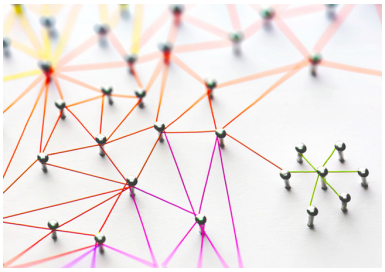
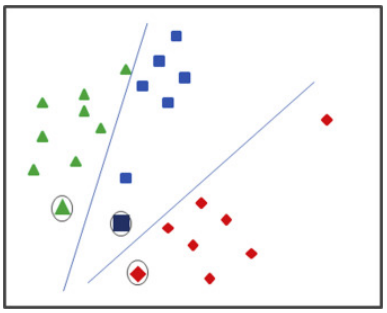
Σύγκριση κλασσικών μεθόδων ταξινόμησης
K-Nearest Neighbors - Nearest Class Centroid



(a) KNN



(b) NCC

Αριστοστέλειο Πανεπιστήμιο Θεσσαλονίκης

10ο Εξάμηνο ECE AUTh

7ο Εξάμηνο CSD AUTh

Νευρωνικά Δίκτυα - Βαθιά Μάθηση

Χάρης Φίλης ΑΕΜ: 9449

GitHub Repository Link :

October,2022

Contents

0.1 Εισαγωγή - Επιλογή δεδομένων 2
0.2 Λίγα Λόγια για τους αλγορίθμους 2
0.2.1 Κ πλησιέστεροι γείτονες - K nearest Neighbours 2
0.2.2 Πλησιέστερο κέντρο κλάσης - Nearest Class Centroid . . . 3
0.3 Ενδεικτικά Αποτελέσματα 3

List of Figures

1 Graphical Representation of KNN algorithm 3
2 Classification Report Knn 4
3 "Best" Knn CM 5
4 Classification Report NCC 5
5 CM-Heatmap 5

0.1 Εισαγωγή - Επιλογή δεδομένων

Η παρούσα εισαγωγική εργασία αποτελεί προαπαιτούμενο για την υλοποίηση ενός Multi-Layer Perceptron Classifier στο πλαίσιο του μαθήματος Νευρωνικά Δίκτυα - Βαθια Μάθηση του τμήματος του Πληροφορικού ΑΠΘ (Διατμηματική Εκπαίδευση).

Εξαρχής γνωρίζω πως με ενδιαφέρει η εικόνα σαν μέσο πληροφορίας και εφόσον μας ενδιαφέρει το task ταξινόμησης σε κλάσεις αποφάσισα να διαλέξω Datasets με αναγνώριση είτε αντικειμένου στην εικόνα όπως το CIFAR10 είτε γενικότερου κτιρίου ή χώρου προβολής στην εικόνα όπως το Intel Image Classification Dataset τα οποία επιλέχθηκαν. Το πρώτο συστήθηκε από τον διδάσκων του μαθήματος ενώ το δευτέρο βρέθηκε μετά από λίγο ψάξιμο στο kaggle-url.

Ο λόγος που έγινε επιλογή και δεύτερου dataset είναι καθώς από την θεωρία του μαθήματος γνωρίζουμε ότι τα νευρωνικά δίκτυα γίνονται πιο robust όταν εκπαιδεύονται σε διαφορετικά δεδομένα μιας και η λειτουργικότητα ενός νευρωνικού δικτύου μπορεί να μεταφερθεί από το ένα dataset στο άλλο χωρίς μεγάλη μετατροπή της αρχιτεκτονικής του.

Στο παρόν κομμάτι της εργασίας αναλύονται δύο κλασσικοί αλγόριθμοι κατηγοριοποίησης, ο KNN(K-nearest-neighbors) και ο NCC(Nearest Class Centroid), οι οποίοι χρησιμοποιούν κλασσικές μεθόδους βελτιστοποίησης. Ωστόσο είναι εκ φύσεως άπληστοι αλγόριθμοι κάτι που θα φανεί στα παρακάτω αποτελέσματα μιας και χάνουν τα καλύτερα και πιο σύντομα μονοπάτια ή και τις βέλτιστες λύσεις ακόμα.

