Data Mining & Machine Learning Project

Κοντόπουλος Θεόδωρος

1084523

May 26,2024

Ερώτημα 1

Αρχικά θα ενοποιήσουμε ολα τα αρχεία csv ,στό σύνολο δεδομένων δεν λείπει καποια τιμή αρα δέν θα χρειαστεί να συμπληρώσουμε κάποια στην συνέχεια θα χρησιμοποιήσουμε την .describe() για να βρούμε τα βασικά συγκεντρωτικά στατιστικά μεγέθη οπως ο μέσος όρος και η διασπορά :

	back_x	back_y	back_z	thigh_x	thigh_y	thigh_z
count	6.461328e+06	6.461328e+06	6.461328e+06	6.461328e+06	6.461328e+06	6.461328e+06
mean	-8.849574e-01	-1.326128e-02	-1.693779e-01	-5.948883e-01	2.087665e-02	3.749160e-01
std	3.775916e-01	2.311709e-01	3.647385e-01	6.263466e-01	3.884511e-01	7.360983e-01
min	-8.000000e+00	-4.307617e+00	-6.574463e+00	-8.000000e+00	-7.997314e+00	-8.000000e+00
25%	-1.002393e+00	-8.312914e-02	-3.720700e-01	-9.742110e-01	-1.000873e-01	-1.557138e-01
50%	-9.748998e-01	2.593677e-03	-1.374510e-01	-4.217309e-01	3.262909e-02	7.004390e-01
75%	-8.123032e-01	7.251000e-02	4.647321e-02	-1.678755e-01	1.549512e-01	9.486747e-01
max	2.291708e+00	6.491943e+00	4.909483e+00	7.999756e+00	7.999756e+00	8.406235e+00

Θα δημιούργησουμε boxplot για καθε στήλη του συνόλου δεδομένων ομαδοποιημένα ανα label για να πάρουμε κάποια πληροφορία για τα δεδομένα ανα label αλλα και για την κατανομή των δεδομένων και πιθανούς outliers .

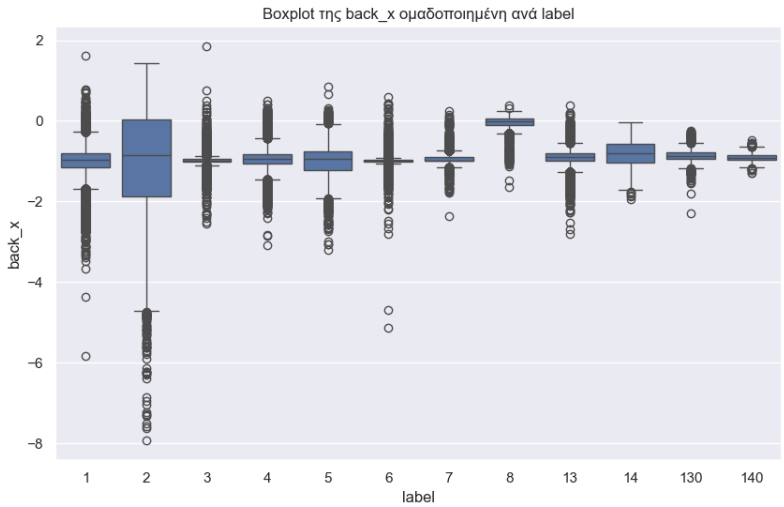


Figure 1: Boxplot της back x ομαδοποιμένο ανά label.

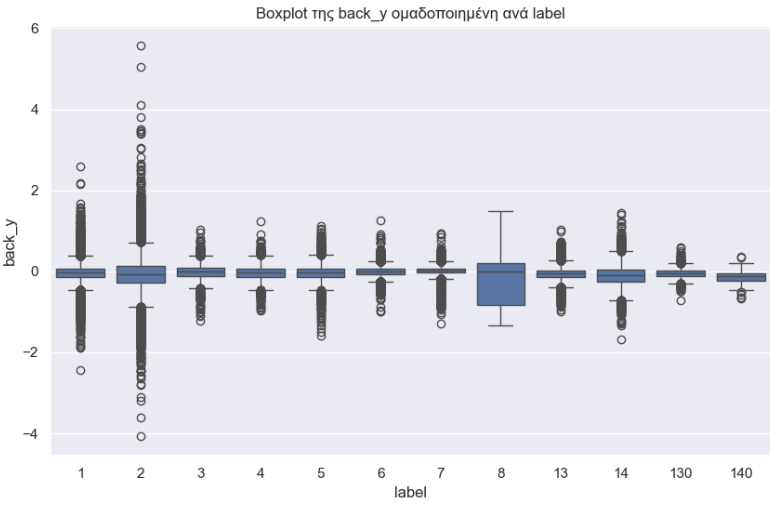


Figure 1 : Boxplot της back y ομαδοποιμένο ανά label.

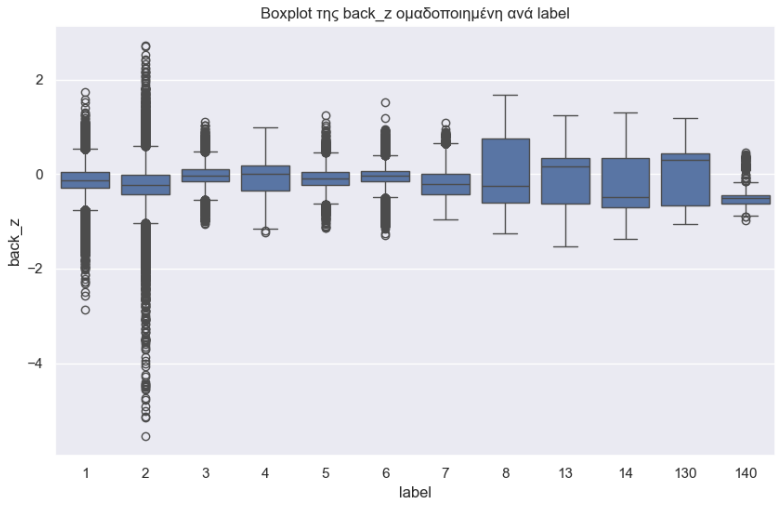


Figure 3: Boxplot της back z ομαδοποιμένο ανά label.

