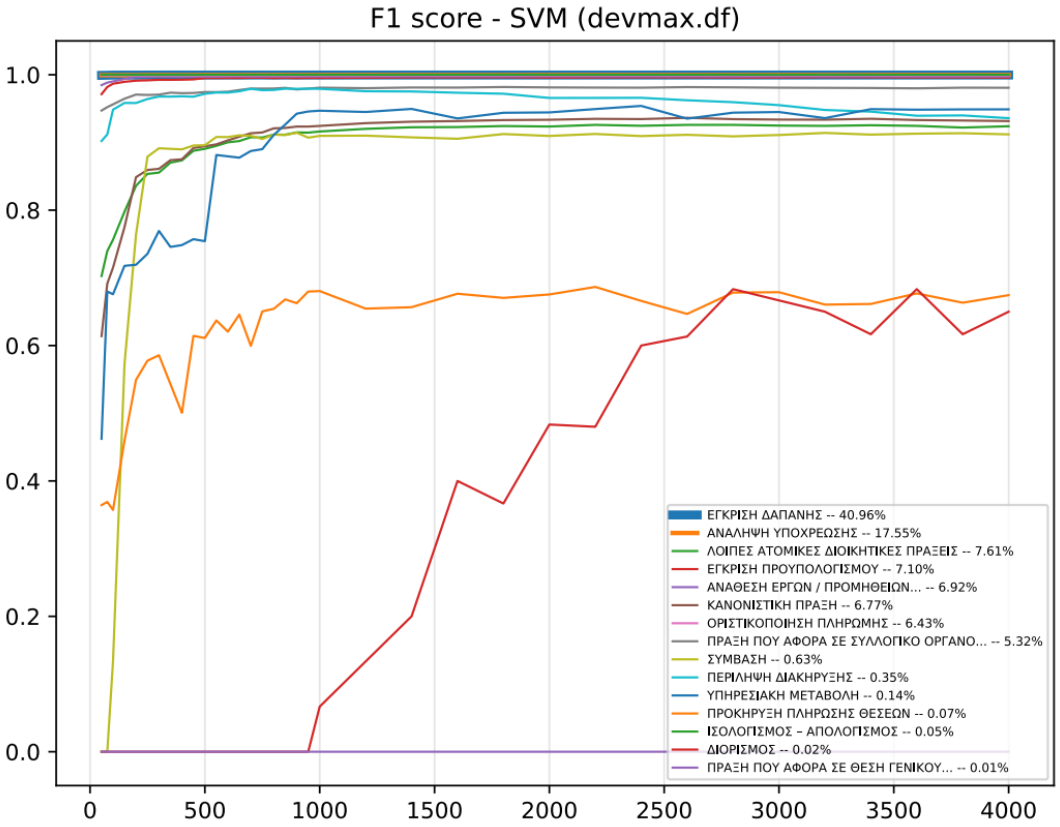
Πίνακας 17: Αποτελέσματα SVM (Είδος πράξης) [10f-Training1]

Λέξεις	DEVMAX.DF				CHI SQUARE			
	wF1	wP	wR	wP*F1	wF1	wP	wR	wP*F1
50	93,63	94,63	93,07	90,42	96,01	96,45	95,62	93,27
75	94,61	95,32	94,12	91,66	96,73	97,32	96,23	94,55
100	95,07	96,11	94,61	92,24	97,07	97,51	96,70	95,01
150	96,18	96,94	95,57	93,80	97,28	97,73	96,89	95,37
200	97,16	97,55	96,83	95,13	97,33	97,82	96,90	95,51
250	97,45	97,72	97,23	95,52	97,60	98,01	97,23	95,90
300	97,51	97,79	97,26	95,63	97,72	98,12	97,36	96,09
350	97,71	98,02	97,46	96,01	97,79	98,16	97,47	96,19
400	97,74	98,05	97,48	96,06	97,97	98,17	97,79	96,37
450	97,98	98,28	97,73	96,48	98,01	98,20	97,84	96,43
500	98,04	98,35	97,79	96,61	98,21	98,39	98,05	96,79
550	98,12	98,41	97,88	96,73	98,23	98,44	98,05	96,85
600	98,21	98,48	97,97	96,87	98,22	98,42	98,06	96,82
650	98,27	98,52	98,06	96,97	98,33	98,51	98,16	97,00
700	98,36	98,59	98,16	97,12	98,36	98,54	98,20	97,05
750	98,37	98,60	98,16	97,14	98,45	98,65	98,28	97,23
800	98,45	98,66	98,25	97,26	98,46	98,66	98,28	97,26
850	98,46	98,68	98,26	97,29	98,51	98,71	98,32	97,34
900	98,49	98,70	98,30	97,33	98,54	98,74	98,36	97,41
950	98,49	98,70	98,30	97,34	98,55	98,76	98,36	97,42
1000	98,52	98,72	98,34	97,38	98,56	98,76	98,37	97,44
1200	98,58	98,79	98,39	97,49	98,63	98,83	98,46	97,57
1400	98,62	98,83	98,43	97,58	98,65	98,86	98,46	97,62
1600	98,63	98,85	98,43	97,59	98,68	98,88	98,51	97,67
1800	98,66	98,88	98,46	97,65	98,70	98,90	98,53	97,70
2000	98,65	98,88	98,44	97,64	98,69	98,89	98,50	97,67
2200	98,68	98,91	98,48	97,70	98,69	98,88	98,53	97,68
2400	98,67	98,90	98,46	97,68	98,70	98,89	98,53	97,69
2600	**98,70**	**98,92**	98,49	**97,71**	98,69	98,88	98,52	97,67
2800	98,68	98,91	98,47	97,69	98,68	98,88	98,50	97,67
3000	98,67	98,91	98,44	97,67	98,68	98,89	98,49	97,66
3200	98,66	98,91	98,43	97,67	98,66	98,87	98,47	97,63
3400	98,67	98,92	98,45	97,69	98,67	98,88	98,49	97,65
3600	98,65	98,91	98,41	97,66	98,66	98,86	98,48	97,63
3800	98,63	98,88	98,40	97,61	98,66	98,88	98,45	97,64
4000	98,63	98,89	98,40	97,63	98,66	98,89	98,45	97,66

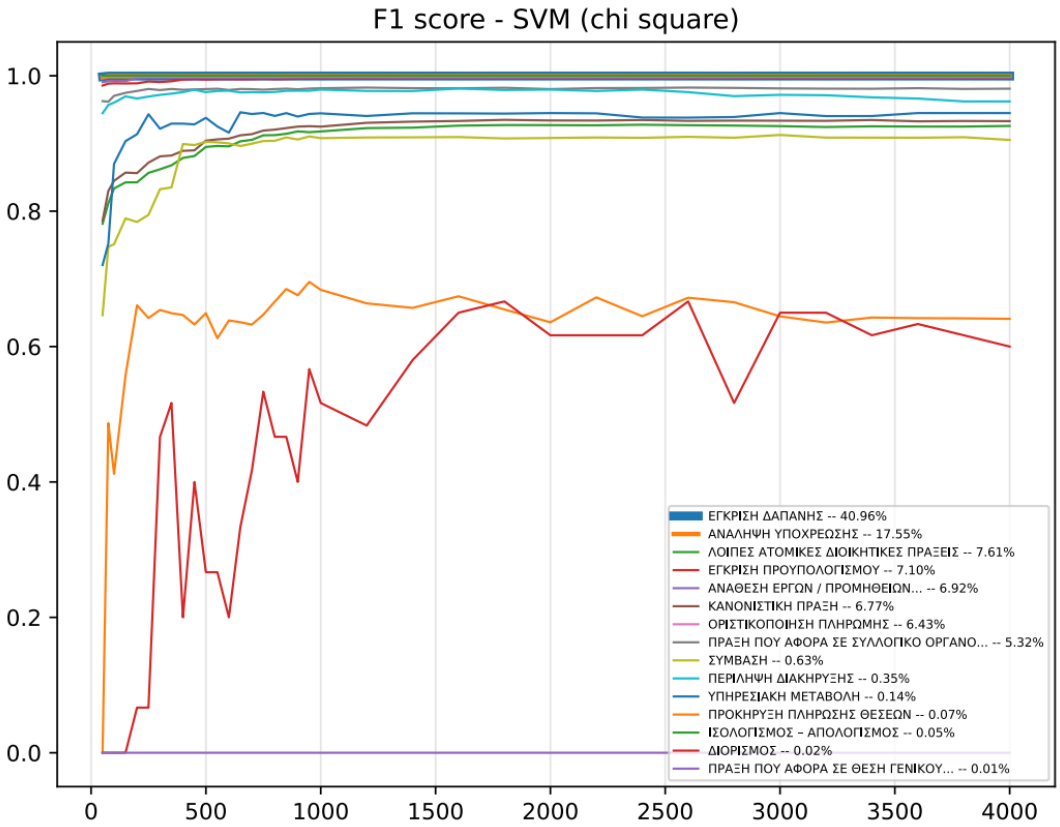
Στις παρακάτω εικόνες εμφανίζονται το συγκριτικό διάγραμμα wP*F_1 για τις δυο μετρικές και τα διαγράμματα του F_1 score για κάθε μια Θεματική κατηγορία (κλάση) και κάθε μια από τις δυο μετρικές.



Εικόνα 49: Διάγραμμα wP*F_1 - SVM (Είδος πράξης) [10f-Training1]



Εικόνα 50: F_1 score ανά Είδος πράξης (DEVMAX.DF – SVM) [10f-Training1]



Εικόνα 51: F_1 score ανά Είδος πράξης (Chi square – SVM) [10f-Training1]

4.2.1.4 Deep Learning [10f-Training1]

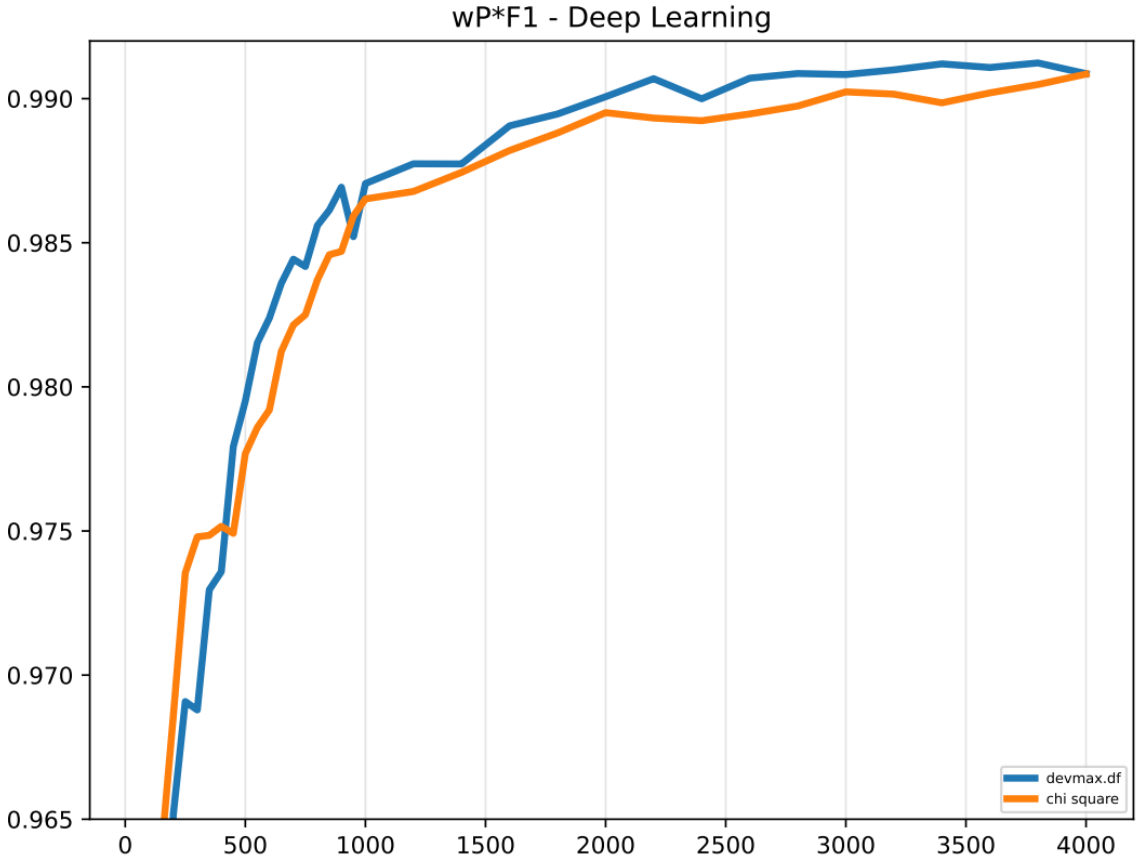
Η επιλογή της απόδοσης που αναφέρεται στη συνέχεια, βασίστηκε σε εκτίμηση βάσει των διαγραμμάτων και σύμφωνα με την διακύμανση των αποδόσεων των κατηγοριών με τον μικρότερο πληθυσμό.

