ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

Αναφορά εργαστηριακής άσκησης 2025

ΙΩΑΝΝΗΣ ΜΠΟΥΖΑΣ
ΑΜ:5025

Εισαγωγή

Ο στόχος αυτής της άσκησης είναι να αναπτυχθεί ένα σύστημα αποφάσεων που ταξινομεί εικόνες προσώπου σε δύο κατηγορίες (δυαδική ταξινόμηση). Θα υλοποιηθούν και θα συγκριθούν πολλαπλές μεθοδολογίες ταξινόμησης, αναλύοντας τον αντίκτυπο των διαφορετικών προσεγγίσεων στο χειρισμό πληροφοριών που βασίζονται σε εικόνες.

Direct Image-Based Classification using a CNN

Στην πρώτο κομμάτι αυτής της εργασίας θα χρησιμοποιήσουμε διάφορα CNN μοντέλα κατευθείαν στις εικόνες και θα συγκρίνουμε τις αρχιτεκτονικές και τις αποδόσεις τους.

Πιο συγκεκριμένα για την προεπεξεργασία των εικόνων χρησιμοποιήθηκε η συνάρτηση ImageDataGenerator της βιβλιοθήκης Keras. Στο σύνολο εκπαίδευσης εφαρμόστηκαν τεχνικές ενίσχυσης δεδομένων (data augmentation), όπως τυχαία περιστροφή, μετατόπιση, μεγέθυνση, αναστροφή και παραμόρφωση, ώστε να αυξηθεί η ποικιλομορφία των εισόδων και να βελτιωθεί η γενίκευση του μοντέλου. Επιπλέον, έγινε κανονικοποίηση των τιμών των pixels στο διάστημα [0, 1] και κρατήθηκε το 20% των δεδομένων ως σύνολο επικύρωσης. Αντίθετα, στο σύνολο δοκιμής εφαρμόστηκε μόνο κανονικοποίηση, χωρίς ενίσχυση, για αντικειμενική αξιολόγηση της απόδοσης του μοντέλου.

Για τη φόρτωση των εικόνων χρησιμοποιήθηκε η μέθοδος flow_from_directory, η οποία επιτρέπει την αυτόματη δημιουργία παρτίδων (batches) εικόνων από φακέλους. Ο φάκελος περιέχει δύο υποφακέλους (unpleasant και pleasant), οι οποίοι καθορίζουν τις κατηγορίες. Το σύνολο εκπαίδευσης (train_generator) περιλαμβάνει τυχαία ανακατεμένα δείγματα και αντιστοιχεί στο 80% των δεδομένων, ενώ το σύνολο επικύρωσης (val_ds) περιλαμβάνει το υπόλοιπο 20%, χωρίς ανακάτεμα, ώστε να είναι σταθερό κατά την αξιολόγηση. Και στις δύο περιπτώσεις, οι εικόνες μετατρέπονται τόσο σε διαστάσεις 64x64 αλλά και σε 128x128 και φορτώνονται σε παρτίδες των 64 δειγμάτων με δυαδικό τρόπο κατηγοριοποίησης (class_mode='binary'), δεδομένου ότι το πρόβλημα είναι δυαδικής ταξινόμησης.

Έπειτα η συνάρτηση train_and_evaluate υλοποιεί τη διαδικασία εκπαίδευσης και αξιολόγησης ενός μοντέλου Keras. Αρχικά, γίνεται μεταγλώττιση του μοντέλου με τον βελτιστοποιητή Adam και συνάρτηση κόστους binary_crossentropy, κατάλληλη για δυαδικά προβλήματα ταξινόμησης. Στη συνέχεια, το μοντέλο εκπαιδεύεται με τα δεδομένα του συνόλου εκπαίδευσης και επικυρώνεται με το σύνολο επικύρωσης για έναν προκαθορισμένο αριθμό εποχών. Μετά την εκπαίδευση, γίνεται πρόβλεψη των κλάσεων στο validation set και υπολογίζονται βασικοί μετρικοί δείκτες απόδοσης (accuracy, precision, recall, F1 score), καθώς και πλήρης αναφορά ταξινόμησης (classification report), αξιοποιώντας τις πραγματικές και προβλεπόμενες ετικέτες.

Όπως αναφέραμε παραπάνω τα μοντέλα μας εκπαιδεύτηκαν και εξετάστηκαν σε εικόνες μεγέθους τόσο 64x64 όσο και 128x128.

Ακολουθεί η περιγραφή των μοντέλων για τα δυο μεγέθη εικόνων

1. Μοντέλο 1 (Απλό – Μονοστρωματικό Συνελικτικό Δίκτυο)
Δομή: Περιλαμβάνει ένα συνελικτικό επίπεδο με 32 φίλτρα και πυρήνα 3x3, ακολουθούμενο από flatten και ένα πλήρως συνδεδεμένο επίπεδο εξόδου.
Χαρακτηριστικά: Πολύ απλό και ρηχό δίκτυο χωρίς κανονικοποίηση ή dropout.
Χρήση: Ιδανικό για μικρά ή απλά datasets, κατάλληλο για γρήγορο πειραματισμό ή ως βασική γραμμή σύγκρισης (baseline).
Πλεονεκτήματα: Ταχύτατη εκπαίδευση, απλή υλοποίηση.
Μειονεκτήματα: Υψηλός κίνδυνος υπερεκπαίδευσης (overfitting), αδύναμη γενίκευση σε σύνθετα δεδομένα.