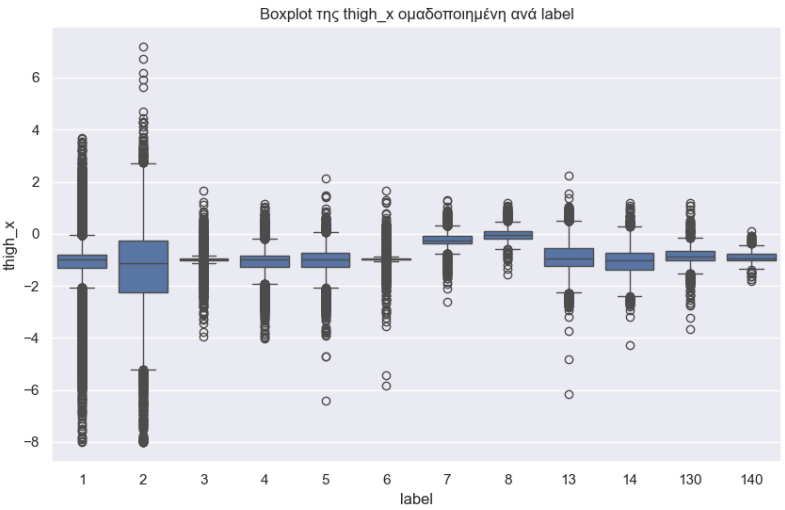


Figure 4: Boxplot της thigh x ομαδοποιμένο ανά label.

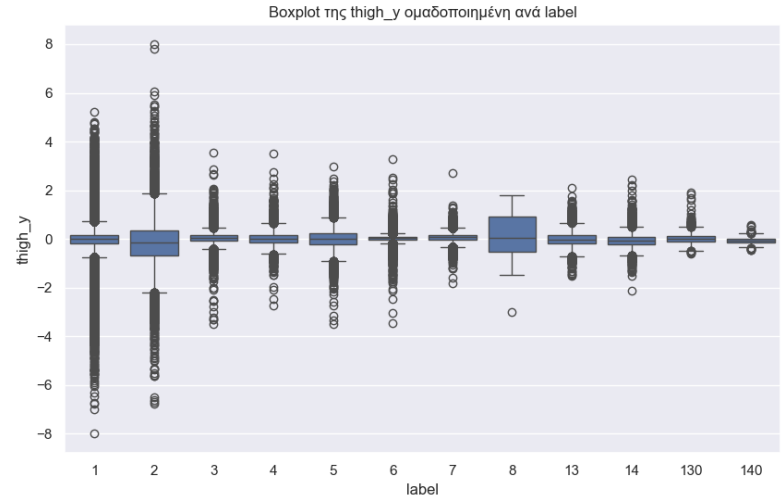


Figure 2: Boxplot της thigh y ομαδοποιημένη ανά label.

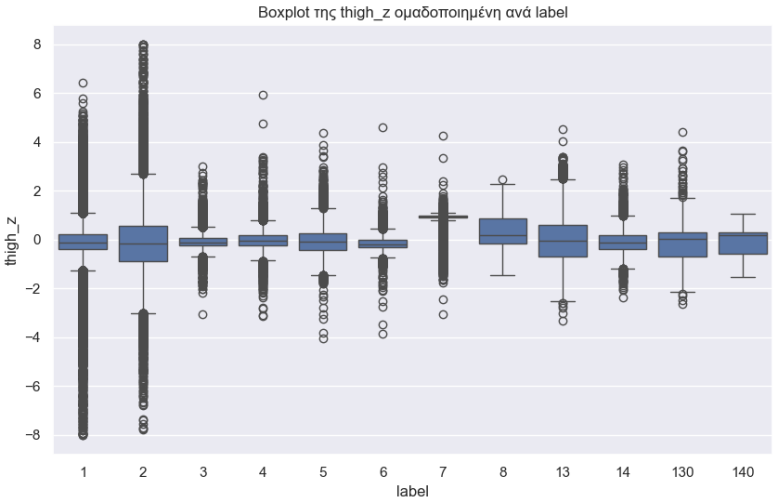


Figure 6: Boxplot της thigh z ομαδοποιημένη ανά label.

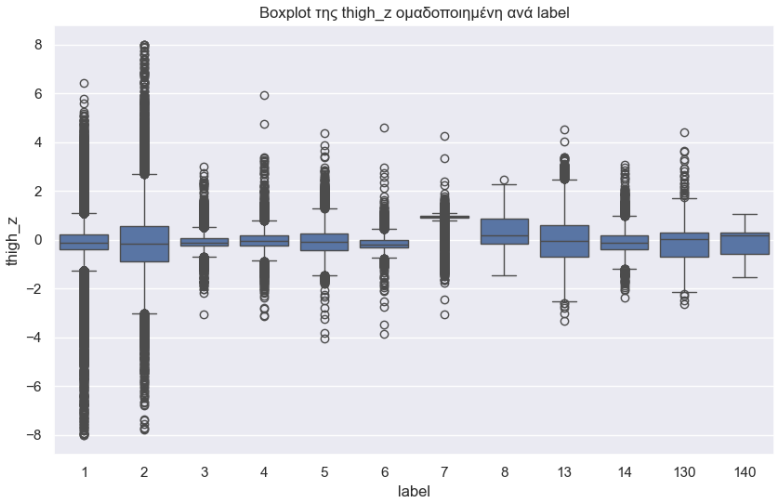


Figure 7: Boxplot της thigh z ομαδοποιημένη ανά label.

Στην συνέχεια , δημιουργήσαμε το ιστόγραμμα καθε στήλης του συνόλου δεδομένων για να κατανοήσουμε καλύτερα το σύνολο των δεδομένων αλλα και την κατανομή τους συνδυασμό με τα boxplot.

