NATIONAL CENTRE FOR
SCIENTIFIC RESEARCH "DEMOKRITOS"

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
UNIVERSITY OF PIRAEUS

Δ.Π.Μ.Σ.
ΤΕΧΝΗΤΗ
ΝΟΗΜΟΣΥΝΗ

Object Classification for Autonomous Driving: Comparing ML Models

Αλέξανδρος Νταγιάντας, Ηρακλής Σπύρου

Μηχανική Μάθηση

Ιανουάριος 2025

Περίληψη

Η παρούσα εργασία διερευνά την ταξινόμηση αντικειμένων για εφαρμογές αυτόνομης οδήγησης, αξιοποιώντας διάφορα μοντέλα μηχανικής μάθησης. Αρχικά, χρησιμοποιείται το **KITTI dataset**, από το οποίο πραγματοποιείται το **cropping** των εικόνων με σκοπό την απομόνωση των αντικειμένων ενδιαφέροντος (π.χ. Cars, Pedestrians, Cyclists, Trams). Στη συνέχεια, εφαρμόζεται ο αλγόριθμος **HOG** (Histogram of Oriented Gradients) για το feature extraction, παρέχοντας μια αναπαράσταση που αποδεικνύεται ανθεκτική σε μεταβολές φωτισμού και μικρές παραμορφώσεις. Τα δεδομένα διαχωρίζονται σε σύνολα **training**, **validation** και **testing**, ενώ υλοποιούνται τεχνικές pre-processing όπως **scaling**, η εξισορρόπηση μέσω **SMOTE/undersampling** και η μείωση της διαστατικότητας με τη χρήση **PCA**. Το training διεξάγεται στα μοντέλα **SVM**, **k-NN**, **Gaussian Naive Bayes** και **Random Forests**, με την επιλογή των βέλτιστων υπερπαραμέτρων μέσω **k-fold cross validation**. Τα πειραματικά αποτελέσματα δείχνουν ότι τα μοντέλα **SVM** και **k-NN** απέδωσαν καλύτερα συνολικά, ενώ τα μοντέλα **GNB** και **RF** αντιμετώπισαν προκλήσεις στις κατηγορίες **Cyclist** και **Tram**, οι οποίες είχαν και τα λιγότερα instances. Η εφαρμογή μεθόδων εξισσορόπησης αποδείχθηκε σημαντική για τα Random Forests, ενώ στους k-NN και GNB έριξε την απόδοση τους. Τέλος, τα μοντέλα συγκρίθηκαν με classification report, confusion matrices και ROC curves. Συνολικά, η εργασία καταδεικνύει τη σημασία της σωστής προεπεξεργασίας και της επιλογής κατάλληλων χαρακτηριστικών και υπεραπαραμέτρων για την επιτυχημένη ταξινόμηση αντικειμένων σε συστήματα αυτόνομης οδήγησης.

Περιεχόμενα

1 Εισαγωγή

Η αυτόνομη οδήγηση αποτελεί ένα από τα πλέον αναπτυσσόμενα πεδία στην τεχνολογία των τελευταίων ετών, με την υπόσχεση να βελτιώσει την ασφάλεια, την αποδοτικότητα και την άνεση των μεταφορών. Στο επίκεντρο της ανάπτυξης των αυτόνομων οχημάτων βρίσκεται η ικανότητά τους να αντιλαμβάνονται και να αλληλεπιδρούν με το περιβάλλον τους με ακρίβεια και αξιοπιστία.
Η ταξινόμηση αντικειμένων, όπως πεζοί, αυτοκίνητα, τραμ και ποδηλάτες, είναι κρίσιμη για την ασφαλή λειτουργία των αυτόνομων οχημάτων. Η ακριβής αναγνώριση αυτών των αντικειμένων επιτρέπει στα οχήματα να λαμβάνουν ορθές αποφάσεις, αποφεύγοντας πιθανές συγκρούσεις και διασφαλίζοντας την ασφάλεια των επιβατών και των πεζών.
Η παρούσα εργασία επικεντρώνεται στη μελέτη και αξιολόγηση διαφόρων μοντέλων machine learning για την ταξινόμηση αντικειμένων σε περιβάλλοντα αυτόνομων οχημάτων. Συγκεκριμένα, εξετάζονται οι Support Vector Machines (SVM) και k-Nearest Neighbour (k-NN), Gaussian Naive Bayes (GNB), καθώς και Random Forests (RF), αξιολογώντας την απόδοσή τους με βάση διάφορες μετρικές.
Οι κύριοι στόχοι της εργασίας είναι η επιλογή των βέλτιστων παραμέτρων για κάθε μοντέλο, η εκπαίδευσή τους με τα επιλεγμένα δεδομένα, και η συγκριτική ανάλυση των αποτελεσμάτων προκειμένου να προσδιοριστεί το πιο αποτελεσματικό μοντέλο για την συγκεκριμένη εφαρμογή.
Η δομή του report είναι ως εξής: αρχικά θα παρουσιαστούν τα δεδομένα που χρησιμοποιήθηκαν καθώς και η εξαγωγή των features. Ύστερα θα περιγραφούν οι μεθοδολογίες και τα μοντέλα που εφαρμόστηκαν. Αργότερα θα παρουσιαστούν τα αποτελέσματα και η ανάλυσή τους, ενώ τέλος, θα εξαχθούν συμπεράσματα και προτάσεις για μελλοντική εργασία.