Για 64x64 εικόνες έχουμε:

Total params: 123,905 (484.00 KB)

Accuracy: 0.7292

Precision: 0.8101

Recall: 0.6416

F1 Score: 0.7160

Ενώ για 128x128:

Total params: 508,929 (1.94 MB)

Accuracy: 0.7292

Accuracy: 0.7111

Precision: 0.7023

Recall: 0.7935

F1 Score: 0.7452

2. Μοντέλο 2 (Περισσότερα Επίπεδα, Χωρίς Κανονικοποίηση)
Δομή: Περιλαμβάνει δύο συνελικτικά επίπεδα (το δεύτερο με 64 φίλτρα), ακολουθούμενα από max pooling, flatten και έξοδο.
Χαρακτηριστικά: Αυξημένο βάθος αλλά χωρίς dropout ή batch normalization.
Χρήση: Κατάλληλο για ελαφρώς πιο σύνθετα δεδομένα.
Πλεονεκτήματα: Καλύτερη εξαγωγή χαρακτηριστικών σε σχέση με το μοντέλο 1.
Μειονεκτήματα: Κίνδυνος υπερεκπαίδευσης λόγω έλλειψης regularization.

Εικόνες 64x64:

Total params: 31,649 (123.63 KB)

Accuracy: 0.7276

Precision: 0.7819

Recall: 0.6770

F1 Score: 0.7257

Εικόνες 128x128:

Total params: 127,905 (499.63 KB)

Accuracy: 0.7480

Precision: 0.7829

Recall: 0.7286

F1 Score: 0.7548

3. Μοντέλο 3 (Δύο Συνελικτικά Blocks + Dropout)
Δομή: Δύο συνελικτικά blocks (32 και 64 φίλτρα), max pooling και dropout για regularization.
Χαρακτηριστικά: Καλύτερη γενίκευση λόγω dropout.
Χρήση: Κατάλληλο για datasets μεσαίας πολυπλοκότητας.
Πλεονεκτήματα: Περιορίζει την υπερεκπαίδευση.
Μειονεκτήματα: Ίσως ανεπαρκές για πολύ μεγάλα ή σύνθετα δεδομένα.

Εικόνες 64x64:

Total params: 31,937 (124.75 KB)

Accuracy: 0.7363

Precision: 0.7485

Recall: 0.7596

F1 Score: 0.7540

Εικόνες 128x128:

Total params: 76,993 (300.75 KB)

Accuracy: 0.7316

Precision: 0.7577

Recall: 0.7286

F1 Score: 0.7429

4. Μοντέλο 4 (Batch Normalization + Dropout)
Δομή: Περιλαμβάνει batch normalization μετά από κάθε block και dropout. Προστίθεται ένα dense επίπεδο 64 νευρώνων.
Χαρακτηριστικά: Σταθερή εκπαίδευση και καλή γενίκευση.
Χρήση: Αποτελεσματικό σε μεσαίας πολυπλοκότητας δεδομένα.
Πλεονεκτήματα: Βελτιώνει τη σταθερότητα και μειώνει το overfitting.
Μειονεκτήματα: Περισσότερες παραμέτρους, άρα πιθανώς μεγαλύτερος χρόνος εκπαίδευσης.

Εικόνες 64x64:

Total params: 822,721 (3.14 MB)

Accuracy: 0.7143

Precision: 0.7929

Recall: 0.6268

F1 Score: 0.7002

Εικόνες 128x128:

Total params: 3,706,305 (14.14 MB)

Accuracy: 0.7308

Precision: 0.7467

Recall: 0.7478

F1 Score: 0.7472

5. Μοντέλο 5 (Μεγαλύτερο Δίκτυο – 3 Blocks + Dropout)
Δομή: Τρία συνελικτικά blocks (32, 64, 128 φίλτρα), με χρήση batch normalization και dropout.
Χαρακτηριστικά: Δυνατό στην εξαγωγή χαρακτηριστικών, καλύπτει πιο πολύπλοκα patterns.
Χρήση: Ιδανικό για μεγάλα ή πολύπλοκα σύνολα δεδομένων.
Πλεονεκτήματα: Ισχυρό δίκτυο, καλή γενίκευση.
Μειονεκτήματα: Υψηλός υπολογιστικός φόρτος.

Εικόνες 64x64:

Total params: 684,225 (2.61 MB)

Accuracy: 0.7370

Precision: 0.6942

Recall: 0.9041

F1 Score: 0.7854

Εικόνες 128x128:

Total params: 3,305,665 (12.61 MB)

Accuracy: 0.6774

Precision: 0.9033

Recall: 0.4410

F1 Score: 0.5927

6. Μοντέλο 6 (Προχωρημένη Αρχιτεκτονική με Πολλαπλές Συνελικτικές Στρώσεις)
Δομή: Πέντε συνελικτικά επίπεδα με padding, batch normalization και dropout.
Χαρακτηριστικά: Διατηρεί χωρική πληροφορία, προσφέρει βαθιά εξαγωγή χαρακτηριστικών.
Χρήση: Ιδανικό για datasets με μεγάλη ποικιλία και βάθος.
Πλεονεκτήματα: Αντιμετωπίζει αποτελεσματικά σύνθετα patterns.
Μειονεκτήματα: Απαιτεί περισσότερο χρόνο και υπολογιστικούς πόρους.