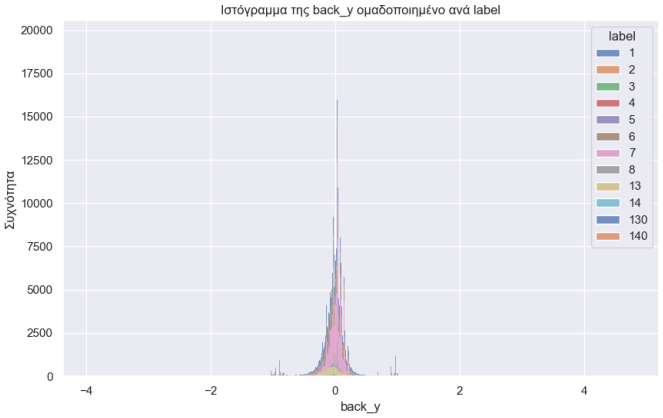


Figure 8: Ιστόγραμμα της back_y ανά label

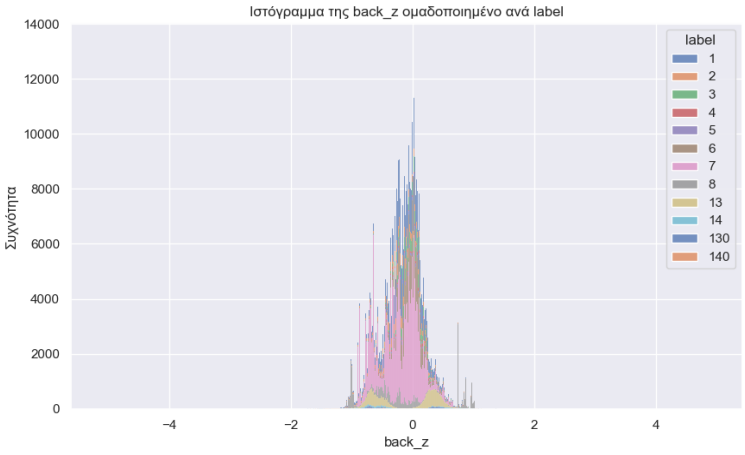


Figure 10 : Ιστόγραμμα της back_z ανά label

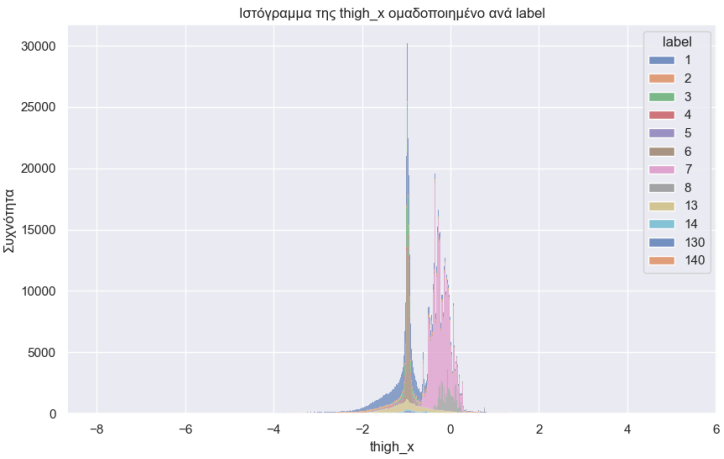


Figure 11: Ιστόγραμμα της thigh_x ανά label

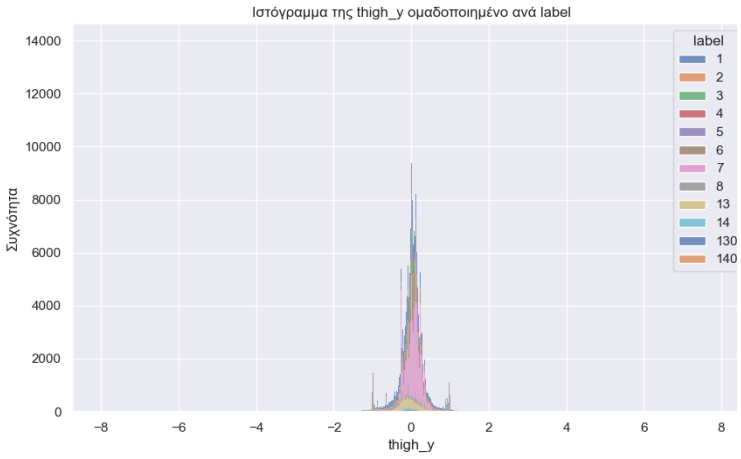


Figure 12: Ιστόγραμμα της thigh_y ανά label

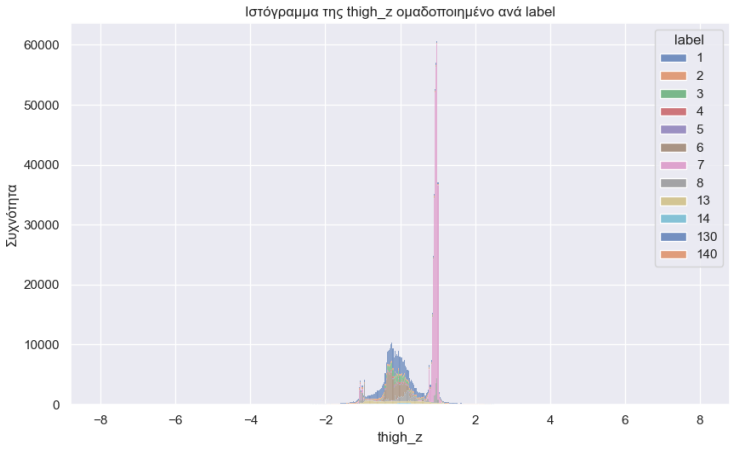
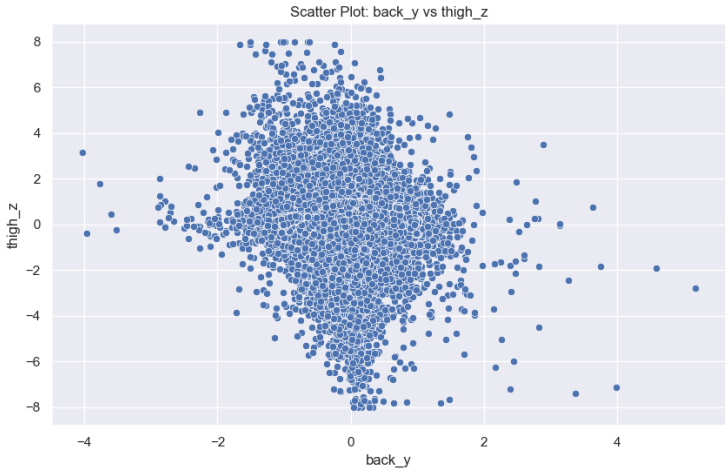
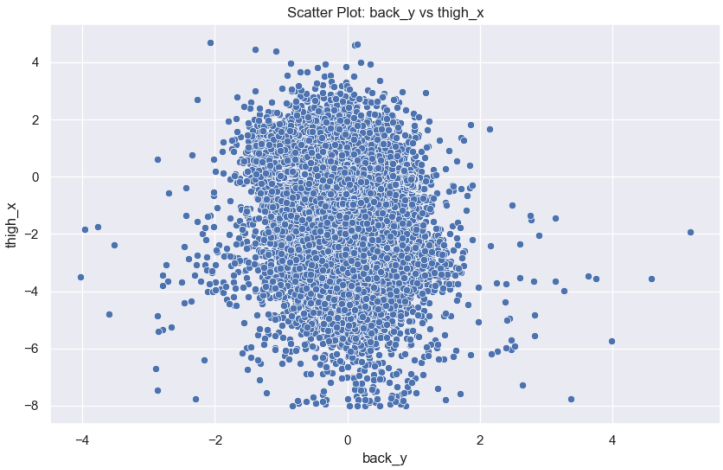