Η βέλτιστη απόδοση επιτυγχάνεται για διάνυσμα μεγέθους 3400 και την μετρική DEVMAX.DF όπως φαίνεται στον ακόλουθο πίνακα.

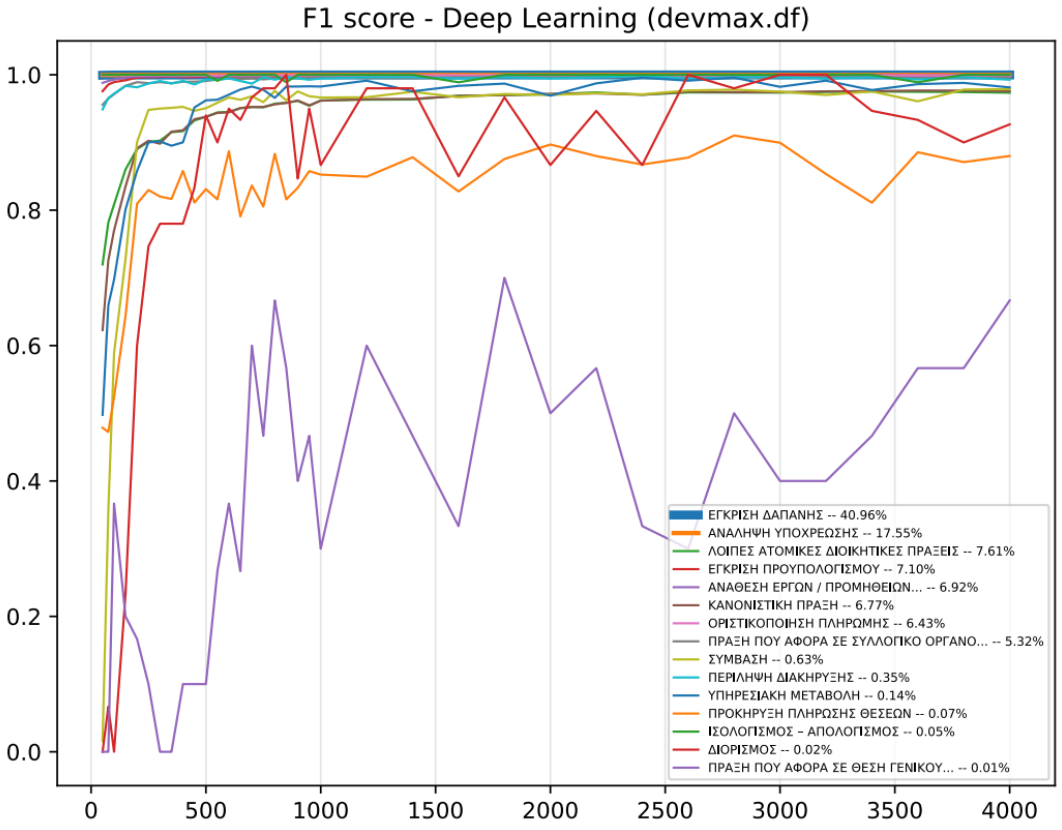
Πίνακας 18: Αποτελέσματα Deep Learning (Είδος πράξης) [10f-Training1]

Λέξεις	DEVMAX.DF				CHI SQUARE			
	wF1	wP	wR	wP*F1	wF1	wP	wR	wP*F1
50	93,97	95,35	93,38	91,02	96,53	96,72	96,41	93,89
75	95,54	96,37	95,19	92,83	97,62	97,63	97,63	95,59
100	96,28	96,66	96,13	93,65	97,84	97,91	97,80	96,03
150	97,34	97,47	97,31	95,23	98,06	98,09	98,05	96,38
200	98,16	98,13	98,22	96,51	98,31	98,36	98,30	96,86
250	98,36	98,38	98,37	96,91	98,60	98,62	98,60	97,35
300	98,36	98,35	98,38	96,88	98,68	98,67	98,71	97,48
350	98,57	98,59	98,57	97,30	98,67	98,69	98,68	97,48
400	98,60	98,62	98,60	97,36	98,68	98,71	98,68	97,52
450	98,84	98,86	98,83	97,79	98,66	98,71	98,64	97,49
500	98,93	98,94	98,93	97,95	98,81	98,85	98,79	97,77
550	99,03	99,04	99,03	98,15	98,87	98,90	98,86	97,86
600	99,07	99,09	99,06	98,24	98,90	98,92	98,89	97,92
650	99,14	99,16	99,13	98,36	99,01	99,04	98,99	98,12
700	99,18	99,21	99,16	98,44	99,06	99,08	99,05	98,21
750	99,16	99,19	99,16	98,42	99,06	99,12	99,02	98,25
800	99,25	99,26	99,24	98,56	99,14	99,16	99,14	98,37
850	99,27	99,29	99,27	98,61	99,19	99,21	99,17	98,46
900	99,32	99,32	99,32	98,69	99,19	99,21	99,18	98,47
950	99,22	99,24	99,22	98,52	99,26	99,27	99,26	98,59
1000	99,32	99,34	99,31	98,71	99,29	99,32	99,27	98,65
1200	99,36	99,37	99,35	98,77	99,31	99,31	99,31	98,68
1400	99,36	99,37	99,37	98,77	99,34	99,35	99,34	98,74
1600	99,42	99,44	99,41	98,91	99,38	99,40	99,38	98,82
1800	99,46	99,45	99,47	98,95	99,41	99,42	99,40	98,88
2000	99,48	99,49	99,47	99,01	99,45	99,46	99,44	98,95
2200	99,51	99,53	99,50	99,07	99,43	99,45	99,41	98,93
2400	99,47	99,49	99,46	99,00	99,44	99,44	99,44	98,92
2600	99,52	99,51	99,53	99,07	99,44	99,46	99,44	98,95
2800	99,52	99,53	99,52	99,09	99,46	99,47	99,46	98,97
3000	99,52	99,53	99,52	99,08	99,48	99,50	99,47	99,02
3200	99,53	99,53	99,54	99,10	99,48	99,49	99,47	99,02
3400	**99,54**	**99,55**	99,54	**99,12**	99,47	99,47	99,48	98,99
3600	99,53	99,54	99,53	99,11	99,49	99,50	99,48	99,02
3800	99,54	99,55	99,54	99,12	99,50	99,51	99,49	99,05
4000	99,52	99,53	99,53	99,09	99,52	99,53	99,51	99,09

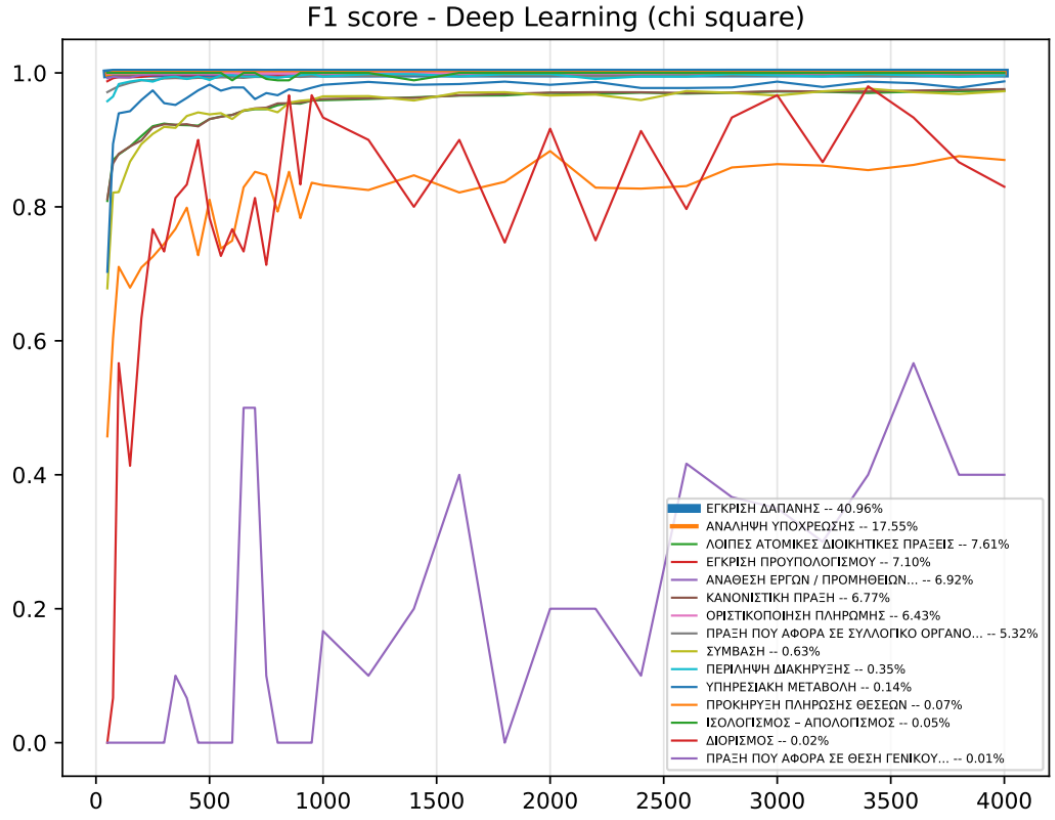
Στις παρακάτω εικόνες εμφανίζονται το συγκριτικό διάγραμμα wP*F_1 για τις δυο μετρικές και τα διαγράμματα του F_1 score για κάθε μια Θεματική κατηγορία (κλάση) και κάθε μια από τις δυο μετρικές.



Εικόνα 52: Διάγραμμα wP*F_1 – Deep Learning (Είδος πράξης) [10f-Training1]



Εικόνα 53: F_1 score ανά Είδος πράξης (DEVMAX.DF - Deep Learning) [10f-Training1]



Εικόνα 54: F_1 score ανά Είδος πράξης (Chi square - Deep Learning) [10f-Training1]

4.2.2 [Val-Training1Test1] Validation: Training Dataset (22/03/2018 – 21/03/2023) => Test Dataset (22/03/2023 – 23/06/2023)

Στον παρακάτω πίνακα εμφανίζονται τα αποτελέσματα που πέτυχε κάθε αλγόριθμος στις μετρικές wF_1, wP και $wP*F_1$ κατά τη φάση της επικύρωσης (validation) στην κατηγοριοποίηση των δεδομένων του Test Dataset. Τα αποτελέσματα είναι υποδεέστερα αυτών που παρήχθησαν κατά την διαδικασία αξιολόγησης (evaluation phase) στο σύνολο των ταξινομητών.

Η μετρική DEVMAX.DF στον Deep Learning ταξινομητή πέτυχε την υψηλότερη απόδοση.