Εικόνες 64x64:

Total params: 1,188,257 (4.53 MB)

Accuracy: 0.7418

Precision: 0.7616

Recall: 0.7493

F1 Score: 0.7554

Εικόνες 128x128:

Total params: 4,333,985 (16.53 MB)

Accuracy: 0.7496

Precision: 0.7724

Recall: 0.7507

F1 Score: 0.7614

7. Μοντέλο 7 (Πιο Σύνθετο – 6 Συνελικτικές Στρώσεις, BatchNorm Παντού)
Δομή: Έξι συνελικτικά επίπεδα, εκτενής χρήση batch normalization και dropout. Το τελευταίο block έχει 256 φίλτρα.
Χαρακτηριστικά: Πολύ ισχυρή αρχιτεκτονική με έμφαση στην κανονικοποίηση.
Χρήση: Ιδανικό για ιδιαίτερα απαιτητικά datasets.
Πλεονεκτήματα: Υψηλή ακρίβεια σε δύσκολα tasks.
Μειονεκτήματα: Πολύ αργή εκπαίδευση, απαιτεί ισχυρούς υπολογιστικούς πόρους.

Εικόνες 64x64:

Total params: 9,604,673 (36.64 MB)

Accuracy: 0.7370

Precision: 0.8046

Recall: 0.6681

F1 Score: 0.7301

Εικόνες 128x128:

Total params: 34,770,497 (132.64 MB)

Accuracy: 0.7441

Precision: 0.7262

Recall: 0.8333

F1 Score: 0.7761

8. Μοντέλο 8 (Ισορροπημένη Πολυπλοκότητα με Dropout και BatchNorm)
Δομή: Τρία συνελικτικά blocks (32 και 64 φίλτρα), με batch normalization και dropout.
Χαρακτηριστικά: Καλή ισορροπία μεταξύ βάθους και regularization.
Χρήση: Ιδανικό για datasets μέτριας πολυπλοκότητας.
Πλεονεκτήματα: Σταθερή απόδοση, μικρότερος χρόνος εκπαίδευσης από τα βαθύτερα μοντέλα.
Μειονεκτήματα: Δεν φτάνει την απόδοση μοντέλων όπως το 7 σε πολύ δύσκολα προβλήματα.

Εικόνες 64x64:

Total params: 582,017 (2.22 MB)

Accuracy: 0.6939

Precision: 0.7491

Recall: 0.6386

F1 Score: 0.6895

Εικόνες 128x128:

Total params: 2,154,881 (8.22 MB)