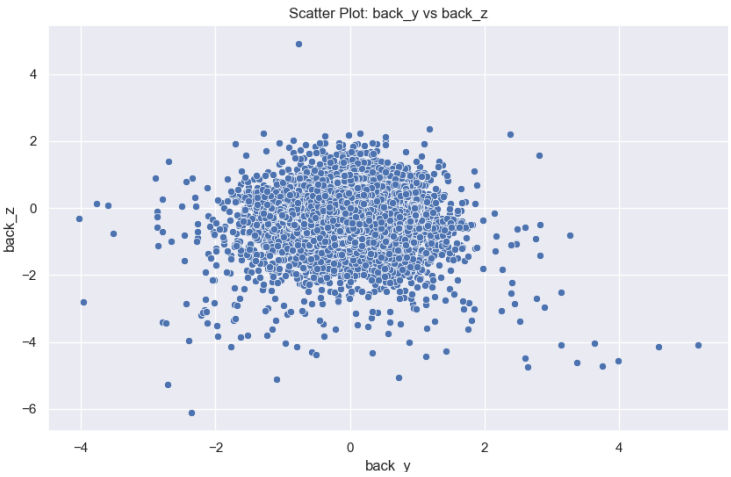
Figure 13:Ιστόγραμμα της thigh_z ανά label

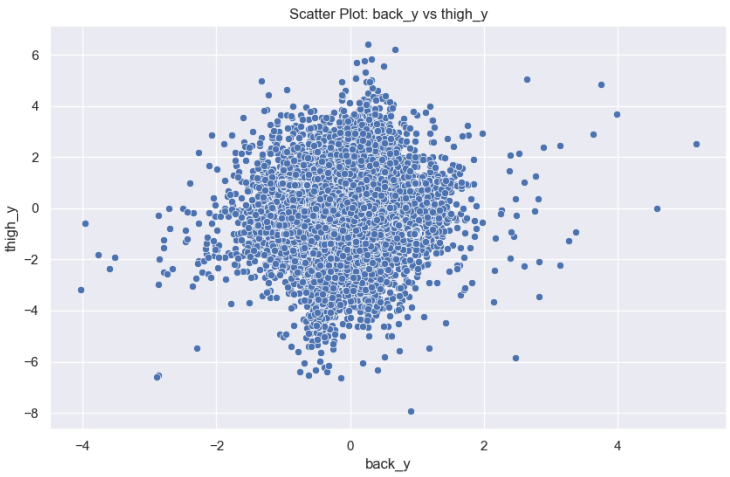
Από τα διαγράμματα μπορούμε να διακρίνουμε οτι τα δεδομένα ακολουθούν κατα κύριο λογο την κανονική κατανομή με mean ≈ 0 και std ≈ 1 και ότι υπάρχουν πιθανοί outliers οι οποίοι είναι πολυ πιθανόν να χρειαστεί να διαγραφούν .

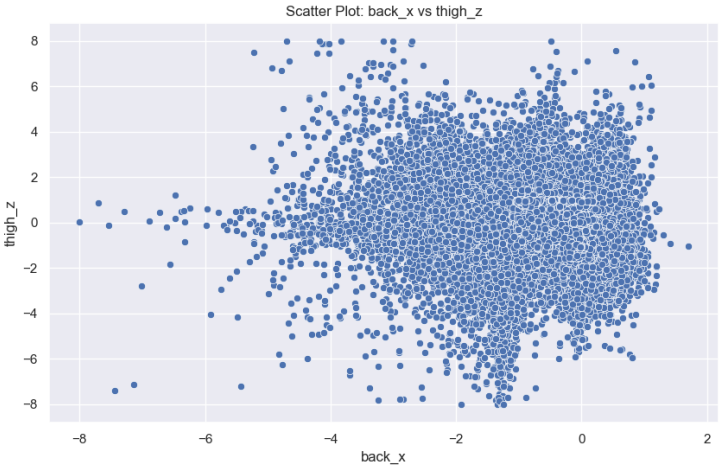
Σε αυτό το σημείο θα προσπαθήσουμε να διακρίνουμε τυχόν συσχετίσεις μεταξύ των μεταβλητών δημιουργώντας scatter plot ανά δύο μεταβλητές