Πίνακας 19: Συγκριτικά αποτελέσματα φάσεων ταξινόμησης (Είδος πράξης) [Val-Training1Test1]

			Evaluation Phase Training Dataset (22/03/2018 – 21/03/2023)			**Validation Phase Training => Test Dataset (22/03/2023 – 23/06/2023)**		
Ταξινομητής	**Μέθοδος**	**Λέξεις (διάνυσμα)**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**
Decision Tree	chi square	1800	98,39	98,39	**96,92**	75,94	73,64	**58,98**
Random Forest	chi square	1600	98,94	99,15	**98,16**	77,83	75,79	**62,26**
SVM	devmax.df	2600	98,70	98,72	**97,71**	77,40	75,63	**61,74**
Deep Learning	devmax.df	3400	99,54	99,55	**99,12**	78,36	75,84	**62,79**

Στις επόμενες ενότητες παρουσιάζονται αναλυτικά τα αποτελέσματα ανά μέθοδο.

4.2.2.1 Decision Tree [Val-Training1Test1]

Λέξεις	**CHI SQUARE**			
	wF1	**wP**	**wR**	**wP*F1**
1200	76,01	73,66	89,83	58,97
1400	75,45	73,10	89,27	58,25
1600	76,00	73,73	89,65	59,08
1800	75,94	73,64	89,62	58,98
2000	76,31	73,90	90,15	59,52
2200	75,81	73,28	89,65	58,73
2400	76,16	73,72	89,88	59,30

4.2.2.2 Random Forest [Val-Training1Test1]

Λέξεις	**CHI SQUARE**			
	wF1	**wP**	**wR**	**wP*F1**
1000	77,75	75,71	91,12	62,13
1200	77,76	75,69	91,14	62,11
1400	77,83	75,75	91,23	62,23
1600	77,83	75,79	91,20	62,26
1800	77,64	75,58	91,02	61,91
2000	77,86	75,70	91,33	62,21
2200	77,85	75,80	91,24	62,29

4.2.2.3 SVM [Val-Training1Test1]

Λέξεις	DEVMAX.DF			
	wF1	wP	wR	wP*F1
2000	77,74	75,75	91,06	62,15
2200	77,52	75,73	90,70	61,93
2400	77,48	75,68	90,68	61,85
2600	77,40	75,63	90,57	61,74
2800	77,21	75,64	90,27	61,58
3000	77,27	75,64	90,35	61,63
3200	77,34	75,67	90,43	61,71

4.2.2.4 Deep Learning [Val-Training1Test1]

Λέξεις	DEVMAX.DF			
	wF1	wP	wR	wP*F1
2800	78,35	75,85	92,15	62,78
3000	77,92	75,67	91,48	62,24
3200	77,67	75,44	91,24	61,82
3400	78,36	75,84	92,20	62,79
3600	78,10	75,35	92,26	62,12
3800	77,90	75,68	91,43	62,26
4000	77,81	75,50	91,43	61,99

4.2.3 Επιπλέον ενέργειες – Βήματα

Στη συνέχεια παρουσιάζονται τα αποτελέσματα που προέκυψαν μετά τις ενέργειες που περιγράφονται στην παράγραφο 0.

4.2.3.1 Μείωση του εύρους της χρονικής περιόδου κατά ένα μήνα [Val-Training1Test2]

Πίνακας 20: Συγκριτικά αποτελέσματα - μείωση του εύρους της χρονικής περιόδου του Test Dataset κατά ένα μήνα (Είδος πράξης) [Val-Training1Test2]

			Evaluation Phase			Validation Phase					
			Training Dataset (22/03/2018 – 21/03/2023)			Training => Test Dataset22/03/2023 – 23/06/2023			Training => Test Dataset 22/03/2023 – 23/05/2023		
Ταξινομητής	Μέθοδος	Λέξεις (διάνυσμα)	wF_1	wP	$wP*F_1$	wF_1	wP	$wP*F_1$	wF_1	wP	$wP*F_1$
Decision Tree	chi square	1800	98,39	98,39	**96,92**	75,94	73,64	**58,98**	78,06	77,94	**62,33**
Random Forest	chi square	1600	98,94	99,15	**98,16**	77,83	75,79	**62,26**	83,35	82,35	**70,17**
SVM	devmax.df	2600	98,70	98,72	**97,71**	77,40	75,63	**61,74**	83,13	82,08	**69,65**
Deep Learning	devmax.df	3400	99,54	99,55	**99,12**	78,36	75,84	**62,79**	83,31	81,95	**69,70**

4.2.3.2 Μείωση του εύρους της χρονικής περιόδου κατά δυο μήνες [Val-Training1Test3]

Πίνακας 21: Συγκριτικά αποτελέσματα - μείωση του εύρους της χρονικής περιόδου του Test Dataset κατά δυο μήνες (Είδος πράξης) [Val-Training1Test3]

			Evaluation Phase			Validation Phase					
			Training Dataset (22/03/2018 – 21/03/2023)			Training => Test Dataset 22/03/2023 – 23/06/2023			Training => Test Dataset 22/03/2023 – 30/04/2023		
Ταξινομητής	**Μέθοδος**	**Λέξεις (διάνυσμα)**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**
Decision Tree	chi square	1800	98,39	98,39	**96,92**	75,94	73,64	**58,98**	88,76	89,59	**81,71**
Random Forest	chi square	1600	98,94	99,15	**98,16**	77,83	75,79	**62,26**	93,42	94,19	**89,02**
SVM	devmax.df	2600	98,70	98,72	**97,71**	77,40	75,63	**61,74**	93,19	93,88	**88,46**
Deep Learning	devmax.df	3400	99,54	99,55	**99,12**	78,36	75,84	**62,79**	93,53	93,62	**88,59**

4.2.3.3 Νέα Training και Test Datasets [10f-Training2] [Val-Training2Test4]

Πίνακας 22: Συγκριτικά αποτελέσματα - νέα Training και Test Datasets (Είδος πράξης) [10f-Training2] [Val-Training2Test4]

			Evaluation Phase Training Dataset (22/03/2018 – 23/06/2023)			Validation Phase Training => Test Dataset (24/06/2023 – 23/08/2023)		
Ταξινομητής	**Μέθοδος**	**Λέξεις (διάνυσμα)**	**wF_1**	**wP**	**$wP*F_1$**	**wF_1**	**wP**	**$wP*F_1$**
Decision Tree	chi square	1800	98,19	98,21	**96,58**	92,05	90,42	**84,38**
Random Forest	chi square	1600	98,87	99,08	**98,02**	96,18	95,13	**92,40**
SVM	devmax.df	2600	98,67	98,92	**97,69**	96,57	95,62	**93,21**
Deep Learning	devmax.df	3400	99,50	99,52	**99,06**	98,73	98,83	**97,68**

Κεφάλαιο 5. Συμπεράσματα

Με τα αρχικά Datasets, οι αποδόσεις των ταξινομητών στη φάση αξιολόγησης (evaluation phase) στην κατηγοριοποίηση των πράξεων βάσει της «Θεματικής κατηγορίας» ήταν εξαιρετικά υψηλές. Στα ίδια υψηλά επίπεδα κυμάνθηκαν οι αποδόσεις των ταξινομητών και στη φάση επικύρωσης (validation).

Αντίθετα η προσπάθεια κατηγοριοποίησης βάσει του «Είδους πράξης» πέτυχε πολύ χαμηλές αποδόσεις στη φάση της επικύρωσης παρόλες τις υψηλές αποδόσεις που είχαν πετύχει οι ταξινομητές στη φάση της αξιολόγησης.

Ως αιτία των χαμηλών αποδόσεων θεωρήθηκε η στατιστική ασυνέπεια μεταξύ των δομών των Training και Test Dataset. Αρχικά έγιναν προσπάθειες βελτίωσης των αποτελεσμάτων με Test Dataset που κάλυπταν διαφορετικές χρονικές περιόδους με τα αποτελέσματα να βελτιώνονται αισθητά. Στη συνέχεια δημιουργήθηκε νέο Training Dataset, το οποίο προέκυψε από τη συγχώνευση των παλιών Training και Test Datasets, καθώς και νέο Test Dataset που κάλυπτε επόμενη χρονική περίοδο.

Με την δημιουργία των νέων Datasets, έγινε εκ νέου αξιολόγηση και επικύρωση των αλγορίθμων τόσο για το Είδος πράξης, που παρουσίασε με τα αρχικά Datasets χαμηλές αποδόσεις, όσο και για την Θεματική κατηγορία. Η επανάληψη της διαδικασίας για την Θεματική κατηγορία είχε ως σκοπό την επιβεβαίωση ότι τα υψηλά αποτελέσματα που είχαν επιτευχθεί με τα αρχικά Datasets δεν οφείλονταν σε κάποιο τυχαίο γεγονός.

Για την ταξινόμηση της Θεματικής κατηγορίας τα υψηλότερα αποτελέσματα στην φάση της επικύρωσης (validation) επιτεύχθηκαν με την μετρική Devmax.df και τον Deep Learning ταξινομητή με την μετρική wP*F1 να επιτυγχάνει ποσοστό 97,32%.

Για την ταξινόμηση του Είδους πράξης τα υψηλότερα αποτελέσματα στην φάση της επικύρωσης (validation) επιτεύχθηκαν με την μετρική Devmax.df και τον Deep Learning ταξινομητή με την μετρική wP*F1 να επιτυγχάνει ποσοστό 97,68%.

Πίνακας 23: Συγκεντρωτικά αποτελέσματα ταξινομητών – Θεματική κατηγορία/Είδος πράξης [Val-Training2Test4]

Ταξινόμηση	Ταξινομητής	Μέθοδος	Λέξεις (διάνυσμα)	wF_1	wP	wR	$wP*F_1$
Θεματική κατηγορία	Deep Learning	devmax.df	4200	98,74	98,50	99,08	**97,32**
Είδος πράξης	Deep Learning	devmax.df	3400	98,73	98,83	98,71	**97,68**

Τα παραπάνω αποτελέσματα, με αναλυτικές μετρήσεις για κάθε κλάση καθώς και τα confusion matrices, παρουσιάζονται στα Παραρτήματα Η και Θ αντίστοιχα.

Λόγω των υψηλών τιμών των αποτελεσμάτων αλλά και χρονικών περιορισμών δεν έγινε προσπάθεια να βρεθούν οι βέλτιστες παράμετροι των ταξινομητών. Επίσης, δεν δοκιμάστηκαν διαφορετικές πιο σύγχρονες τοπολογίες νευρωνικών δικτύων (CNN, LSTM) η χρήση των οποίων θα οδηγούσε πιθανώς σε καλύτερα αποτελέσματα.

Συμπερασματικά, η υπολογιστική αρχειακή επιστήμη αντιπροσωπεύει ένα αναπτυσσόμενο επιστημονικό πεδίο που ενσωματώνει τις παραδοσιακές πρακτικές της αρχειακής επιστήμης με τις τεχνολογικές εξελίξεις στον τομέα των υπολογιστών και της ανάλυσης μεγάλων δεδομένων. Στο πλαίσιο της διαχείρισης πανεπιστημιακών αρχείων, αυτός ο αναδυόμενος κλάδος έχει τη δυνατότητα να φέρει επανάσταση στον τρόπο με τον οποίο τα εκπαιδευτικά ιδρύματα χειρίζονται, επεξεργάζονται και διατηρούν τις τεράστιες αποθήκες διοικητικών, ακαδημαϊκών και ιστορικών δεδομένων τους. Αξιοποιώντας τη δύναμη των υπολογιστικών μεθόδων επεξεργασίας φυσικής γλώσσας και μηχανικής μάθησης τα πανεπιστήμια μπορούν να εξορθολογήσουν τις διαδικασίες διαχείρισης αρχείων τους, να βελτιώσουν την ανάκτηση πληροφοριών και να διευκολύνουν την αποτελεσματική ανάλυση πολύπλοκων αρχειακών συλλογών. Η παρούσα διπλωματική ανέδειξε τις αξιοσημείωτες δυνατότητες αυτοματοποιημένης ταξινόμησης και κατηγοριοποίησης διαφορετικών τύπων αρχειακών εγγράφων του ΠΑΔΑ, λειτουργώντας πιλοτικά, με τελικό στόχο και όφελος την απλοποίηση του περίπλοκου έργου της οργάνωσης και ευρετηρίασης μεγάλων δεδομένων ψηφιακών εγγράφων.