Accuracy: 0.7425

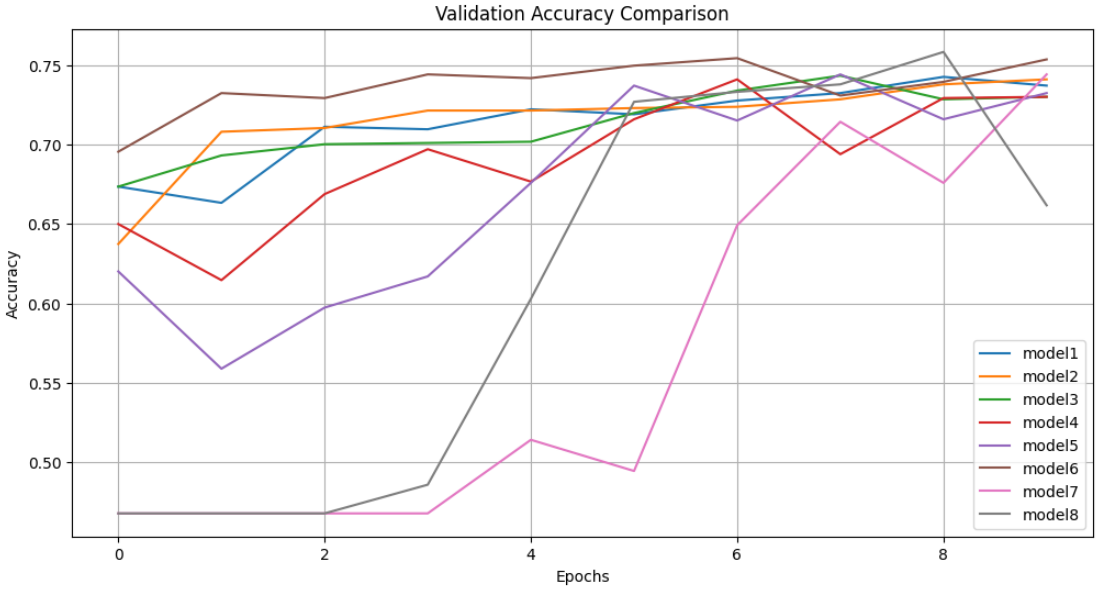
Precision: 0.7734

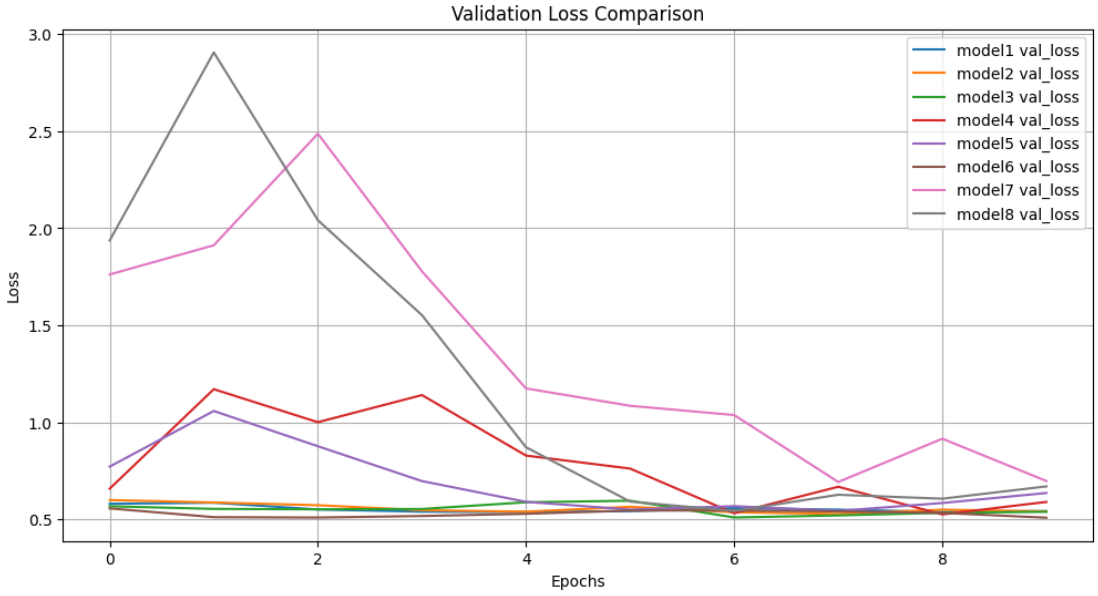
Recall: 0.7301

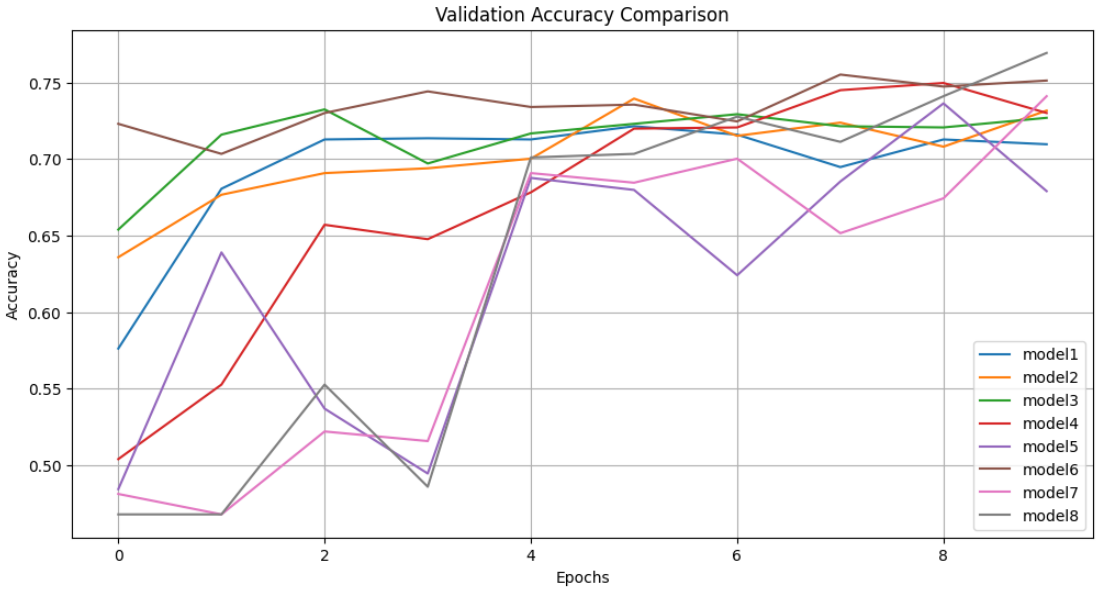
F1 Score: 0.7511

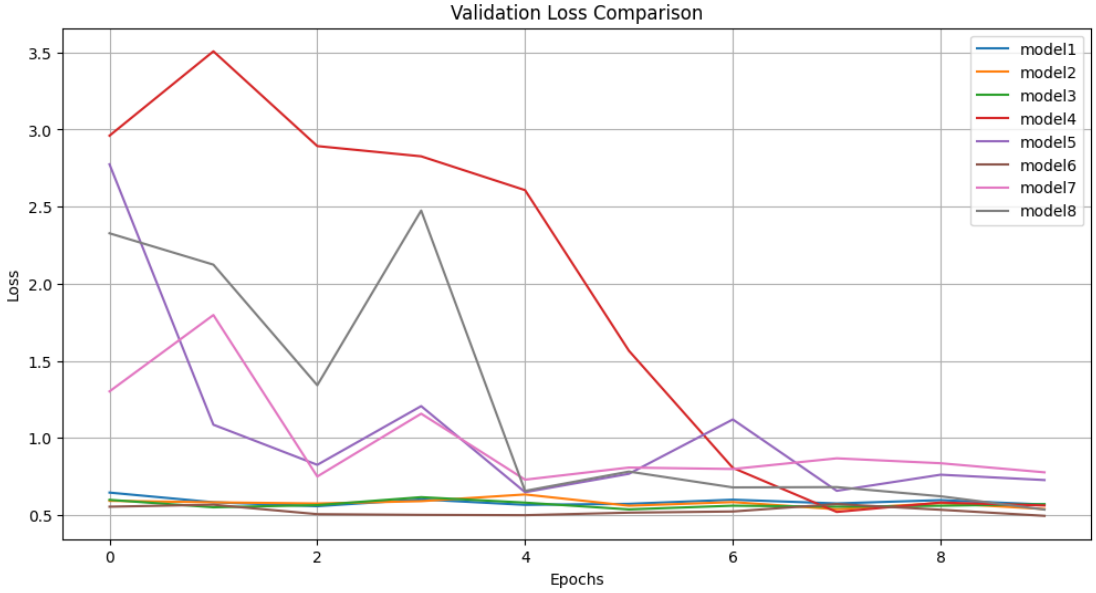
Μπορούμε να καταλήξουμε στα εξής συμπεράσματα:

1. **Για Μικρά Datasets:** Το μοντέλο 1 είναι κατάλληλο για γρήγορο πειραματισμό και προβλήματα με απλά patterns.
2. **Για Μέτρια Πολυπλοκότητα:** Τα μοντέλα 2, 3 και 8 αποτελούν εξαιρετικές επιλογές με ισορροπία μεταξύ απόδοσης και απλότητας.
3. **Για Πολύπλοκα Δεδομένα:** Τα μοντέλα 4, 5, 6 και 7 είναι πιο κατάλληλα, λόγω της αυξημένης τους ικανότητας για γενίκευση και εξαγωγή σύνθετων χαρακτηριστικών.
4. **Για Ισορροπία Απόδοσης και Πόρων:** Το μοντέλο 8 προσφέρει έναν πολύ καλό συμβιβασμό μεταξύ πολυπλοκότητας, regularization και χρόνου εκπαίδευσης.









Εικόνες 1,2,3,4. Διαγράμματα για val accuracy και val loss για όλα τα μοντέλα για 64x64 και 128x128 αντίστοιχα

Στο Kaggle για τα 3 μοντέλα με το καλύτερο f1 score για εικόνες 64x64, δηλαδή τα μοντέλα 3,5 και 6 είχαμε αντίστοιχα 0.90036 , 0.88435 , 0.91851

Ενώ για 128x128 εικόνες είχαμε στο Kaggle για τα μοντέλα 4, 6 και 8 σκορ της τάξεως 0.89733 , 0.92720 , 0.93436

Feature-Based Classification using a Pretrained CNN

Για το δεύτερο μέρος της άσκησης χρησιμοποιήσαμε ένα pretrained cnn το ResNet50 το οποίο έχει **Top-5 Accuracy 92.1%** και **Top-1 Accuracy 74.9% Parameters 25.6M**

Πιο συγκείμενα χρησιμοποιεί βάρη που έχουν εκπαιδευτεί πάνω στο μεγάλο και γενικής χρήσης σύνολο δεδομένων **ImageNet**. Αυτό επιτρέπει στο μοντέλο να έχει ήδη μάθει βασικά χαρακτηριστικά εικόνας, όπως άκρες, σχήματα και μοτίβα. Αποκλείει τα τελικά πλήρως συνδεδεμένα επίπεδα ταξινόμησης του ResNet50. Αυτό γίνεται επειδή θέλουμε να χρησιμοποιήσουμε το μοντέλο ως **feature extractor** και να προσθέσουμε δικά μας επίπεδα εξόδου, ανάλογα με το πρόβλημα που αντιμετωπίζουμε. Καθορίζει ότι οι εικόνες εισόδου θα έχουν μέγεθος 224x224 pixels και 3 κανάλια (RGB), που είναι το τυπικό σχήμα εισόδου του ResNet50.

Έπειτα κάνουμε extract features για όλα τα set μας μέσω του pretrained cnn, train set, validation set και test set.

Αρχικά, γίνεται **κανονικοποίηση των χαρακτηριστικών** με τη βοήθεια της StandardScaler από τη βιβλιοθήκη sklearn.preprocessing. Ο σκοπός είναι να μετασχηματιστούν τα χαρακτηριστικά ώστε να έχουν μέσο όρο 0 και τυπική απόκλιση 1. Αυτό είναι σημαντικό βήμα πριν την εφαρμογή αλγορίθμων μηχανικής μάθησης, καθώς εξασφαλίζει ότι όλα τα χαρακτηριστικά έχουν ίδια κλίμακα και δεν κυριαρχούν χαρακτηριστικά με μεγάλες τιμές.

Οι διαστάσεις των συνόλων δεδομένων μετά την κανονικοποίηση παραμένουν ίδιες:

Train: (5102, 100352)

Val: (1274, 100352)

Test: (500, 100352)

Λόγω του πολύ μεγάλου αριθμού χαρακτηριστικών, γίνεται χρήση του αλγορίθμου **SelectKBest** σε συνδυασμό με τη συνάρτηση f_classif για να επιλεχθούν τα **100 σημαντικότερα χαρακτηριστικά**. Ο αλγόριθμος αυτός υπολογίζει την **στατιστική τιμή F (ANOVA)** για κάθε χαρακτηριστικό και διατηρεί τα πιο διακριτά σε σχέση με τις κατηγορίες εξόδου. Με αυτόν τον τρόπο μειώνεται η διαστατικότητα του προβλήματος, κάτι που βοηθά στη **βελτίωση της απόδοσης** του μοντέλου και στη **μείωση του χρόνου εκπαίδευσης**.