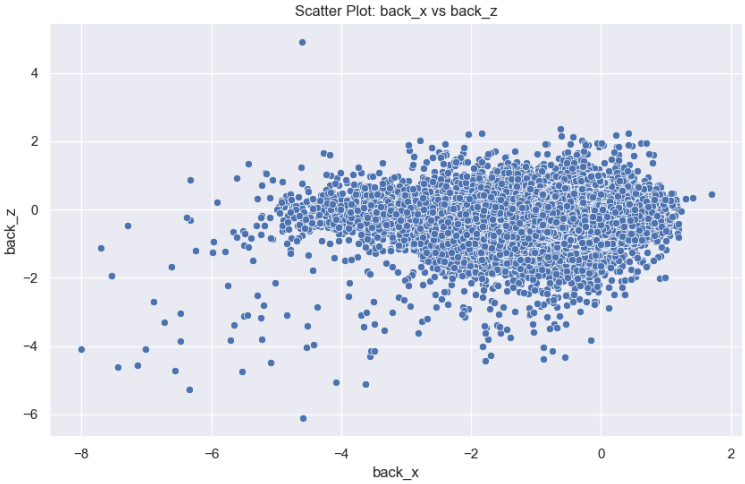


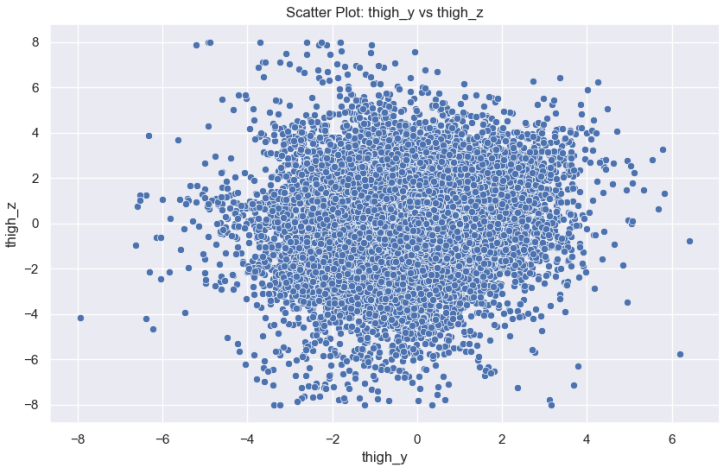


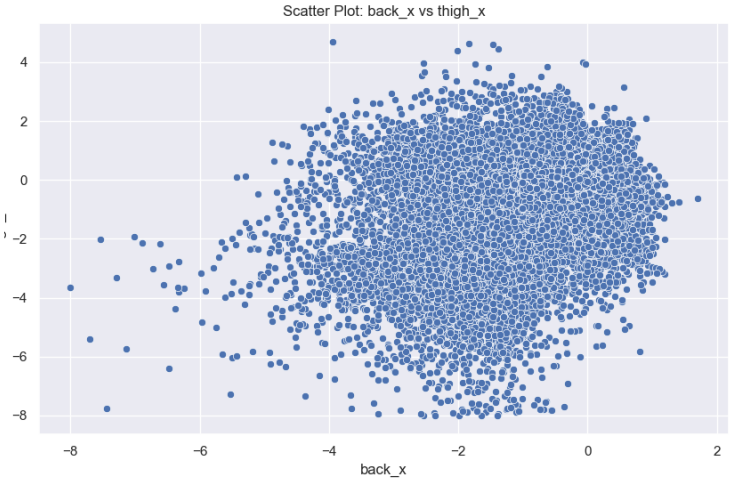


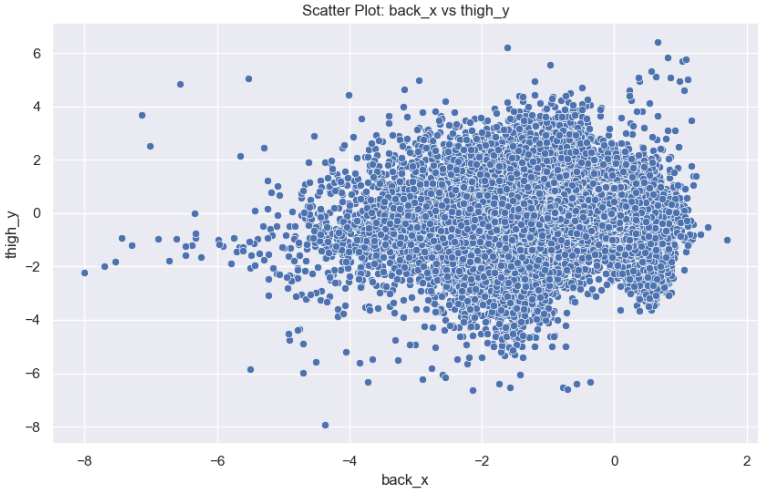


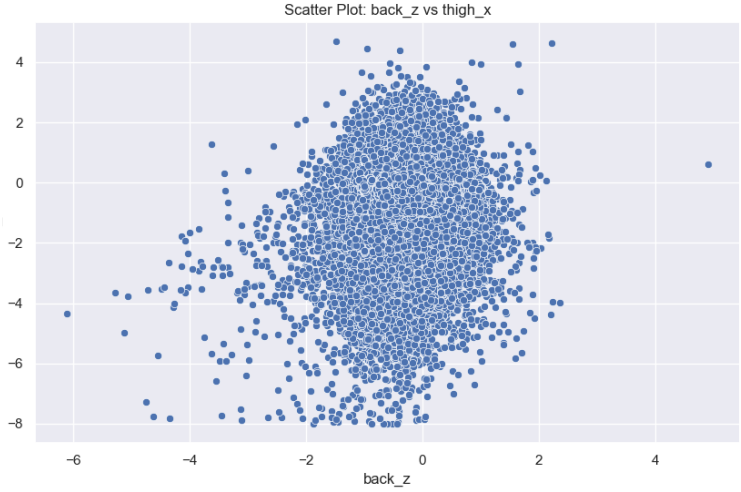


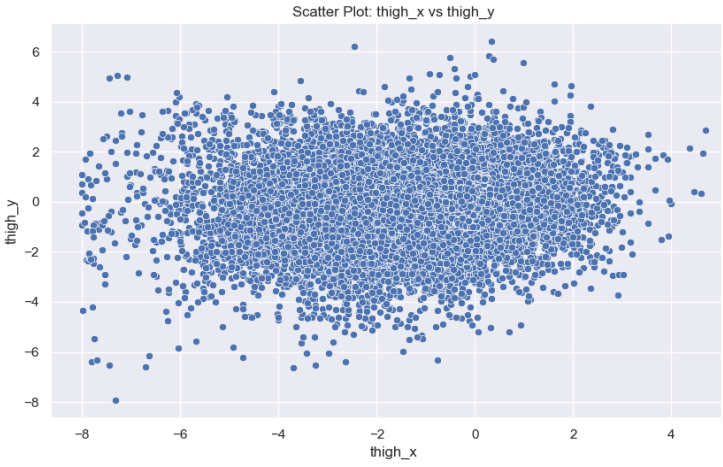


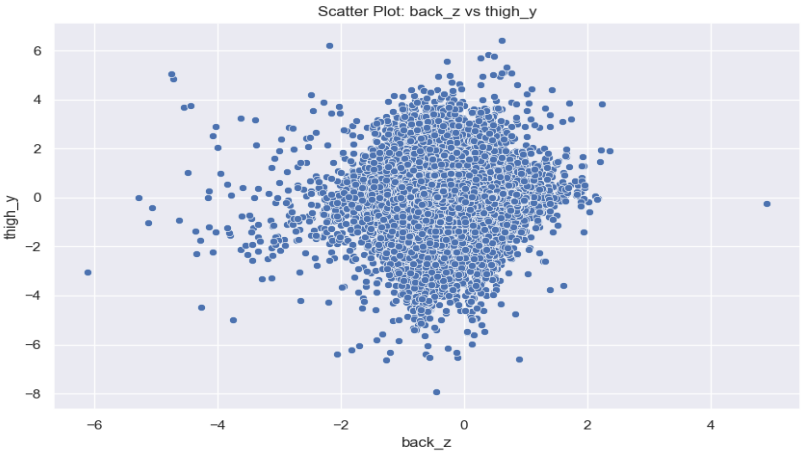

Scatter Plot: thigh_y vs thigh_z
thigh_z
thigh_y

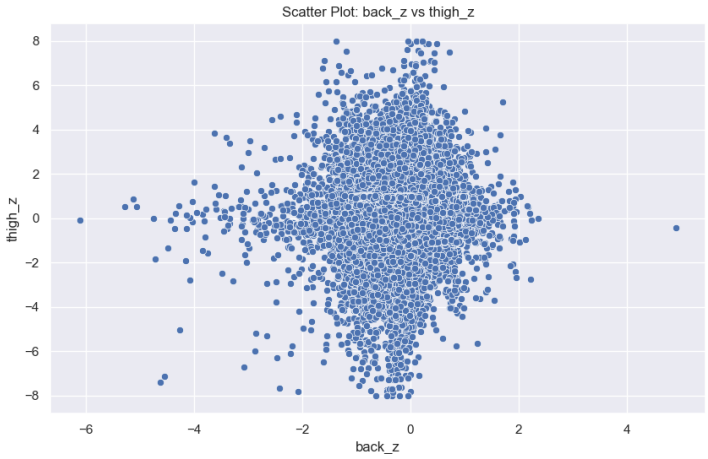

Scatter Plot: back_x vs thigh_x
back_x

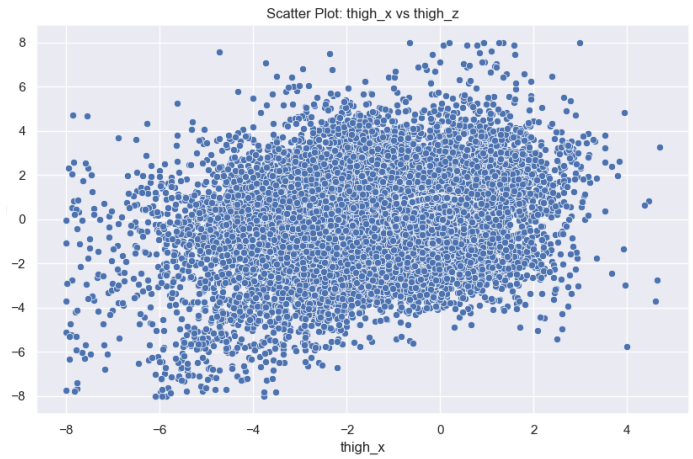

Scatter Plot: back_x vs thigh_y
thigh_y
back_x


Scatter Plot: back_z vs thigh_x
back_z

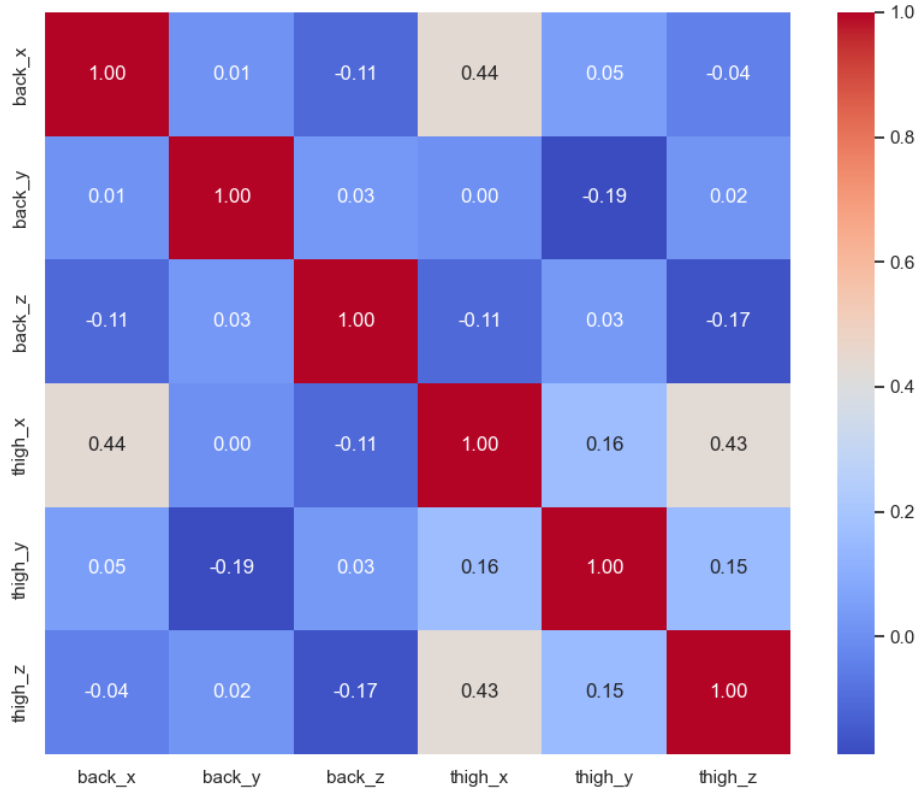

Scatter Plot: thigh_x vs thigh_y
thigh_y
thigh_x


Scatter Plot: back_z vs thigh_y
thigh_y
back_z





Απο τα scatter plot δεν φαινέται καποια μεταβλητή να έχει ισχυρή συσχέτηση με κάποια άλλη ,ωστόσο θα φτιάξουμε και ένα Correlation Heatmap χρησιμοποιώντας την .heatmap της βιβλιοθήκης seaborn .



Το ίδιο βλέπουμε και στο heatmap αφου δεν υπάρχει καποιος αριθμός κοντά στο 1 ή στο -1 για να θεωρήσουμε οτι υπάρχει κάποια ισχυρή θετική ή αρνητική συσχέτιση .

Ερώτημα 2

Στο 2ο ερώτημα θα υλοποιήσουμε και θα εκπαιδέυσουμε 3 ταξινομητές : έναν βασισμένο σε Neural Networks, έναν σε Random Forests και έναν σε Bayesian Networks

Neural Networks :

Θα χρησιμοποιήσουμε ένα ενα νευρωνικό δίκτυο (μοντέλο Sequential της tensorflow.keras) για να μαντέψουμε το label μιας εγγραφής απο τους Axivity AX3 accelerometer . Για τήν εκπαίδευση και δοκιμή του μοντέλου χρησιμοποιήσαμε ενα 70-30 training-test split του ενοποιημένου dataset. Πριν τροφοδοτήσουμε το μοντέλο με τα dataframes εκπαίδευσης και δοκιμής, θα κάνουμε ενα sample (frac=0.1) και OneHotEncoding για τα labels τα οποία θα οριστούν ως έξοδο στο δίκτυο.Τέλος ορίσαμε early stopping με patience=3 στο val_accuracy, mode=max .Οι μετρήσεις που κάναμε ηταν οι εξής :