Βιβλιογραφικές Αναφορές

- Amin-Nejad, A., Ive, J., & Velupillai, S. (2020). Exploring Transformer Text Generation for Medical Dataset Augmentation. *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 4699–4708. https://aclanthology.org/2020.lrec-1.578
- Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python: Analyzing text with the natural language toolkit*. OReilly Media, Inc.
- Chen, H., Wu, L., Chen, J., Lu, W., & Ding, J. (2022). A comparative study of automated legal text classification using random forests and deep learning. *Information Processing & Management*, *59*(2), 102798. https://doi.org/10.1016/j.ipm.2021.102798
- Chen, P.-F., Wang, S.-M., Liao, W.-C., Kuo, L.-C., Chen, K.-C., Lin, Y.-C., Yang, C.-Y., Chiu, C.-H., Chang, S.-C., & Lai, F. (2021). Automatic ICD-10 Coding and Training System: Deep Neural Network Based on Supervised Learning. *JMIR Medical Informatics*, *9*(8), e23230. https://doi.org/10.2196/23230
- Chollet, F. (2020). *Deep Learning with Python MEAP* (2nd ed.). Manning Publications.
- Chollet, F., & others. (2015). *Keras*. GitHub. https://github.com/fchollet/keras
- Gasparetto, A., Marcuzzo, M., Zangari, A., & Albarelli, A. (2022). A Survey on Text Classification Algorithms: From Text to Predictions. *Information*, *13*(2), Article 2. https://doi.org/10.3390/info13020083
- Geron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow* (2nd ed.). OReilly Media, Inc.
- Goldberg, Y. (2017). *Neural Network Methods in Natural Language Processing* (G. Hirst, Ed.). Morgan & Claypool Publishers.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
- Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2
- Honnibal, M., & Montani, I. (2017). *spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing*.

- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55
- Kesiku, C. Y., Chaves-Villota, A., & Garcia-Zapirain, B. (2022). Natural Language Processing Techniques for Text Classification of Biomedical Documents: A Systematic Review. *Information*, *13*(10), Article 10. https://doi.org/10.3390/info13100499
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539
- Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P. S., & He, L. (2022). A Survey on Text Classification: From Traditional to Deep Learning. *ACM Transactions on Intelligent Systems and Technology*, *13*(2), 31:1-31:41. https://doi.org/10.1145/3495162
- Lipton, Z. C., Elkan, C., & Narayanaswamy, B. (2014). *Thresholding Classifiers to Maximize F1 Score* (arXiv:1402.1892). arXiv. https://doi.org/10.48550/arXiv.1402.1892
- Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Jia, Y., Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, … Xiaoqiang Zheng. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. https://www.tensorflow.org/
- McKie, J. X., & Liu, R. (2016). *PyMuPDF* [Computer software]. Artifex. https://github.com/pymupdf/PyMuPDF
- McKinney, W. (2010). *Data Structures for Statistical Computing in Python*. 56–61. https://doi.org/10.25080/Majora-92bf1922-00a
- Mirończuk, M. M., & Protasiewicz, J. (2018). A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications*, *106*, 36–54. https://doi.org/10.1016/j.eswa.2018.03.058
- Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.
- Nobles, A. L., Glenn, J. J., Kowsari, K., Teachman, B. A., & Barnes, L. E. (2018). Identification of Imminent Suicide Risk Among Young Adults using Text Messages. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI Conference*, *2018*, 413. https://doi.org/10.1145/3173574.3173987
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825–2830.

- Raschka, S., & Mirjalili, V. (2019). *Python Machine Learning* (3rd ed.). Packt Publishing Ltd.
- Sarkar, D. (2016). *Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from your Data*. Apress.
- Sechidis, K., Tsoumakas, G., & Vlahavas, I. (2011). On the Stratification of Multi-label Data. In D. Gunopulos, T. Hofmann, D. Malerba, & M. Vazirgiannis (Eds.), *Machine Learning and Knowledge Discovery in Databases* (pp. 145–158). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-23808-6_10
- Sharma, S., Sharma, S., & Athaiya, A. (2020). ACTIVATION FUNCTIONS IN NEURAL NETWORKS. *International Journal of Engineering Applied Sciences and Technology*, *04*(12), 310–316. https://doi.org/10.33564/IJEAST.2020.v04i12.054
- Singh, D. Y., & Chauhan, A. S. (2009). Neural Networks in Data Mining. *Journal of Theoretical and Applied Information Technology*, *5*(1), 37–42.
- Szymański, P., & Kajdanowicz, T. (2017). A Network Perspective on Stratification of Multi-Label Data. *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*, 22–35. https://proceedings.mlr.press/v74/szyma%C5%84ski17a.html
- Szymański, P., & Kajdanowicz, T. (2018). *A scikit-based Python environment for performing multi-label classification* (arXiv:1702.01460). arXiv. https://doi.org/10.48550/arXiv.1702.01460
- Triantafyllou, I., Drivas, I. C., & Giannakopoulos, G. (2020). How to Utilize My App Reviews? A Novel Topics Extraction Machine Learning Schema for Strategic Business Purposes. *Entropy (Basel, Switzerland)*, *22*(11), 1310. https://doi.org/10.3390/e22111310
- Vajjala, S., Majumder, B., Gupta, A., & Surana, H. (2020). *Practical Natural Language Processing*. OReilly Media, Inc.
- Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2
- Vorgia, F., Triantafyllou, I., & Koulouris, A. (2017). Hypatia Digital Library: A Text Classification Approach Based on Abstracts. In A. Kavoura, D. P. Sakas, & P. Tomaras

(Eds.), *Strategic Innovative Marketing* (pp. 727–733). Springer International Publishing. https://doi.org/10.1007/978-3-319-33865-1_89

- Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2017). *Data Mining Practical Machine Learning Tools and Techniques* (4th ed.). Morgan Kaufmann.
- Γεωργούλη, Κ. (2015). *Τεχνητή Νοημοσύνη*. ΣΕΑΒ.

- **Παράρτημα – Α**

Ο παρακάτω κώδικας, χρησιμοποιώντας συναρτήσεις του API της «Διαύγειας», κάνει λήψη των μεταδεδομένων των προς επεξεργασία πράξεων και τα αποθηκεύει σε μορφή JSON.

```
while operation != 0:
    if operation == 1:
        # Client για το API της Διαύγειας
        client_metadata = od.OpendataClient()
        # Οργανικές μονάδες του ΠΑΔΑ
        pada_units = client_metadata.get_organization_units(padaUid)['units']
        # Κατηγορίες αποφάσεων εγγράφων
        dec_types = client_metadata.get_decision_types()['decisionTypes']
        # Θεματικές κατηγορίες αποφάσεων εγγράφων
        them_cat = client_metadata.get_dictionary('THK')['items']

        # Χρονικά διαστήματα με διαφορά 180 ημερών μεταξύ τους για την αναζήτηση των εγγράφων από τη Διαύγεια
        # Η πρώτη ημερομηνία είναι αυτή του παλαιότερου καταχωρισμένου εγγράφου
        intervals = ['2018-03-22', '2018-09-18', '2019-03-17', '2019-09-13', '2020-03-11', '2020-09-07',
                     '2021-03-06', '2021-09-02', '2022-03-01', '2022-08-28', '2023-02-24']

        dataset = int(input('Δημιουργία (1) Training dataset - (2) Test dataset:\n'))
        while dataset != 1 or dataset != 2:
            dataset = int(input('Δημιουργία (1) Training dataset - (2) Test dataset:\n'))
        if dataset == 2:
            intervals = ['2023-03-22']

        # Όνομα πρώτου συνθετικού του ονόματος για την αποθήκευση των αποτελεσμάτων.
        # Η αποθήκευση γίνεται ανά χρονικό διάστημα και το όνομα του αρχείου έχει τη μορφή: filename+interval
        filename: str = str(input('Όνομα αρχείου για αποθήκευση:\n'))

        # Για κάθε ένα χρονικό διάστημα...
        for period in range(len(intervals)):
            try:
                # ...ξεκινώντας από την πρώτη σελίδα αποτελεσμάτων...
                page = 0
                # Για την προσωρινή αποθήκευση μεταδεδομένων
                final = []
                while True:
                    # ...ζητάμε ανά 500 εγγραφές...
                    res = client_metadata.get_simple_search_results(
                        org=padaUid, from_issue_date=intervals[period], page=page, size=500)
                    decisions = res['decisions']
                    # ...μέχρι αυτές να εξαντληθούν, οπότε συνεχίζουμε με το επόμενο χρονικό διάστημα.
                    if len(decisions) == 0:
                        break
                    # ...ΑΛΛΙΩΣ...
                    # ...αντλούμε και κρατάμε τα ζητούμενα μεταδεδομένα...
                    res = rm.get_metadata(decisions, pada_units, dec_types, them_cat)
                    final += res
                    # ...και συνεχίζουμε με την επόμενη σελίδα αυτού του χρονικού διαστήματος.
                    page += 1
            except requests.exceptions.Timeout:
                print('Πρόβλημα στη σελίδα {}'.format(period))
            # Για κάθε χρονικό διάστημα αποθηκεύουμε σε αρχεία json και csv
            rm.save_to_json_file(filename + str(period + 1), final)
            rm.save_to_csv_file(filename + str(period + 1), final)
```

```
def get_metadata(decisions: list, org_units: list, decision_types: list, them_cat: list):
    # Για την αποθήκευση των αποτελεσμάτων
    res = []
    # Προσπέλαση των σελίδων αποτελεσμάτων...
    for decision in decisions:
        try:
            # Κλειδιά του λεξικού
            keys = ['ada', 'subject', 'datetime', 'decType', 'unitIds']
            # Ημερομηνία ανάρτησης - μετατροπή σε ημερομηνία από UNIX timestamp
            issue_date = None
            if decision.get('issueDate') is not None:
                issue_date = str(datetime.fromtimestamp(decision['issueDate'] / 1000).date())
            # Έλεγχος ύπαρξης κωδικού οργανικής μονάδας και αντιστοίχιση με όνομα από το org_units
            unit_name = None
            if decision.get('unitIds') is not None:
                uid = str(decision['unitIds'][0])
                for org in org_units:
                    if org['uid'] == uid:
                        unit_name = org['label']
            # Ανάκτηση κωδικού τύπου απόφασης και αντιστοίχιση με όνομα από το decision_types
            decision_type = None
            if decision.get('decisionTypeId') is not None:
                dt_id = str(decision['decisionTypeId'])
                for dec_type in decision_types:
                    if dec_type['uid'] == dt_id:
                        decision_type = dec_type['label']
            # Δημιουργία λίστας τιμών μεταδεδομένων
            values = [decision['ada'], decision['subject'], issue_date, decision_type, unit_name]
            # Ανάκτηση κωδικού θεματικής κατηγορίας και αντιστοίχιση με όνομα από το them_cat
            # Εύρεση θεματικών κατηγοριών...
            length = len(decision['thematicCategoryIds'])
            for i in range(length):
                uid = str(decision['thematicCategoryIds'][i])
                label = list(cat['label'] for cat in them_cat if cat['uid'] == uid)[0]
                # ...και προσθήκη τους στα αποτελέσματα
                values.append(label)
                keys.append('themCat' + str(i + 1))
            # Δημιουργία τελικής εγγραφής
            res.append(dict(zip(keys, values)))
        except Exception:
            print('Σφάλμα - get_metadata')
            pass
    return res
```