KNN

Αρχικά, χρησιμοποιούμε το μοντέλο **K-Nearest Neighbors (KNN)** για ταξινόμηση των δεδομένων, επιλέγοντας τις καλύτερες υπερπαραμέτρους μέσω δοκιμών με διαφορετικές τιμές του **κ (αριθμός γειτόνων)** και **μετρικών απόστασης**. Η διαδικασία αυτή βοηθά να βρούμε τον βέλτιστο συνδυασμό παραμέτρων για το συγκεκριμένο πρόβλημα.

Ευκλείδεια απόσταση:

Η ακρίβεια κυμαίνεται μεταξύ 0,52 (K=1) και 0,54 (K=9).

Οι βαθμολογίες ακρίβειας, ανάκλησης και F1 παρουσιάζουν επίσης μικρές διακυμάνσεις, αλλά παραμένουν σχετικά σταθερές στις διάφορες τιμές του K.

Απόσταση Μανχάταν:

Η ακρίβεια κυμαίνεται από 0,53 (K=3) έως 0,54 (K=5).

Παρόμοια με την ευκλείδεια, η ακρίβεια, η ανάκληση και η βαθμολογία F1 είναι αρκετά σταθερές αλλά ελαφρώς υψηλότερες από ό,τι με την ευκλείδεια.

Απόσταση Chebyshev:

Η ακρίβεια είναι η χαμηλότερη και κυμαίνεται μεταξύ 0,47 (K=7) και 0,49 (K=3).

Αυτή η μετρική φαίνεται λιγότερο αποτελεσματική σε σύγκριση με τις άλλες, με χαμηλότερες επιδόσεις σε όλες τις μετρικές.

Cosine:

Η ακρίβεια κυμαίνεται από 0,52 (K=9) έως 0,54 (K=5).

Παρόμοια απόδοση με την απόσταση Manhattan, με ελαφρύ προβάδισμα έναντι της Chebyshev.

Παρατηρήσεις:

Οι μετρικές Manhattan και Cosine τείνουν να αποδίδουν ελαφρώς καλύτερα από τις μετρικές Euclidean και Chebyshev. Η Chebyshev έχει σταθερά χειρότερη απόδοση σε όλες τις τιμές των K, γεγονός που υποδηλώνει ότι αυτή η μετρική απόστασης μπορεί να μην είναι η καταλληλότερη για το σύνολο δεδομένων σας. Η απόδοση δεν είναι ιδιαίτερα μεταβλητή, καθώς το K αυξάνεται, με μικρές βελτιώσεις που σημειώνονται για μεγαλύτερες τιμές του K, αλλά δεν υπάρχουν σημαντικά άλματα στην ακρίβεια ή σε άλλες μετρικές.

Στο Kaggle το submission μας το οποίο προερχόταν από την καλύτερη μετρική f1 είχε σκορ 0.53281

MLP

Εδώ εξετάζουμε διαφορά mlp μοντέλα. Πιο συγκεκριμένα

Μοντέλο 1 (1 κρυφό επίπεδο, 512 μονάδες, ενεργοποίηση 'relu', 'Adam')
Αυτό το μοντέλο είναι απλό με μόνο ένα κρυφό επίπεδο. Αν και η επιλογή του **relu** είναι συχνά καλή για μη γραμμικές σχέσεις, με ένα μόνο κρυφό επίπεδο το μοντέλο ενδέχεται να μην έχει αρκετή ικανότητα να μάθει πολύπλοκες σχέσεις στα δεδομένα. Η ακρίβεια είναι περίπου 50%, υποδεικνύοντας ότι το μοντέλο δεν μάθει πολύ καλά τα δεδομένα.

Μοντέλο 2 (2 κρυφά επίπεδα, 512 και 256 μονάδες, ενεργοποίηση 'relu', 'SGD')
Έχει δύο κρυφά επίπεδα με διαφορετικό αριθμό μονάδων. Η χρήση του **SGD** (Stochastic Gradient Descent) μπορεί να οδηγήσει σε αργή σύγκλιση ή σε προβλήματα με την εκμάθηση αν δεν επιλέγεται σωστά το learning rate. Αυτό το μοντέλο παρουσιάζει χειρότερα αποτελέσματα στην ακρίβεια και στην F1 score σε σχέση με το Μοντέλο 1.

Μοντέλο 3 (3 κρυφά επίπεδα, 256, 128, 64 μονάδες, ενεργοποίηση 'sigmoid', 'Adam')
Αυτό το μοντέλο έχει περισσότερα κρυφά επίπεδα και διαφορετικό τύπο ενεργοποίησης (**sigmoid**), η οποία μπορεί να μην είναι τόσο αποτελεσματική για το hidden layer, καθώς συνήθως χρησιμοποιείται για την έξοδο (συνήθως για binary classification). Η συγκεκριμένη διαμόρφωση φαίνεται να μην έχει πολύ καλή απόδοση.

Μοντέλο 4 (4 κρυφά επίπεδα, 512, 256, 128, 64 μονάδες, ενεργοποίηση 'tanh', 'SGD')
Αυτό το μοντέλο έχει τον πιο πολύπλοκο αρχιτεκτονικό σχεδιασμό με 4 κρυφά επίπεδα. Η χρήση του **tanh** ως ενεργοποίηση μπορεί να έχει καλύτερη απόδοση για το hidden layer σε σχέση με το **sigmoid**. Ωστόσο, το **SGD** δεν φαίνεται να δίνει τα καλύτερα αποτελέσματα, παρόλο που το μοντέλο είναι πιο βαθύ.