Το κομμάτι της εργασίας είναι εισαγωγικό οπότε προφανώς έχει μη ολοκληρωμένα κομμάτια. Ωστόσο στο τελικό κομμάτι της πρώτης εργασίας θα υπάρχουν αποτελέσματα και για τα δύο dataset μιας και παρουσιάζω τρόπους πως γίνονται load και πως τα χειριζόμαστε. Τέλος η from scratch υλοποίηση του knn αλγορίθμου δεν μπορούσε να χειριστεί σωστά τις tensor δομές που εισάγει η pytorch ώστε να βρεθεί η ευκλείδεια απόσταση μεταξύ εικόνων(έγινε προσπάθεια για χρήση και του cdist αλλά λογικά κάποιο data preprocessing χαλούσε το δεδομένα) οπότε πιθανά αποτελέσματα λογικά αν υπάρχουν θα βρίσκονται στο notebook - MetricsTestKnnNcc.ipynb

0.2 Λίγα Λόγια για τους αλγορίθμους

0.2.1 Κ πλησιέστεροι γείτονες - K nearest Neighbours

Ο αλγόριθμος των **k** πλησιέστερων γειτόνων (**k-NN**) είναι ένας μη παραμετρικός αλγόριθμος αλγόριθμος επιβλεπόμενης μάθησης ο οποίος μπορεί να λύσει και το πρόβλημα της ταξινόμησης αλλά και της παλινδρόμησης.

Και στις δύο περιπτώσεις στην είσοδο έχουμε τα δεδομένα εκπαίδευσης τα οποία με ειδικό τρόπο μετασχηματίζονται ως πλησιέστερα σε ορισμένο πλήθος γειτόνων τους (εκεί έγκειται και το **k**)με βάση μια μετρική συνήθως την ευκλείδεια τους απόσταση στο χώρια διαστάσεων τους. Στο κομμάτι την ταξινόμησης που μας ενδιαφέρει η έξοδος αποτελεί την συμμετοχή ή όχι στην ομάδα με βάση την ετικέτα που έχουν τα πλησιέστερα δεδομένα. Στο συγκεκριμένο αλγόριθμο στον οποίο έχουμε τα labels εφόσον είναι επιβλεπόμενος δεν μας ενδιαφέρει το κομμάτι της συσσταδοποίησης το οποίο υλοποιεί ο k-means αλγόριθμος. Αντιθέτως ένα δεδομένο ταξινομείται σε μια κλάση από το πλήθος των ψήφων των κλάσεων που έχουν οι γείτονές του. Ουσιαστικά το δεδομένο ανατίθεται στην πιο κοινή κλάση που έχουν οι γεί-

τονές του.

Με βάση αυτή την αρχή γίνεται μια προσπάθεια υλοποίησης ενός αλγορίθμου σε python from scratch στο αρχείο Knn-Generic. Ωστόσο το τρέξιμό του ήταν προβληματικό τουλάχιστον λόγω της χρήσης tensor μιας και το dataset το διαχειρίζεται η pytorch.

Όπως και να έχει στο τελικο notebook γίνεται χρήση των αλγορίθμων που παρέχει το *sklearn.clustering* πακέτο.

Παρακάτω απεικονίζεται γραφικά πως δουλεύει ο knn σε δισδιάστατη μορφή δεδομένων. Είναι φανερό ότι κάθε νέο δεδομένο το οποίο στον κώδικα δίνεται και ως querry_sample συγκρίνεται με ενα πλήθος γειτόνων του, οι οποίοι ψηφίζουν με την σειρά τους μέσω της problem_function *mode* που πρακτικά μετρά πόσοι είναι υπέρ μιας κλάσης και πόσοι είναι υπέρ άλλης κλάσης. Για αυτό γίνεται και εισαγωγή της δομής Counter.