Παράρτημα – Β

Ο παρακάτω κώδικας κάνει λήψη και αποθηκεύει τοπικά τα PDF αρχεία των ζητούμενων πράξεων, βάσει του URL τους (αποθηκευμένο στο JSON αρχείο των μεταδεδομένων των πράξεων).

```
# Αποθήκευση των PDF αρχείων από τη σελίδα της ΔΙΑΥΓΕΙΑΣ
# Αποθήκευση σε φάκελο των αρχείων pdf των πράξεων βάσει του, αποθηκευμένου σε json αρχείο, url
wallia
def pdf2folder_fromUrl(jsonfile: str, start=0):
    # Άνοιγμα του json αρχείου και φόρτωση των δεδομένων του
    with open(localFolder + '/' + jsonfile + '.json', encoding='utf-8') as f:
        data = json.load(f)
    # Προσπέλαση των δεδομένων
    for i in range(start, len(data)):
        # Εύρεση των url και ΑΔΑ και δημιουργία της διαδρομής για το προς αποθήκευση αρχείο
        # url = data[i]['pdfUrl']
        ada = data[i]['ada']
        url = diavgeiaPdfUrl + ada
        path = os.path.join(localFolderPDFs, ada + '.pdf')
        # Κατέβασμα και αποθήκευση στον φάκελο
        if not os.path.exists(path):
            try:
                p = Path(path)
                url_retrieve(url, p)
            except Exception:
                print("Δεν μπορεί να γίνει λήψη του pdf με url: {}".format(url))
                pass


# Βασική συνάρτηση για την pdf2folder_fromUrl / Λήψη από url και αποθήκευση σε φάκελο
wallia
def url_retrieve(url: str, outfile: Path):
    r = requests.get(url, allow_redirects=True)
    if r.status_code != 200:
        raise ConnectionError("could not download {}\nerror code: {}".format(url, r.status_code))
    outfile.write_bytes(r.content)
```

Παράρτημα – Γ

Τα παρακάτω τμήματα κώδικα διαβάζουν τα τοπικά αποθηκευμένα PDF αρχεία των πράξεων, εξάγουν το κείμενο χρησιμοποιώντας την βιβλιοθήκη PyMuPDF της γλώσσας προγραμματισμού Python και το αποθηκεύουν ενημερώνοντας το JSON αρχείο μεταδεδομένων.

```python
# Εξαγωγή κειμένου από τοπικό ή διαδικτυακό pdf για κάθε εγγραφή που υπάρχει σε json μεταδεδομένων
# Δημιουργία ενημερωμένου αρχείου json
# Extracting method: 'F' για PyMuPdf και 'M' για PyPDF2
wallia
def addExtractedText(json_file, extracting_method):
    # Τοποθεσία αρχείου μεταδεδομένων
    filename = json_file + '.json'
    path = os.path.join(localFolder, filename)
    # Άνοιγμα του json αρχείου και φόρτωση των δεδομένων του
    with open(path, encoding='utf-8') as f:
        data = json.load(f)
    # Προσπέλαση των δεδομένων και αναζήτηση σχετικού pdf αρχείου
    for i in range(len(data)):
        ada = data[i]['ada']
        # url = data[i]['pdfUrl']
        url = diavgeiaPdfUrl + data[i]['ada']
        path = os.path.join(localFolderPDFs, ada+'.pdf')
        text = ""
        has_text = data[i].get('exText')
        if has_text == "" or has_text is None:
            # Αν το pdf αρχείο υπάρχει τοπικά, γίνεται εξαγωγή του κειμένου...
            if os.path.exists(path):
                text = extract_localPdfFile(path, extracting_method)
            else:
                try:
                    # ...αλλιώς κατεβαίνει βάσει του url του, εξάγεται το κείμενο...
                    text = extract_urlPdfFile(url, extracting_method)
                except Exception:
                    print(f'Δεν έγινε εξαγωγή από το {url}')
                    pass
            # ...και ενημερώνονται τα δεδομένα.
            data[i].update(exText=text)
        if i % 500 == 0:
            print(f'Έχουν επεξεργαστεί {i+1} αρχεία και υπολείπονται {len(data)-i-1}')
    # Φάκελος και όνομα νέου αρχείου
    new_path = os.path.join(localFolder, json_file)
    # Αποθήκευση σε json αρχείο, κωδικοποίηση utf-8
    with open(new_path + 'TEXT.json', 'w', encoding='utf8') as ff:
        json.dump(data, ff, ensure_ascii=False, indent=4)
    # Αποθήκευση σε csv αρχείο, χρήση της βιβλιοθήκης pandas
    df = pd.DataFrame.from_dict(data)
    df.to_csv(new_path + 'TEXT.csv', index=False, header=True)
```

```python
# Κύρια συνάρτηση εξαγωγής κειμένου από pdf αρχείο
# Extracting method: 'F' για PyMuPdf και 'M' για PyPDF2
wallia
def extractText(file, method):
    text = ""
    if method == 'M':
        # Άνοιγμα του αρχείου από τον reader της βιβλιοθήκη
        reader = Py.PdfFileReader(file, strict=False)
        # Εξαγωγή του κειμένου όλων των σελίδων του αρχείου στη μεταβλητή text
        for pageNum in range(reader.getNumPages()):
            page = reader.getPage(pageNum)
            text += page.extractText().strip()
    elif method == 'F':
        with fitz.open(file) as doc:
            for page in doc:
                text += page.get_text()
    return text


# Εξαγωγή του κειμένου ενός διαδικτυακού pdf αρχείου, βάσει του url του
wallia
def extract_urlPdfFile(pdf_url, method):
    # Εξαγωγή σε bytes των περιεχομένων του αρχείου
    r = requests.get(pdf_url)
    f = io.BytesIO(r.content)
    # Κλήση της κύριας συνάρτησης και εξαγωγή κειμένου
    return extractText(f, method)


# Εξαγωγή του κειμένου ενός pdf αρχείου, αποθηκευμένο στο δίσκο
wallia
def extract_localPdfFile(pdf_path, method):
    # Άνοιγμα του τοπικού αρχείου
    f = open(pdf_path, "rb")
    # Κλήση της κύριας συνάρτησης και εξαγωγή κειμένου
    return extractText(f, method)
```

Παράρτημα – Δ

Το παρακάτω τμήμα κώδικα, χρησιμοποιώντας συναρτήσεις πολυεπεξεργασίας της Python, κάνει προ-επεξεργασία κειμένου στα πεδία text (κείμενο) και subject (θέμα) κάθε πράξης αλλά και του θέματος, όπως αυτά βρίσκονται αποθηκευμένα στο αρχείο JSON.

```
elif operation == 4:
    # Εμφάνιση του αρχείου json στο οποίο θα γίνει επεξεργασία του κειμένου
    # Πρέπει να περιέχεται στο βασικό φάκελο
    data, filepath = openfile()

    with Pool() as pool:
        processed_data = pool.map(multi_textprocess, data)

    filepath = filepath + '_processed'
    with open(filepath + '.json', 'w', encoding='utf8') as json_file:
        json.dump(processed_data, json_file, ensure_ascii=False, indent=4)
    df = pd.DataFrame(processed_data)
    df.to_csv(filepath + '.csv', index=False, header=True)
```

```
def multi_textprocess(item):
    try:
        text = item['exText']
        subject = item['subject']
        text = pre.prePro(text)
        subject = pre.prePro(subject)
        item['exText'] = text
        item['subject'] = subject
        del item['datetime']
        print('Ok')
    except Exception:
        print("Problem", flush=True)
        pass
    return item
```

Οι παρακάτω συναρτήσεις εκτελούν ενέργειες που αφορούν την προ-επεξεργασία κειμένου.

```
wallia
def fix_capital_delta(text):
    for i in range(len(text)):
        text[i] = text[i].replace(u"\u2206", "\u0394")  # Το κεφαλαίο Δ: \xce\x94 ή U+0394
    return text


wallia
def removeAccent(text):
    d = {ord('\N{COMBINING ACUTE ACCENT}'): None}
    text = ud.normalize('NFD', text).translate(d)
    return text


new *
def removePunctuation(text):
    translator = str.maketrans('', '', string.punctuation)
    text = text.translate(translator)
    return text
```

```
wallia *
def removeStopWords(text):
    tokens = word_tokenize(text)
    text = " ".join([w for w in tokens if w.lower() not in STOP_WORDS])
    return text


wallia
def removeSingleChar(text):
    tokens = word_tokenize(text)
    text = " ".join([w for w in tokens if len(w) > 1])
    return text


wallia *
def prePro(text):
    text = removePunctuation(text)
    text = removeSingleChar(text)
    text = removeStopWords(text)
    text = removeAccent(text)
    text = text.split()
    text = fix_capital_delta(text)
    text = " ".join(text)
    return text.upper()
```

Παράρτημα – Ε

Ο παρακάτω κώδικας ελέγχει την ομοιότητα μεταξύ δυο λέξεων ώστε να δημιουργήσει ομάδες όμοιων λέξεων.

```
def SimilarWords(word1, word2, similarity=SimilarPercent):
    similar_words = False
    if similarity < 100:
        # Έλεγχος για παρόμοιες λέξεις
        ml = (len(word1) + len(word2)) / 2
        match_area = math.ceil(ml * similarity / 100)
        # Ελέγχουμε το 66% της λέξης (default similarity)
        if word1[:match_area] == word2[:match_area]:
            similar_words = True
    else:
        # Έλεγχος για πανομοιότυπες λέξεις (similarity=100)
        if word1 == word2:
            similar_words = True
    return similar_words
```