Μοντέλο 5 (1 κρυφό επίπεδο, 256 μονάδες, ενεργοποίηση 'relu', 'Adam')
Αυτό το μοντέλο έχει μόνο ένα κρυφό επίπεδο με **relu** και είναι το πιο απλό σε σχέση με τα υπόλοιπα, αλλά με καλή απόδοση. Η ακρίβεια του είναι σχετικά χαμηλή, αλλά το **F1 score** είναι το καλύτερο από όλα τα μοντέλα (0.5705), πράγμα που υποδηλώνει ότι υπάρχει μια καλύτερη ισορροπία μεταξύ **precision** και **recall**, δηλαδή το μοντέλο κάνει λιγότερα λάθη με ψευδώς θετικές και παραλείψεις.

Μοντέλο 6 (2 κρυφά επίπεδα, 256 και 128 μονάδες, ενεργοποίηση 'tanh', 'Adam')
Αυτό το μοντέλο έχει δύο κρυφά επίπεδα με διαφορετική ενεργοποίηση (**tanh**). Η απόδοση του είναι πολύ χαμηλή (μόλις 36% ακρίβεια), με τη μέθοδο **Adam** να μην φαίνεται να βοηθά πολύ. Αυτό μπορεί να σημαίνει ότι η συνδυασμένη χρήση δύο κρυφών επιπέδων με το **tanh** δεν είναι αποδοτική για το συγκεκριμένο πρόβλημα.

Μοντέλο 7 (4 κρυφά επίπεδα, 128, 128, 64, 64 μονάδες, ενεργοποίηση 'sigmoid', 'SGD')
Το συγκεκριμένο μοντέλο έχει αρκετά περισσότερα επίπεδα, αλλά η χρήση του **sigmoid** ως ενεργοποίηση στα κρυφά επίπεδα και ο **SGD** φαίνεται να μην το καθιστούν αποτελεσματικό. Αν και το **recall** για την κλάση 1 είναι σχετικά καλό, η ακρίβεια και η συνολική απόδοση δεν είναι ικανοποιητικές.

Μοντέλο 8 (1 κρυφό επίπεδο, 64 μονάδες, ενεργοποίηση 'tanh', 'SGD')
Ένα πιο απλό μοντέλο, με μόνο 1 κρυφό επίπεδο, έχει κακή απόδοση τόσο στην ακρίβεια όσο και στο **F1 score**. Η χρήση του **SGD** χωρίς βελτιστοποίηση του learning rate φαίνεται να περιορίζει την απόδοση του μοντέλου.

Τα μοντέλα με **relu** ενεργοποίηση και **Adam** φαίνεται να έχουν καλύτερη απόδοση, με το **Μοντέλο 5** να είναι το καλύτερο από όλα τα μοντέλα σύμφωνα με το **F1 score**. Τα πιο βαθιά μοντέλα (με περισσότερα κρυφά επίπεδα) δεν δείχνουν να αποδίδουν πάντα καλύτερα, και ο **SGD** δεν φαίνεται να είναι τόσο αποτελεσματικός όσο ο **Adam** για το συγκεκριμένο πρόβλημα.

Στο Kaggle το submission μας το οποίο προερχόταν από το μοντέλο 5 με το καλύτερο f1 είχε σκορ 0.43445

SVC

Εδώ δοκιμάζουμε διαφορετικούς συνδυασμούς **πυρήνα (kernel)** και **παραμέτρου C** για ένα SVM μοντέλο, και κρατάμε τον καλύτερο συνδυασμό σύμφωνα με το **F1 score**

Τα αποτελέσματα είναι τα εξής για linear kernel:

C	Accuracy	Recall (`pleasant`)	Recall (`unpleasant`)	F1 score
0.1	0.4482	0.6667	0.20	0.5625
1	0.4474	0.6637	0.20	0.5611
10	0.4474	0.6637	0.20	0.5611
100	0.4474	0.6637	0.20	0.5611

Τα αποτελέσματα είναι τα εξής για rbf kernel:

C	Accuracy	Recall (pleasant)	Recall (unpleasant)	F1 Score
0.1	0.5330	0.9853	0.02	0.6919
1	0.4623	0.7802	0.10	0.6070
10	0.4631	0.6519	0.25	0.5638
100	0.4992	0.5870	0.40	0.5551

Τι σημαίνει η παράμετρος C πρακτικά;

- **C μικρό** = χαμηλή ποινή σε λάθος ταξινομήσεις ⇒ **μεγάλα margins**, soft margin SVM.
- **C μεγάλο** = μεγάλη ποινή σε λάθος ταξινομήσεις ⇒ **στενά margins**, προσπαθεί πολύ να μην κάνει λάθος ⇒ μπορεί να overfit.

Το RBF (Gaussian) kernel επιτρέπει **μη γραμμικούς διαχωρισμούς**. Αυτό σημαίνει ότι το decision boundary **δεν είναι ευθεία γραμμή**, αλλά καμπύλη. Είναι πιο ικανό να χειριστεί περίπλοκες σχέσεις μεταξύ των χαρακτηριστικών. Γι' αυτό είδαμε **μεγαλύτερη προσαρμοστικότητα** σε σύγκριση με τον γραμμικό SVM.