```
Epoch 1/50
12721/12721 [==============================] - 36s 3ms/step - loss: 0.5366 - accuracy: 0.8257 - val_loss: 0.4705 - val_accuracy: 0.8476
Epoch 2/50
12721/12721 [==============================] - 30s 2ms/step - loss: 0.4712 - accuracy: 0.8447 - val_loss: 0.4570 - val_accuracy: 0.8522
Epoch 3/50
12721/12721 [==============================] - 31s 2ms/step - loss: 0.4579 - accuracy: 0.8484 - val_loss: 0.4416 - val_accuracy: 0.8552
Epoch 4/50
12721/12721 [==============================] - 31s 2ms/step - loss: 0.4494 - accuracy: 0.8508 - val_loss: 0.4393 - val_accuracy: 0.8543
Epoch 5/50
12721/12721 [==============================] - 31s 2ms/step - loss: 0.4434 - accuracy: 0.8526 - val_loss: 0.4361 - val_accuracy: 0.8566
Epoch 6/50
12721/12721 [==============================] - 31s 2ms/step - loss: 0.4384 - accuracy: 0.8541 - val_loss: 0.4327 - val_accuracy: 0.8575
Epoch 7/50
12721/12721 [==============================] - 31s 2ms/step - loss: 0.4348 - accuracy: 0.8550 - val_loss: 0.4290 - val_accuracy: 0.8567
Epoch 8/50
12721/12721 [==============================] - 31s 2ms/step - loss: 0.4322 - accuracy: 0.8554 - val_loss: 0.4314 - val_accuracy: 0.8572
Epoch 9/50
12721/12721 [==============================] - 31s 2ms/step - loss: 0.4295 - accuracy: 0.8564 - val_loss: 0.4418 - val_accuracy: 0.8527
Epoch 9: early stopping
6058/6058 [==============================] - 12s 2ms/step - loss: 0.4472 - accuracy: 0.8503
```