Παράρτημα - ΣΤ

Ο παρακάτω κώδικας αποτελεί τμήμα του κώδικα που επεξεργάζεται το εφαρμόζοντας επιλογή χαρακτηριστικών (feature selection), δημιουργία λεξικών και vectorization όσο αφορά την κατηγοριοποίηση - ταξινόμηση των πράξεων βάσει Θεματικής κατηγορίας.

```
def WordCount(text, cur_classes):
    global ClassPopulation, UniqueW, UniqueW_Freq, SimilarPercent, SimilarW, WAppear_inClasses
    # Για κάθε ένα έγγραφο, προετοιμάζεται το κείμενο...
    text = text.upper().split()
    # Για να αποθηκεύεται το id των λέξεων που εμφανίζονται στο επεξεργαζόμενο έγγραφο
    have_appeared = []
    # ...μετρούνται στο ClassPopulation οι θεματικές κατηγορίες του...
    for i in range(len(cur_classes)):
        for diction in ClassPopulation:
            if diction['label'] == cur_classes[i]:
                diction['count'] = diction.get('count', 0) + 1
    # ...και για κάθε μια λέξη...
    for i in range(len(text)):
        cur_word = text[i].strip()
        if IsWord(cur_word):
            # ...γίνεται αναζήτηση αν υπάρχει ή αν θα προστεθεί ως νέα.
            w_id = FindWordID(cur_word)
            # Ενημερώνονται:...
            # ...αν είναι νέα λέξη...
            if w_id == -1:
                # ...η λίστα με μοναδικές λέξεις...
                UniqueW.append(cur_word)
                # ...η συχνότητα εμφάνισης της,...
                UniqueW_Freq.append(1)
                # ...και η λίστα με παρόμοιες λέξεις...
                SimilarW.append({'base': cur_word, 'similar': []})
                # Το w_id της νέας λέξης είναι η τελευταία θέση της λίστας
                w_id = len(UniqueW)-1
                # Αρχικοποιήσεις για το WAppear_inClasses
                zeros = [0] * len(ClassPopulation)
                WAppear_inClasses.append(zeros)
            # ...αλλιώς αν υπάρχει ήδη...
            else:
                # ...ενημερώνεται η συχνότητα εμφάνισης της.
                UniqueW_Freq[w_id] += 1
            # Προσθέτει τη λέξη (αν δεν υπάρχει ακόμα και για τις νέες) στο πεδίο similar της λίστας SimilarW
            if SimilarPercent < 100 and cur_word not in SimilarW[w_id].get('similar'):
                SimilarW[w_id]['similar'].append(cur_word)
            # Προσθέτει το id της λέξης για την ενημέρωση του df που ακολουθεί
            if w_id not in have_appeared:
                have_appeared.append(w_id)
    # print('have', have_appeared)
    for word_id in have_appeared:
        # Για κάθε μια από τις θεματικές κατηγορίες...
        for j in range(len(cur_classes)):
            # ...που το κείμενο επομένως και η λέξη ανήκει, βρίσκεται η θέση της (χρήση του ClassPopulation)...
            index = find_pos_inlistOf_diction(ClassPopulation, 'label', cur_classes[j])
            # ...και μετριέται η εμφάνιση της λέξης στη συγκεκριμένη θεματική κατηγορία.
            WAppear_inClasses[word_id][index] += 1
```

```
def WordCountCalculations(population_of_categories):
    # Ονόματα στηλών DataFrame
    columns = []
    for d in ClassPopulation:
        columns.append(d.get('label'))
    # Ονόματα γραμμών DataFrame
    rows = []
    similar_words = []
    for d in SimilarW:
        rows.append(d.get('base'))
        similar_words.append(' '.join(d['similar']))
    # Δημιουργία αρχικού DataFrame
    df1 = pd.DataFrame(WAppear_inClasses, columns=columns, index=rows)
    # Προσθήκη στο DataFrame
    df1['Freq'] = UniqueW_Freq
    df1['Similar'] = similar_words
    # Αλλαγή της σειράς των στηλών
    df1 = df1[['Freq'] + [col for col in df1.columns if col != 'Freq']]
    df1 = df1[['Similar'] + [col for col in df1.columns if col != 'Similar']]

    # Έγγραφα ανά θεματική
    # Κατηγορίες θεματικών
    cp = [d.get('count', 0) for d in ClassPopulation]
    word_freq_ratio = []
    d_freq = []
    for line in WAppear_inClasses:
        word_calc = []
        for i in range(len(cp)):
            if cp[i] != 0:
                res = round(line[i] / cp[i], 5)
            else:
                res = 0
            word_calc.append(res)
        word_freq_ratio.append(word_calc)
        d_freq.append(sum(line))
    # Δημιουργία νέου DataFrame
    df2 = pd.DataFrame(word_freq_ratio, columns=columns, index=rows)
    # Συνένωση των δυο DataFrame
    df = pd.concat([df1, df2], axis=1)
    # Προσθήκη στο DataFrame
    df['DF'] = d_freq
```

```
    # Υπολογισμός DEVMAX.DF
    maxes = []
    for line in word_freq_ratio:
        maxes.append(max(line))
    abs_devmax = []
    for i in range(len(maxes)):
        abs_devmax.append(
            (len(active_thematics) * maxes[i] - sum(word_freq_ratio[i]))
            / ((len(active_thematics) - 1) * maxes[i]))
    devmax_df = [round(abs_devmax[i] * math.log10(d_freq[i]), 5) for i in range(len(abs_devmax))]
    # Προσθήκη στο DataFrame
    df['maxes'] = maxes
    df['abs_devmax'] = abs_devmax
    df['devmaxdf'] = devmax_df

    # Υπολογισμός Chi-square
    df_avg = [round(sum(line)/len(line), 5) for line in WAppear_inClasses]
    chi2 = []
    for i in range(len(df_avg)):
        value = df_avg[i]
        res = 0
        for j in range(len(WAppear_inClasses[i])):
            res += pow(WAppear_inClasses[i][j] - value, 2)
        res = round(res / value, 5)
        chi2.append(res)
    # Προσθήκη στο DataFrame
    df['df_avg'] = df_avg
    df['chi2'] = chi2

    for method in ['devmaxdf', 'chi2']:
        # Ταξινόμηση ως προς τη στήλη DEVMAX.DF ή Chi-square
        df.sort_values(by=[method], inplace=True, ascending=False)
        # Δημιουργία λεξικού και αποθήκευση σε json
        lexicon = list(df.index.values.tolist())
        filepath = localFolder + '/thematics_lexicon_' + method + '_' \
                   + str(population_of_categories) + '.json'
        with open(filepath, 'w', encoding='utf8') as json_file:
            json.dump(lexicon, json_file, ensure_ascii=False, indent=4)
        # Αποθήκευση σε csv
        filepath = localFolder + '/thematics_results_' + method + '_' \
                   + str(population_of_categories) + '.csv'
        # df.to_csv(filepath, encoding='utf-8')
        df.to_csv(filepath, encoding='utf-8-sig')
```

```
def DocumentTermAnalysis(ada, unit_id, dec_type, th1, th2, text, cur_classes):
    global DocumentId, UnitId, DocumentThematics, DocumentTerm
    # ...μετρούνται στο ClassPopulation οι θεματικές κατηγορίες του...
    count_thematics = [0] * len(active_thematics[:chosen_thematics])
    for i in range(len(cur_classes)):
        for j in range(len(active_thematics[:chosen_thematics])):
            if active_thematics[j] == cur_classes[i]:
                count_thematics[j] += 1
    DocumentId.append(ada)
    UnitId.append(unit_id)
    DecType.append(dec_type)
    Them1.append(th1)
    Them2.append(th2)
    Thematics.append(cur_classes)
    DocumentThematics.append(count_thematics)
    # Για κάθε ένα έγγραφο, προετοιμάζεται το κείμενο...
    text = text.upper().split()
    # ...αρχικοποιούμε για τις μετρήσεις στο DocumentTerm...
    counter = [0] * len(Lexicon)
    # ...και για κάθε μια λέξη ...
    for i in range(len(text)):
        cur_word = text[i].strip()
        if IsWord(cur_word):
            # ...γίνεται αναζήτηση αν υπάρχει...
            w_id = FindWordID_Lexicon(cur_word)
            # ...και αν υπάρχει στις λέξεις που ελέγχουμε...
            if w_id != -1:
                # ...ενημερώνεται η συχνότητα εμφάνισης της.
                counter[w_id] = 1
    DocumentTerm.append(counter)
```

```
# θεματικές κατηγορίες αποφάσεων εγγράφων
thematics = client_metadata.get_dictionary('THK')['items']
# Δημιουργία λεξικού για τις θεματικές κατηγορίες της μορφής:
# {'uid': '10', 'label': 'ΕΥΡΩΠΑΪΚΗ ΈΝΩΣΗ'}, {'uid': '16', 'label': 'ΟΙΚΟΝΟΜΙΚΗ ΖΩΗ'},
# {'uid': '20', 'label': 'ΟΙΚΟΝΟΜΙΚΕΣ ΚΑΙ ΕΜΠΟΡΙΚΕΣ ΣΥΝΑΛΛΑΓΕΣ'}, {'uid': '24', 'label': 'ΔΗΜΟΣΙΟΝΟΜΙΚΑ'},
# {'uid': '32', 'label': 'ΕΠΙΚΟΙΝΩΝΙΑ ΚΑΙ ΜΟΡΦΩΣΗ'}, {'uid': '36', 'label': 'ΕΠΙΣΤΗΜΕΣ'},
# {'uid': '40', 'label': 'ΕΠΙΧΕΙΡΗΣΕΙΣ ΚΑΙ ΑΝΤΑΓΩΝΙΣΜΟΣ'}, {'uid': '44', 'label': 'ΑΠΑΣΧΟΛΗΣΗ ΚΑΙ ΕΡΓΑΣΙΑ'},
# {'uid': '64', 'label': 'ΠΑΡΑΓΩΓΗ, ΤΕΧΝΟΛΟΓΙΑ ΚΑΙ ΕΡΕΥΝΑ'}, {'uid': '66', 'label': 'ΕΝΕΡΓΕΙΑ'},
# {'uid': '10004', 'label': 'ΔΗΜΟΣΙΑ ΔΙΟΙΚΗΣΗ'}, {'uid': '10007', 'label': 'ΥΓΕΙΑ'},
# {'uid': '10015', 'label': 'ΑΠΟΦΑΣΗ ΔΙΑΘΕΣΗΣ ΑΝΟΙΚΤΩΝ ΔΕΔΟΜΕΝΩΝ'},
# {'uid': '10027', 'label': 'ΔΑΠΑΝΕΣ ΕΠΙΧΟΡΗΓΟΥΜΕΝΩΝ ΦΟΡΕΩΝ ΑΡΘΡΟΥ 10Β Ν 3861/10'}

# θεματικές κατηγορίες...
ClassPopulation = []
chosen_thematics = 14
# ...σύμφωνα με το αρχείο active_thematics, στο οποίο έχουν καταγραφεί οι θεματικές κατηγορίες των εγγράφων...
# ...μετά από ανάλυση των μεταδεδομένων τους...
with open(localFolder + '/active_thematics.json', encoding='utf-8') as f:
    active_thematics = json.load(f)
# ...διατηρούμε από το λεξικό που έχουμε 'κατεβάσει' από τη Διαύγεια, τις θεματικές που μας ενδιαφέρουν...
for item in thematics:
    # ... διαγράφοντας ότι είναι περιττό...
    del item['parent']
    if item['label'] in active_thematics[:chosen_thematics]:  # Επιλογή των θεματικών με το μεγαλύτερο πλήθος εγγραφών
        ClassPopulation.append(item)
```

Παράρτημα - Z

Στο παρακάτω τμήμα κώδικα παρουσιάζεται η υλοποίησης εφαρμογής των μεθόδων μηχανικής και βαθιάς μάθησης στην κατηγοριοποίηση των πράξεων ως προς την Θεματική κατηγορία τους.

In [7]:
```
import pandas as pd
import _pickle as pkl
from retrievemetadata import *
import time
import bz2
from scipy.sparse import csr_matrix
from sklearn.metrics import f1_score, precision_score, recall_score, multilabel_confusion_matrix
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.multiclass import OneVsRestClassifier
from sklearn.tree import DecisionTreeClassifier
from skmultilearn.model_selection import IterativeStratification
```

In [8]:
```
with open(localFolder + '/active_thematics.json', encoding='utf-8') as f:
    mlabels = json.load(f)
```

In [9]:
```
filepath = localFolder + '/weights_allthematics.json'
with open(filepath, 'rb') as fp:
    weights = json.load(fp)
```

In [10]:
```
def train_model(model, x_train, x_test, y_train, y_test, deep = False):
    if deep:
        model.fit(x_train, y_train, epochs=10, batch_size=32, verbose=0)
    else:
        model.fit(x_train, y_train)
    y_pred = model.predict(x_test)
    if deep:
        # Convert the predicted probabilities to binary labels for the current fold
        y_pred = (y_pred > 0.5).astype(int)
    score = f1_score(y_test, y_pred, zero_division=0, average = None)
    pr = precision_score(y_test, y_pred, zero_division=0, average = None)
    re = recall_score(y_test, y_pred, zero_division=0, average = None)
    cm = multilabel_confusion_matrix(y_test, y_pred)
    return score, pr, re, cm
```

```
methods = ['devmaxdf', 'chi2']
lex_length = [50, 75] + [x for x in range(100,1050,50)] + [x for x in range(1200,4200,200)]

for method in methods:
    # Φόρτωση του κατάλληλου λεξικού
    name = localFolder + '/thematics_analysis_' + method  + '_15000_14_TRAIN'
    with bz2.BZ2File(name + '.pbz2', 'rb') as input_file:
        data = pkl.load(input_file)

    for length in lex_length:
        print(f'Running for vector size: {length}')
        start = time.time()

        X = data.iloc[:, :length].values
        y = data.iloc[:, -14:].values
        res=[]
        for i in range(len(classifiers)):
            res.append([clf_name[i], method, length])
        f1_per_label = [0]*len(classifiers)
        pr_per_label = [0]*len(classifiers)
        re_per_label = [0]*len(classifiers)
        prf1_per_label = [0]*len(classifiers)
        confm = [0]*len(classifiers)

        kf = IterativeStratification(n_splits=10, order=1)
        for train_index, test_index in kf.split(X, y):
            print('New Fold')

            X_train, X_test = X[train_index], X[test_index]
            y_train, y_test = y[train_index], y[test_index]

            for i in range(len(classifiers)):
                score, pr, re, cm = train_model(classifiers[i], csr_matrix(X_train), csr_matrix(X_test),
                                                csr_matrix(y_train), csr_matrix(y_test))
                f1_per_label[i] += score
                pr_per_label[i] += pr
                re_per_label[i] += re
                prf1_per_label[i] += score*pr
                confm[i] += cm

        for i in range(len(classifiers)):
            f1_per_label[i] /=10
            pr_per_label[i] /=10
            re_per_label[i] /=10
            prf1_per_label[i] /= 10
            f1_weighted = 0
            pr_weighted = 0
            re_weighted = 0
            prf1_weighted = 0
            for j in range(len(weights)):
                f1_weighted += f1_per_label[i][j] * weights[j]
                pr_weighted += pr_per_label[i][j] * weights[j]
                re_weighted += re_per_label[i][j] * weights[j]
                prf1_weighted += prf1_per_label[i][j] * weights[j]
                res[i].append(f1_per_label[i][j])
            res[i].append(f1_weighted)
            res[i].append(pr_weighted)
            res[i].append(re_weighted)
            res[i].append(prf1_weighted)
            res[i].append(confm)

        for i in range(len(classifiers)):
            results[i].append(res[i])
        end = time.time()
        print('Finished in: ',(end - start)/60,' min','\n')
```