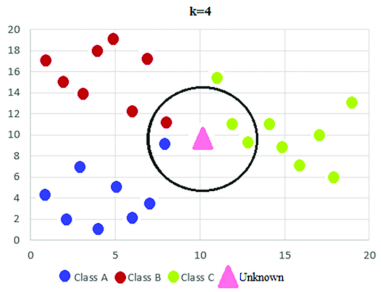


Figure 1: Graphical Representation of KNN algorithm

0.2.2 Πλησιέστερο κέντρο κλάσης - **Nearest Class Centroid**

Ο δεύτερος ταξινομητής που εξετάζεται είναι ένας πιο απλός αγλόριθμος ταξινόμησης από τον παραπάνω ο Nearest class centroid**(Ncc)**.

Με βάση την βιβλιογραφία(μιας και δεν έγινε από το μηδέν υλοποίηση αυτού) περιλαμβάνει πρώτα την περίληψη του dataset σε βασικά κέντρα τα centroids(εδώ μπορεί να ακολοθείται ένας αλγόριθμος συσταδοποίησης όπως ο k-means).Στην συνέχεια τα κέντρα αυτά χρησιμοποιούνται για να γίνει η εκτίμηση σε ποία κλάση ανήκει κάθε νέο sample του dataset. Για κάθε κλάση το κέντρο των δεδομένων βρίσκεται παίρνοντας τον μέσο όρο κάθε εκτιμητή (κάθε κλάσης) για το training set.

Ο αλγόριθμος τρέχει γενικότερα πιο γρήγορα απο τον προηγούμενο αλλά δεν σημαίνει οτι κάνει καλύτερη ταξινόμηση.

0.3 Ενδεικτικά Αποτελέσματα

Και οι δύο αλγόριθμοι είναι κατα κύρια βάση άπληστοι. Αυτο σημαίνει ότι δεν έχουν ορισμένες μεταβλητές που τους σταματούν αν κινούνται προς λάθος σημείο δηλαδή αν ορίζουν κάποιο δεδομένο σε λάθος κλάση του Dataset. Για αυτό παρατηρείται στα παρακάτω confussion matrices και heatmaps πώς πολλές φορές μπορεί ένα αεροπλάνο να ταξινομηθεί ως καράβι ή ένας τάρανδος ώς βάτραχος μιας και μπορεί

οι εικόνες να μην είχαν μεγάλη απόσταση (ως προς φωτεινότητα pixel) και να είχε γίνει λάθος ταξινόμηση.

Παράλληλα δεν βλέπουμε καλές τιμές στις μετρικές όπως στο percision που αφορά τον αριθμό των true-positive αποτελεσμάτων δια όλων των positive αποτελεσμάτων. Τα ίδια και στο recall που είναι ο αριθμός true-positive δια τον αριθμό όλων των δειγμάτων. Έτσι το f1-score ενώ θα έπρεπε να είναι κανονικά κοντά στο 1 τελικά ειναι στο 0.4. Που σημαινει οτι το dataset ταξινομείται κατα κύρια βάση λάθος. Τέλος τα confussion matrices θα έπρεπε να εχουν περισσότερες προβλέψεις στην κύρια διαγώνιο για σωστή ταξινόμηση ωστόσο κάτι τέτοιο δεν συμβαίνει. Οσον αφορά τον knn με 2 γείτονες και με τρεις δεν φαίνεται ουσιαστική διαφορά στα αποτελέσματα ενώ με έναν έχουμε τα χειρότερα αποτελέσματα. Συγκριτικά θα μπορουσαμε να πουμε οτι ο knn είναι πιο άπληστος καθώς για να γίνουν όλα αυτά τα test με όλο το πλήθος των γειτόνων πήρε 40 λεπτά.