	precision	recall	f1-score	support
0	0.76	0.70	0.73	36015
1	0.84	0.60	0.70	8731
2	0.42	0.03	0.06	7480
3	0.59	0.00	0.01	2289
4	0.00	0.00	0.00	1955
5	0.85	0.65	0.73	22209
6	0.99	0.99	0.99	87284
7	1.00	1.00	1.00	12794
8	0.80	0.77	0.78	11882
9	0.53	0.47	0.50	1711
10	0.69	0.16	0.25	1250
11	0.44	0.06	0.10	240
micro avg	0.91	0.80	0.85	193840
macro avg	0.66	0.45	0.49	193840
weighted avg	0.87	0.80	0.82	193840
samples avg	0.80	0.80	0.80	193840

Random Forests:

Δημιουργήσαμε και εκπαιδέυσαμε εναν Random Forest Classifier χρησιμοποιώντας 100 δέντρα (estimators) και παράλληλη εκπαίδευση (n_jobs=-1) για την επιτάχυνση της διεργασίας αλλα

και για να χρησιμοποιήσουμε λιγοτερο sampling αφου θα χρειαστεί λιγότερος χρόνος.Τα αποτελέσματα ηταν τα εξής:

```
Classification Report:

              precision    recall  f1-score   support

          13       0.77      0.89      0.83    107773
           7       0.91      0.88      0.90     26198
           1       0.50      0.28      0.36     22689
           2       0.64      0.19      0.29      6899
           8       0.60      0.06      0.10      6055
           6       0.83      0.88      0.86     66948
           3       1.00      1.00      1.00    261595
          14       1.00      1.00      1.00     38695
           5       0.81      0.88      0.84     35182
           4       0.73      0.57      0.64      5019
         130       0.66      0.44      0.52      3774
         140       0.70      0.43      0.53       693

    accuracy                           0.90    581520
   macro avg       0.76      0.62      0.66    581520
weighted avg       0.89      0.90      0.89    581520

ACCURACY OF THE MODEL: 0.8962546430045398
```

Bayesian Networks:

Στο τέλος δημιουργήσαμε και εκπαιδέυσαμε ενα Naive Bayes Classifier με train-test split 70-30 και optimization σε search space (=10000) για να βρεθεί το καλύτερο state .Τα features δεχτηκαν scaling μεσω ενος Min-Max Scaler .Ως μετρική χρησιμοποιήθηκε το accuracy.

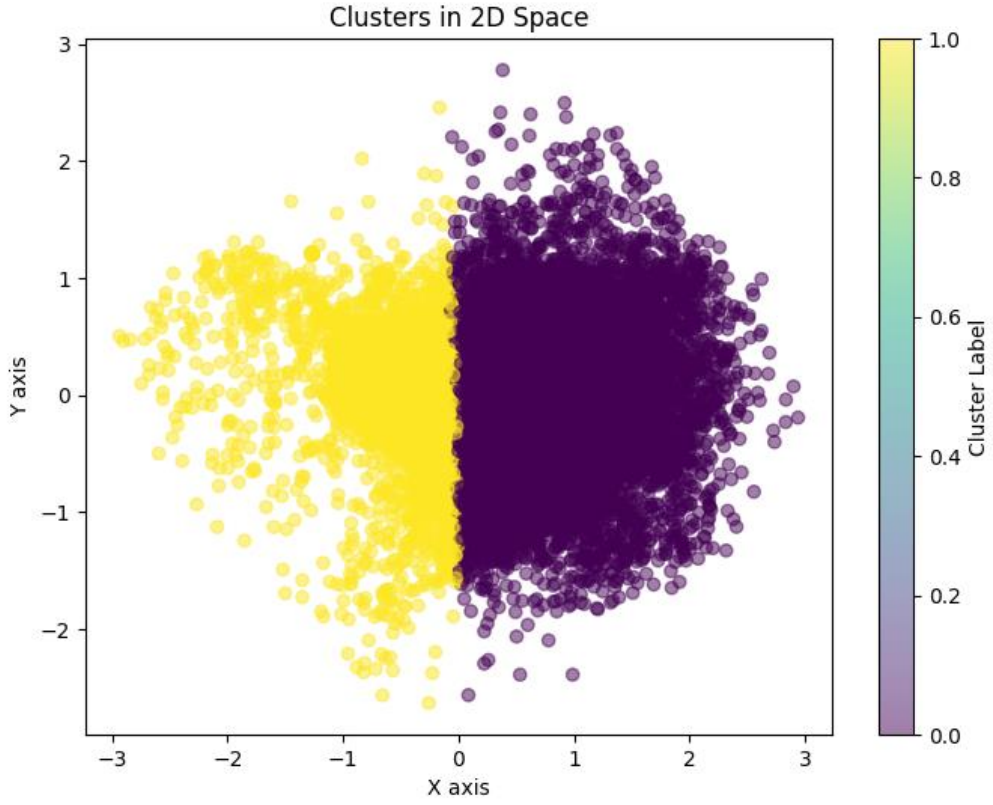
```
Best accuracy: 0.7756514525647197
Best random state: 8330
```

Ερώτημα 3

Στο 3ο ερώτημα επιλέχθηκε ως 1ος αλγόριθμος ο k-means με k-means++ αρχικοποιήση ένας απο τους πιο γνωστούς αλγόριθμος αν όχι ο πιο γνωστός και εύρεος διαδεδομένος και ως 2ος ο agglomerative clustering algorithm της sklearn.clustering ο οποίος πρόκειται γιά εναν ιεραρχικό αλγόριθμο.

k-means:

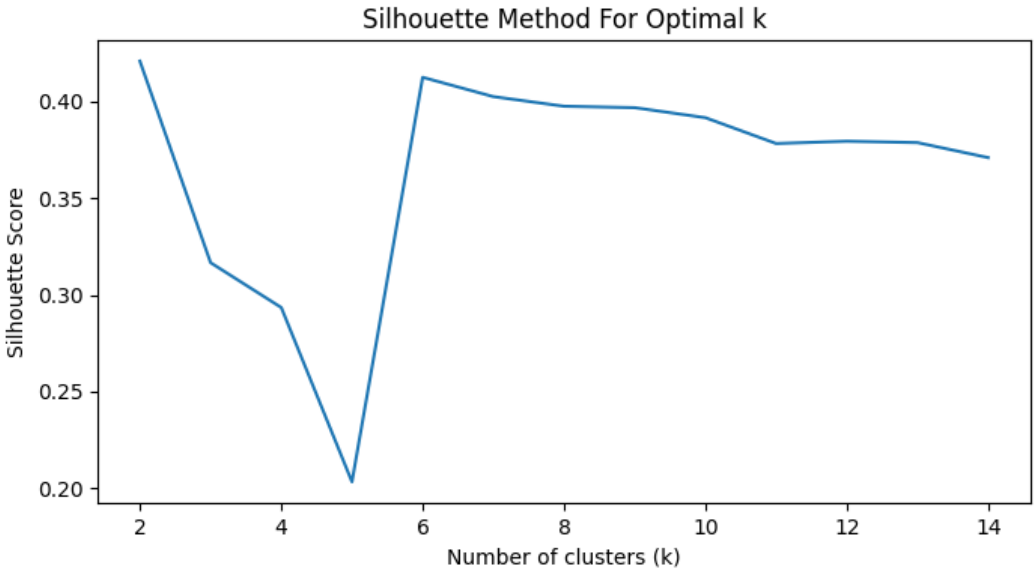
Αρχικά απο τα data κρατήσαμε μονο τις 2 τελευταίες ώρες με την Timedelta(hours =2) της βιβλιοθήκης pandas. Έπειτα διότι ο k-means είναι ευαίσθητος στους outliers εφαρμόστηκε ένα z-score filtering στα data για να διαγραφούν πιθανοί outliers.Για την σωστή επιλογη του k χρησιμοποιήθηκε το silhouette score για k (2,15) και στο τέλος χρησιμοποιήσαμε το PCA της sklearn.decomposition για να μειώσουμε τις διαστάσεις σε 2 και να κάνουμε plot τα clusters αλλα και οι μετρικές silhouette και Calinski-Harabasz Index . Αποτελέσμετα :

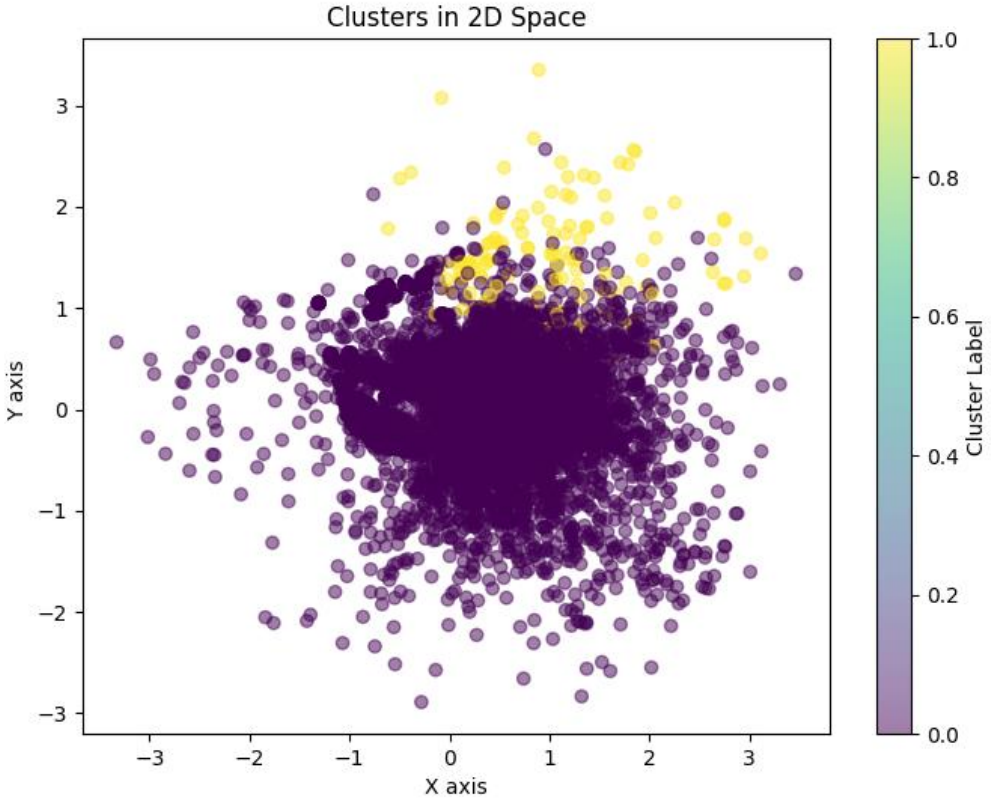



```
Optimal number of clusters based on silhouette score: 2
Silhouette Score for k = 2 is: 0.4711586386217333
Final Calinski-Harabasz Index: 49424.73476036874
```

Agglomerative Clustering:

Ο Agglomerative αλγόριθμος πρόκειται για εναν ιεραρχικό αλγόριθμο ο οποίος επαναληπτικά ενώνει δυο clusters και πρόκειται για εναν αλγόριθμο τυπου ¨rich gets richer¨για αυτο τις περισσότερες φορές δημιουργεί clusters ανομοιόμορφου μεγέθους για αυτο το λόγο επιλέχθηκε ως μετρική η ‘cosine’ (cosine similarity) η οποία λειτουργεί πολυ καλά για πολυδιάστατους χώρους και για διανύσματα με“ linkage = average ”.






```
Optimal number of clusters based on silhouette score: 2
Final Silhouette Score: 0.4209877210685016
Final Calinski-Harabasz Index: 406.6667275435522
```

Σύγκριση :

Παρόλο που και οι δύο αλγόριθμοι ομαδοποίησαν τα δεδομένα σε 2 clusters η χαμηλή πολυπλοκότητα του k-means μας επέτρεψε να έχουμε μεγαλύτερο δείγμα και πιο «καθαρο» clustering αφου παρατηρούμε οτι έχουμε ενα παρόμοιο silhouette score με του k means να είναι ελάχιστα μεγαλύτερο ωστόσο διαπιστώνουμε μεγαλη διαφορά κάτι που υποδηλώνει οτι η πρώτη ομαδοποιήση παρέχει μεγαλύτερη διαχωριστικότητα.

Παράρτημα

Το project υλοποιήθηκε σε περιβάλλον Microsoft Windows 11 και τα εργαλεία που χρησιμοποιήθηκαν ήταν η Python 3.11.9 και το Microsoft Visual Studio Code.
Πέρα από τα βασικά εργαλεία, χρησιμοποιήθηκε μια ποικιλία βιβλιοθηκών της Python για την υλοποίηση των ερωτημάτων. Παρουσιάζονται παρακάτω μαζί με τις εντολές εγκατάστασής τους:

- matplotlib (pip install -U matplotlib)
- pandas (pip install pandas)
- seaborn (pip install seaborn)
- numpy (pip install numpy)
- sklearn (pip install -U scikit-learn)
- skopt (pip install scikit-optimize)
- keras (pip install tensorflow και pip install keras)