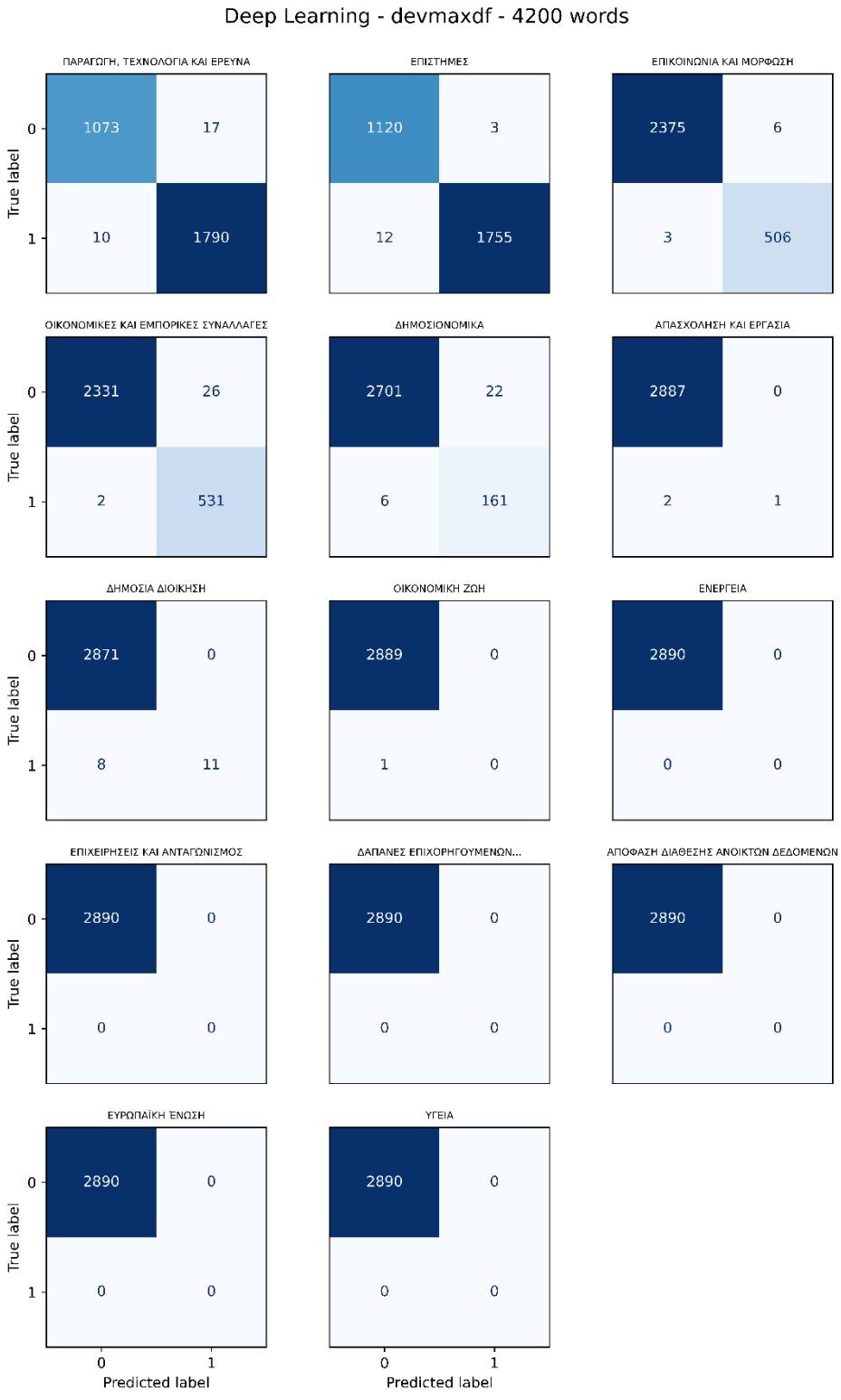
```
lex_length = [50, 75] + [x for x in range(100,1050,50)] + [x for x in range(1200,4200,200)]
methods = ['devmaxdf', 'chi2']

for method in methods:
    # Φόρτωση του κατάλληλου λεξικού
    name = localFolder + '/thematics_analysis_' + method  + '_15000_14_TRAIN'
    with bz2.BZ2File(name + '.pbz2', 'rb') as input_file:
        data = pkl.load(input_file)
    for length in lex_length:
        print(f'Running for vector size: {length}')
        start = time.time()
        X = data.iloc[:, :length].values
        y = data.iloc[:, -14:].values
        model = tf.keras.Sequential()
        model.add(tf.keras.layers.Dense(256, activation='relu', input_shape=(length,)))
        model.add(tf.keras.layers.Dropout(0.5))
        model.add(tf.keras.layers.Dense(128, activation='relu'))
        model.add(tf.keras.layers.Dropout(0.5))
        model.add(tf.keras.layers.Dense(64, activation='relu'))
        model.add(tf.keras.layers.Dense(14, activation='sigmoid'))
        model.compile(loss='binary_crossentropy', optimizer='adam')
        res=[]
        for i in range(len(classifiers)):
            res.append([clf_name[i], method, length])
        f1_per_label = [0]
        pr_per_label = [0]
        re_per_label = [0]
        prf1_per_label = [0]
        confm = [0]
        kf = IterativeStratification(n_splits=10, order=1)
        for train_index, test_index in kf.split(X, y):
            print('New Fold')
            X_train, X_test = X[train_index], X[test_index]
            y_train, y_test = y[train_index], y[test_index]
            for i in range(len(classifiers)):
                score, pr, re, cm = train_model(model, X_train, X_test, y_train, y_test, deep = True)
                f1_per_label[i] += score
                pr_per_label[i] += pr
                re_per_label[i] += re
                prf1_per_label[i] += score*pr
                confm[i] += cm
        for i in range(len(classifiers)):
            f1_per_label[i] /=10
            pr_per_label[i] /=10
            re_per_label[i] /=10
            prf1_per_label[i] /= 10
            f1_weighted = 0
            pr_weighted = 0
            re_weighted = 0
            prf1_weighted = 0
            for j in range(len(weights)):
                f1_weighted += f1_per_label[i][j] * weights[j]
                pr_weighted += pr_per_label[i][j] * weights[j]
                re_weighted += re_per_label[i][j] * weights[j]
                prf1_weighted += prf1_per_label[i][j] * weights[j]
                res[i].append(f1_per_label[i][j])
            res[i].append(f1_weighted)
            res[i].append(pr_weighted)
            res[i].append(re_weighted)
            res[i].append(prf1_weighted)
            res[i].append(confm)
        for i in range(len(classifiers)):
            results[i].append(res[i])
        end = time.time()
        print('Finished in: ',(end - start)/60,' min','\n')
```

Παράρτημα – Η

Ο παρακάτω πίνακας και τα confusion matrices απεικονίζουν τα αποτελέσματα του Deep Learning μοντέλου με την Devmax.df μετρική στην validation phase της κατηγοριοποίησης ανά «Θεματική κατηγορία» και αφορούν τα αποτελέσματα που παρουσιάζονται στη παράγραφο 4.1.3.3 καθώς και στο κεφάλαιο των συμπερασμάτων (Κεφάλαιο 5).

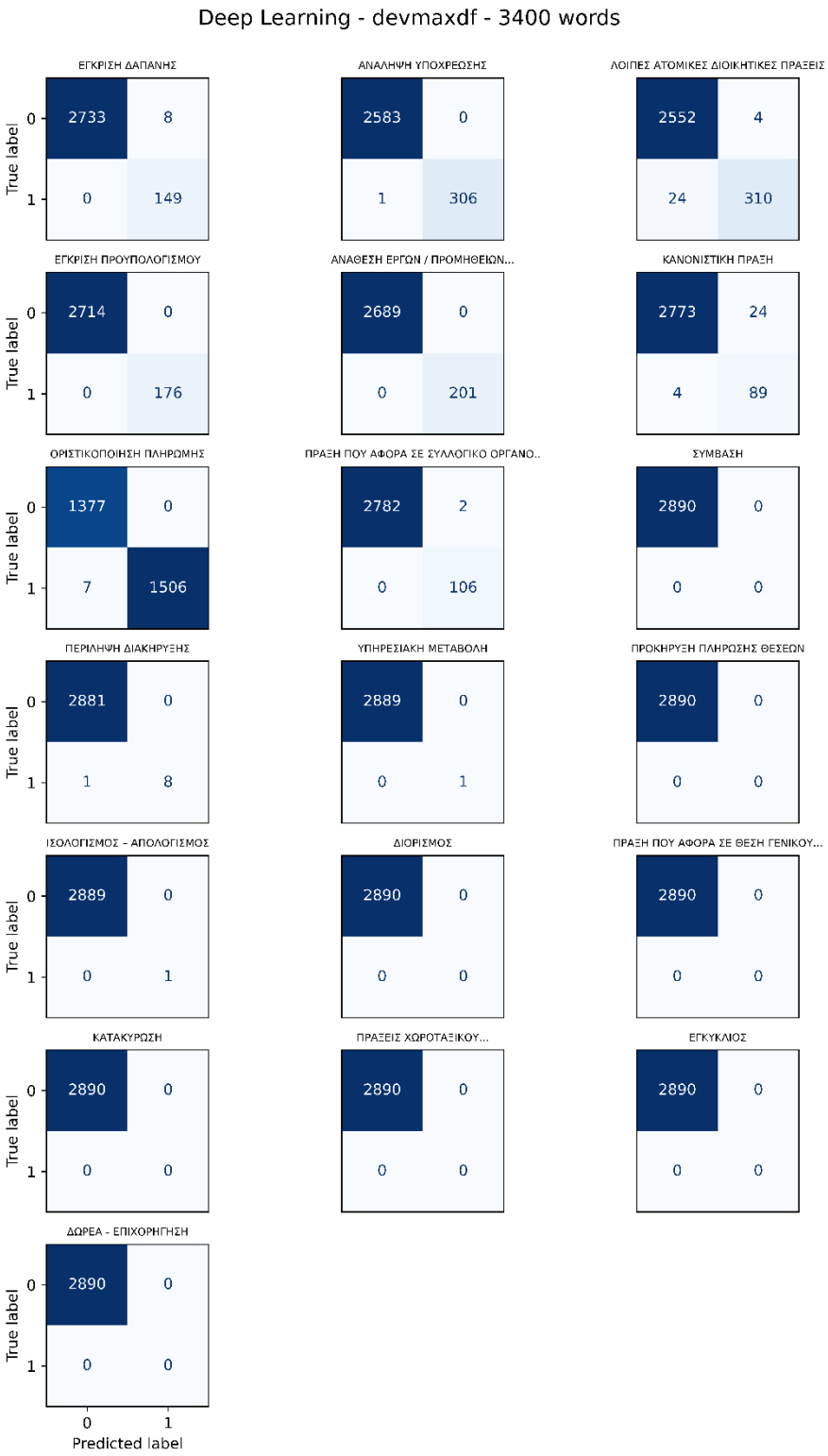
DEVMAX.DF – 4200W – DEEP LEARNING	**TN**	**FN**	**TP**	**FP**	**Precision**	**F1**	**Εκχωρήσεις / κλάση**
ΠΑΡΑΓΩΓΗ, ΤΕΧΝΟΛΟΓΙΑ ΚΑΙ ΕΡΕΥΝΑ	1073	10	1790	17	0.9906	0.9925	1800
ΕΠΙΣΤΗΜΕΣ	1120	12	1755	3	0.9983	0.9957	1767
ΕΠΙΚΟΙΝΩΝΙΑ ΚΑΙ ΜΟΡΦΩΣΗ	2375	3	506	6	0.9883	0.9912	509
ΟΙΚΟΝΟΜΙΚΕΣ ΚΑΙ ΕΜΠΟΡΙΚΕΣ ΣΥΝΑΛΛΑΓΕΣ	2331	2	531	26	0.9533	0.9743	533
ΔΗΜΟΣΙΟΝΟΜΙΚΑ	2701	6	161	22	0.8798	0.9200	167
ΑΠΑΣΧΟΛΗΣΗ ΚΑΙ ΕΡΓΑΣΙΑ	2887	2	1	0	1.0000	0.5000	3
ΔΗΜΟΣΙΑ ΔΙΟΙΚΗΣΗ	2871	8	11	0	1.0000	0.7333	19
ΟΙΚΟΝΟΜΙΚΗ ΖΩΗ	2889	1	0	0	-	-	1
ΕΝΕΡΓΕΙΑ	2890	0	0	0	-	-	0
ΕΠΙΧΕΙΡΗΣΕΙΣ ΚΑΙ ΑΝΤΑΓΩΝΙΣΜΟΣ	2890	0	0	0	-	-	0
ΔΑΠΑΝΕΣ ΕΠΙΧ/ΝΩΝ ΦΟΡΕΩΝ ΑΡΘΡΟΥ 10Β Ν 3861/10	2890	0	0	0	-	-	0
ΑΠΟΦΑΣΗ ΔΙΑΘΕΣΗΣ ΑΝΟΙΚΤΩΝ ΔΕΔΟΜΕΝΩΝ	2890	0	0	0	-	-	0
ΕΥΡΩΠΑΪΚΗ ΈΝΩΣΗ	2890	0	0	0	-	-	0
ΥΓΕΙΑ	2890	0	0	0	-	-	0
Σταθμισμένος ΜΟ:					**wP 0.9850**	**wF1 0.9874**	Σύνολο: 4799
					wP*F1 0.9732		