Στο Kaggle το submission μας το οποίο προερχόταν από το μοντέλο με C: 0.1 και rbf kernel που είχε σκορ 0.70967

Logistic Regression

Για το logistic regression δοκιμάσαμε διάφορες τιμές του C.Το C είναι η **αντίστροφη της regularization strength** (C = 1/λ).

Μικρό C → **ισχυρή regularization** → το μοντέλο γίνεται πιο απλό, αποφεύγει overfitting.

Μεγάλο C → **αδύναμη regularization** → προσπαθεί να ταξινομήσει τέλεια το train set, κίνδυνος overfit.

Αν C πολύ μικρό ⇒ underfitting

Αν C πολύ μεγάλο ⇒ overfitting

C = 0.01

Recall για pleasant: 68.44% → πολύ υψηλό → το μοντέλο πιάνει σχεδόν όλα τα θετικά.

Recall για unpleasant: 18% → σχεδόν αγνοεί τα αρνητικά.

F1 Score (binary): 0.5693 → το υψηλότερο, επειδή ευνοείται από την υψηλή recall των θετικών.

Άρα, το μοντέλο τείνει να **προβλέπει περισσότερο "pleasant"**, ίσως γιατί είναι και η πλειοψηφική κλάση.

C = 0.1

Μικρή πτώση στο recall και F1 → αρχίζει να γίνεται ελαφρώς πιο αυστηρό.

F1 πέφτει σε **0.5654**, κοντά στο προηγούμενο.

C = 1

Ίδια εικόνα: η pleasant κυριαρχεί, unpleasant έχει χαμηλό recall.

F1 πέφτει κι άλλο: **0.5616**

C = 10 και C = 100

Recall για pleasant ≈ 66%, για unpleasant ≈ 21%

F1: 0.5632 και 0.5629

Στο Kaggle το submission μας το οποίο προερχόταν από το μοντέλο με C: 0.01 που είχε σκορ 0.46464

Ensemble Learning methodologies

Random Forest

n_estimators=10:

- **Accuracy**: 47.2%
- **F1 Score**: 0.5608
- Ελαφρώς πιο ισορροπημένη απόδοση μεταξύ των δύο κλάσεων (π.χ., unpleasant recall = 29%).
- Ίσως υπερβολικά λίγα δέντρα, που οδηγούν σε ασταθή συμπεριφορά.

n_estimators=50:

- **Accuracy**: 44.2%
- **F1 Score**: **0.5740** (το καλύτερο μεταξύ των ensemble μοντέλων)
- Υψηλό recall για την κλάση *pleasant* (71%), αλλά πολύ χαμηλό recall για την *unpleasant* (14%).
- Εξισορροπεί τα F1-scores δίνοντας πλεονέκτημα στη "dominant" κλάση.

n_estimators=100:

- **Accuracy**: 43.2%
- **F1 Score**: 0.5670
- Παρόμοια με τα 50 δέντρα, αλλά χωρίς κάποια ουσιαστική βελτίωση.
- Το unpleasant class recall μειώνεται (13%), δείγμα class imbalance ή overfitting στους ευκολότερους θετικούς δείκτες.

AdaBoost

n_estimators=100:

- **Accuracy**: 43.2%
- **F1 Score**: 0.5205
- Χαμηλότερη επίδοση συνολικά.
- Λιγότερο ισχυρό από τα αντίστοιχα Random Forest μοντέλα.

n_estimators=200:

- **Accuracy**: 45.1%
- **F1 Score**: 0.5666

- Σημαντικά καλύτερο recall για την pleasant (67%).
- Λίγο καλύτερη ισορροπία απόδοσης σε σχέση με τα 100 estimators.

Σκορ στο Kaggle με RandomForest (n = 50) και F1 score 0.5740 , 0.53548

CNN

Δοκιμάσαμε ένα τυχαίο cnn μοντέλο με 2 convolution επίπεδα, batchnormalization και max pooling καθώς και dense επίπεδα μετά το flatten και πήραμε αποτελέσματα στο Kaggle 0.93333

Σκέψεις

Πολλά αποτελέσματα δεν ήταν αυτά τα οποία θα επιθυμούσαμε και αυτό μπορεί να οφείλεται σε πολλούς παράγοντες όπως:

Ανισορροπία Δεδομένων

- Η κλάση pleasant φαίνεται να έχει περισσότερα παραδείγματα ή πιο εύκολα διακριτά features.
- Τα μοντέλα ίσως να μαθαίνουν να "παίζουν safe" και **προβλέπουν πολύ συχνά pleasant**, παραμελώντας την unpleasant.

Χαμηλό Recall για unpleasant

- Ίσως τα features για την unpleasant δεν είναι καλά επιλεγμένα ή το μοντέλο χρειάζεται πιο ισχυρό ταξινομητή.
- Υπάρχει κίνδυνος **false negatives** (π.χ. να προβλέπει pleasant ενώ είναι unpleasant).

Σίγουρα θα θέλαμε να δοκιμάσουμε ακόμα περισσότερα features και να μην περιοριστούμε στα 100 αλλά κάτι πέρα από αυτό θα ήταν πολύ κοστοβόρο τόσο χρονικά όσο και σε πόρους. Επίσης θα θέλαμε να εξερευνήσουμε τρόπους να χωρίσουμε ακόμα καλυτέρα τα training και validation set μας