Αποτελέσματα **Knn**

	precision	recall	f1-score	support
0	0.39	0.28	0.33	57
1	0.25	0.12	0.16	41
2	0.19	0.25	0.22	51
3	0.20	0.08	0.12	49
4	0.12	0.36	0.18	44
5	0.50	0.12	0.19	50
6	0.20	0.23	0.22	56
7	0.11	0.06	0.08	48
8	0.32	0.63	0.43	57
9	0.58	0.19	0.28	59
accuracy			0.24	512
macro avg	0.29	0.23	0.22	512
weighted avg	0.30	0.24	0.23	512

(a) K = 1

	precision	recall	f1-score	support
0	0.40	0.35	0.37	57
1	0.09	0.07	0.08	41
2	0.13	0.61	0.21	51
3	0.15	0.06	0.09	49
4	0.13	0.23	0.17	44
5	0.15	0.08	0.11	50
6	0.22	0.12	0.16	56
7	0.11	0.02	0.04	48
8	0.56	0.16	0.25	57
9	0.50	0.08	0.14	59
accuracy			0.18	512
macro avg	0.24	0.18	0.16	512
weighted avg	0.26	0.18	0.17	512

(b) K = 2

	precision	recall	f1-score	support
0	0.20	0.65	0.30	57
1	0.06	0.02	0.03	41
2	0.09	0.12	0.10	51
3	0.14	0.12	0.13	49
4	0.07	0.20	0.11	44
5	0.35	0.14	0.20	50
6	0.05	0.02	0.03	56
7	0.12	0.02	0.04	48
8	0.43	0.16	0.23	57
9	0.00	0.00	0.00	59
accuracy			0.15	512
macro avg	0.15	0.15	0.12	512
weighted avg	0.15	0.15	0.12	512

(c) K = 3

Figure 2: Classification Report Knn

Confusion Matrix

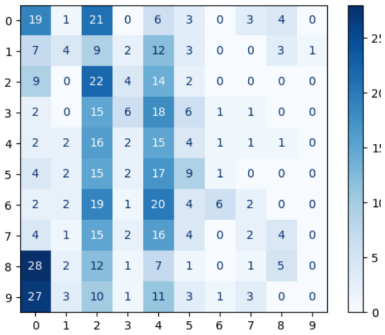


Figure 3: "Best" Knn CM

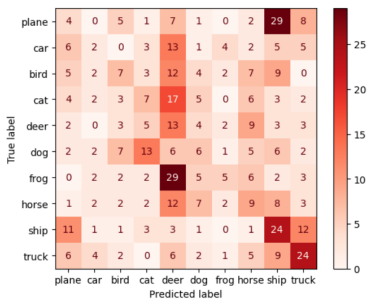
Αποτελέσματα **NCC**

Time elapsed 10.648508787155151.s

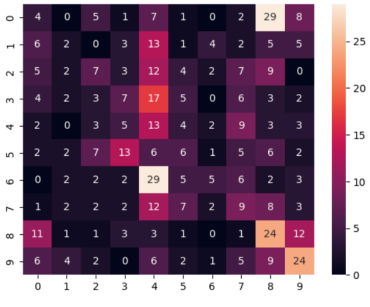
	precision	recall	f1-score	support
0	0.10	0.07	0.08	57
1	0.12	0.05	0.07	41
2	0.22	0.14	0.17	51
3	0.18	0.14	0.16	49
4	0.11	0.30	0.16	44
5	0.17	0.12	0.14	50
6	0.29	0.09	0.14	56
7	0.17	0.19	0.18	48
8	0.24	0.42	0.31	57
9	0.39	0.41	0.40	59
accuracy			0.20	512
macro avg	0.20	0.19	0.18	512
weighted avg	0.21	0.20	0.19	512

Figure 4: Classification Report NCC

Confusion Matrix-Heatmap



(a) CM



(b) Heatmap

Figure 5: CM-Heatmap

Bibliography

[Nearest Centroid] Nearest Shrunken Centroids With Python /nearest-shrunken-centroids

[Dataloader-Torch] Tutorial on pytorch on how to import datasets with iterators https://pytorch.org/tutorials/beginner/basics/data$_{t}utorial.html$

[sklearn KNN] sklearn-knn

[sklearn NCC] sklearn-knn