Deep Learning - devmaxdf - 4200 words
ΠΑΡΑΓΩΓΗ, ΤΕΧΝΟΛΟΓΙΑ ΚΑΙ ΕΡΕΥΝΑ
1073
17
10
1790
ΕΠΙΣΤΗΜΕΣ
1120
3
12
1755
ΕΠΙΚΟΙΝΩΝΙΑ ΚΑΙ ΜΟΡΦΩΣΗ
2375
6
3
506
ΟΙΚΟΝΟΜΙΚΕΣ ΚΑΙ ΕΜΠΟΡΙΚΕΣ ΣΥΝΑΛΛΑΓΕΣ
2331
26
2
531
ΔΗΜΟΣΙΟΝΟΜΙΚΑ
2701
22
6
161
ΑΠΑΣΧΟΛΗΣΗ ΚΑΙ ΕΡΓΑΣΙΑ
2887
0
2
1
ΔΗΜΟΣΙΑ ΔΙΟΙΚΗΣΗ
2871
0
8
11
ΟΙΚΟΝΟΜΙΚΗ ΖΩΗ
2889
0
1
0
ΕΝΕΡΓΕΙΑ
2890
0
0
0
ΕΠΙΧΕΙΡΗΣΕΙΣ ΚΑΙ ΑΝΤΑΓΩΝΙΣΜΟΣ
2890
0
0
0
ΔΑΠΑΝΕΣ ΕΠΙΧΟΡΗΓΟΥΜΕΝΩΝ...
2890
0
0
0
ΑΠΟΦΑΣΗ ΔΙΑΘΕΣΗΣ ΑΝΟΙΚΤΩΝ ΔΕΔΟΜΕΝΩΝ
2890
0
0
0
ΕΥΡΩΠΑΪΚΗ ΕΝΩΣΗ
2890
0
0
0
ΥΓΕΙΑ
2890
0
0
0
True label
Predicted label
0
1

Παράρτημα – Θ

Ο παρακάτω πίνακας και τα confusion matrices απεικονίζουν τα αποτελέσματα του Deep Learning μοντέλου με την Devmax.df μετρική στην validation phase της κατηγοριοποίησης ανά «Είδος πράξης» και αφορούν τα αποτελέσματα που παρουσιάζονται στη παράγραφο 4.2.3.3 καθώς και στο κεφάλαιο των συμπερασμάτων (Κεφάλαιο 5).

DEVMAX.DF – 3400W – DEEP LEARNING	**TN**	**FN**	**TP**	**FP**	**Precision**	**F1**	**Εκχωρήσεις / κλάση**
ΕΓΚΡΙΣΗ ΔΑΠΑΝΗΣ	2733	0	149	8	0.9490	0.9739	149
ΑΝΑΛΗΨΗ ΥΠΟΧΡΕΩΣΗΣ	2583	1	306	0	1.0000	0.9984	307
ΛΟΙΠΕΣ ΑΤΟΜΙΚΕΣ ΔΙΟΙΚΗΤΙΚΕΣ ΠΡΑΞΕΙΣ	2552	24	310	4	0.9873	0.9568	334
ΕΓΚΡΙΣΗ ΠΡΟΥΠΟΛΟΓΙΣΜΟΥ	2714	0	176	0	1.0000	1.0000	176
ΑΝΑΘΕΣΗ ΕΡΓΩΝ/ΠΡΟΜΗΘΕΙΩΝ/ΥΠΗΡΕΣΙΩΝ/ΜΕΛΕΤΩΝ	2689	0	201	0	1.0000	1.0000	201
ΚΑΝΟΝΙΣΤΙΚΗ ΠΡΑΞΗ	2773	4	89	24	0.7876	0.8641	93
ΟΡΙΣΤΙΚΟΠΟΙΗΣΗ ΠΛΗΡΩΜΗΣ	1377	7	1506	0	1.0000	0.9977	1513
ΠΡΑΞΗ ΠΟΥ ΑΦΟΡΑ ΣΕ ΣΥΛΛΟΓΙΚΟ ΟΡΓΑΝΟ – ΕΠΙΤΡΟΠΗ - ΟΜΑΔΑ ΕΡΓΑΣΙΑΣ - ΟΜΑΔΑ ΕΡΓΟΥ - ΜΕΛΗ ΣΥΛΛΟΓΙΚΟΥ ΟΡΓΑΝΟΥ	2782	0	106	2	0.9815	0.9907	106
ΣΥΜΒΑΣΗ	2890	0	0	0	-	-	0
ΠΕΡΙΛΗΨΗ ΔΙΑΚΗΡΥΞΗΣ	2881	1	8	0	1.0000	0.9412	9
ΥΠΗΡΕΣΙΑΚΗ ΜΕΤΑΒΟΛΗ	2889	0	1	0	1.0000	1.0000	1
ΠΡΟΚΗΡΥΞΗ ΠΛΗΡΩΣΗΣ ΘΕΣΕΩΝ	2890	0	0	0	-	-	0
ΙΣΟΛΟΓΙΣΜΟΣ – ΑΠΟΛΟΓΙΣΜΟΣ	2889	0	1	0	1.0000	1.0000	1
ΔΙΟΡΙΣΜΟΣ	2890	0	0	0	-	-	0
ΠΡΑΞΗ ΠΟΥ ΑΦΟΡΑ ΣΕ ΘΕΣΗ ΓΕΝΙΚΟΥ - ΕΙΔΙΚΟΥ ΓΡΑΜΜΑΤΕΑ - ΜΟΝΟΜΕΛΕΣ ΟΡΓΑΝΟ	2890	0	0	0	-	-	0
ΚΑΤΑΚΥΡΩΣΗ	2890	0	0	0	-	-	0
ΠΡΑΞΕΙΣ ΧΩΡΟΤΑΞΙΚΟΥ - ΠΟΛΕΟΔΟΜΙΚΟΥ ΠΕΡΙΕΧΟΜΕΝΟΥ	2890	0	0	0	-	-	0
ΕΓΚΥΚΛΙΟΣ	2890	0	0	0	-	-	0
ΔΩΡΕΑ - ΕΠΙΧΟΡΗΓΗΣΗ	2890	0	0	0	-	-	0
Σταθμισμένος ΜΟ:					**wP 0.9883**	**wF1 0.9872**	Σύνολο: 2890
					wP*F1 0.9768		


Deep Learning - devmaxdf - 3400 words
ΕΓΚΡΙΣΗ ΔΑΠΑΝΗΣ
2733
8
0
149
ΑΝΑΛΗΨΗ ΥΠΟΧΡΕΩΣΗΣ
2583
0
1
306
ΛΟΙΠΕΣ ΑΤΟΜΙΚΕΣ ΔΙΟΙΚΗΤΙΚΕΣ ΠΡΑΞΕΙΣ
2552
4
24
310
ΕΓΚΡΙΣΗ ΠΡΟΥΠΟΛΟΓΙΣΜΟΥ
2714
0
0
176
ΑΝΑΘΕΣΗ ΕΡΓΩΝ / ΠΡΟΜΗΘΕΙΩΝ...
2689
0
0
201
ΚΑΝΟΝΙΣΤΙΚΗ ΠΡΑΞΗ
2773
24
4
89
ΟΡΙΣΤΙΚΟΠΟΙΗΣΗ ΠΛΗΡΩΜΗΣ
1377
0
7
1506
ΠΡΑΞΗ ΠΟΥ ΑΦΟΡΑ ΣΕ ΣΥΛΛΟΓΙΚΟ ΟΡΓΑΝΟ..
2782
2
0
106
ΣΥΜΒΑΣΗ
2890
0
0
0
ΠΕΡΙΛΗΨΗ ΔΙΑΚΗΡΥΞΗΣ
2881
0
1
8
ΥΠΗΡΕΣΙΑΚΗ ΜΕΤΑΒΟΛΗ
2889
0
0
1
ΠΡΟΚΗΡΥΞΗ ΠΛΗΡΩΣΗΣ ΘΕΣΕΩΝ
2890
0
0
0
ΙΣΟΛΟΓΙΣΜΟΣ – ΑΠΟΛΟΓΙΣΜΟΣ
2889
0
0
1
ΔΙΟΡΙΣΜΟΣ
2890
0
0
0
ΠΡΑΞΗ ΠΟΥ ΑΦΟΡΑ ΣΕ ΘΕΣΗ ΓΕΝΙΚΟΥ...
2890
0
0
0
ΚΑΤΑΚΥΡΩΣΗ
2890
0
0
0
ΠΡΑΞΕΙΣ ΧΩΡΟΤΑΞΙΚΟΥ...
2890
0
0
0
ΕΓΚΥΚΛΙΟΣ
2890
0
0
0
ΔΩΡΕΑ - ΕΠΙΧΟΡΗΓΗΣΗ
2890
0
0
0
True label
Predicted label
0
1