2 Dataset και feature extraction

2.1 Dataset

Για την παρούσα εργασία χρησιμοποιήθηκε το **KITTI dataset**, το οποίο αποτελεί ένα ευρέως χρησιμοποιούμενο σύνολο δεδομένων για την αυτόνομη οδήγηση. Συγκεκριμένα, χρησιμοποιήθηκαν τα ακόλουθα αρχεία:

- `data_object_image_2.zip`: Περιέχει έγχρωμες εικόνες (left color images) για το object dataset, συνολικού μεγέθους 12GB.
- `data_object_label.zip`: Περιέχει τις ετικέτες (labels) για το training set, συνολικού μεγέθους 5MB.

Το αρχείο με τις εικόνες περιέχει δύο υποφακέλους:

1. Ο πρώτος υποφάκελος περιέχει 7481 εικόνες για **training**.
2. Ο δεύτερος υποφάκελος περιέχει εικόνες για **testing**.

Αντίστοιχη διάταξη υπάρχει και για τα αρχεία των ετικετών.
Για τις ανάγκες της παρούσας εργασίας, πραγματοποιήθηκε **crop** των εικόνων ώστε να διατηρηθεί μόνο κάθε αντικείμενο από την εικόνα. Ως εκ τούτου, χρησιμοποιήθηκε μόνο το πρώτο αρχείο με τις εικόνες για training, καθώς μετά το cropping παράχθηκαν συνολικά **24613 εικόνες**, αριθμός επαρκής για τη δημιουργία των *train*, *validation*, και *test* datasets.

Το αρχείο labels του KITTI dataset περιέχει πληροφορίες σχετικά με τα αντικείμενα που έχουν ανιχνευθεί σε κάθε εικόνα. Κάθε γραμμή του αρχείου αντιστοιχεί σε ένα αντικείμενο και περιλαμβάνει τα εξής πεδία:

- **Object Type**: Ο τύπος του αντικειμένου.

- **Truncation**: Επίπεδο περικοπής του αντικειμένου (0 = πλήρως ορατό, 1 = εντελώς εκτός εικόνας).
- **Occlusion**: Επίπεδο απόκρυψης από άλλα αντικείμενα (0 = όχι κρυμμένο, 1 = μερικώς κρυμμένο, 2 = εντελώς κρυμμένο).
- **Alpha**: Ο παρατηρούμενος προσανατολισμός του αντικειμένου.
- **Bounding Box**: Οι συντεταγμένες του ορθογώνιου περιγράμματος (`xmin, ymin, xmax, ymax`).
- **3D Dimensions**: Οι διαστάσεις του αντικειμένου στο 3D χώρο (`height, width, length`).
- **3D Location**: Η θέση του αντικειμένου στο 3D χώρο (`x, y, z`).
- **Rotation Y**: Ο προσανατολισμός του αντικειμένου γύρω από τον κάθετο άξονα.

Παρακάτω φαίνεται μία τυχαία εικόνα του dataset (0008.png) καθώς και το αντίστοιχο label της (0008.txt).

Figure 1: Παράδειγμα εικόνας από το KITTI dataset.

```
Car 0.88 3 -0.69 0.00 192.37 402.31 374.00 1.60 1.57 3.23 -2.70 1.74 3.68 -
1.29
Car 0.00 1 2.04 334.85 178.94 624.50 372.04 1.57 1.50 3.68 -
1.17 1.65 7.86 1.90
Car 0.34 3 -1.84 937.29 197.39 1241.00 374.00 1.39 1.44 3.08 3.81 1.64 6.15 -
1.31
Car 0.00 1 -1.33 597.59 176.18 720.90 261.14 1.47 1.60 3.66 1.07 1.55 14.44 -
1.25
Car 0.00 0 1.74 741.18 168.83 792.25 208.43 1.70 1.63 4.08 7.24 1.55 33.20 1.95
Car 0.00 0 -1.65 884.52 178.31 956.41 240.18 1.59 1.59 2.47 8.48 1.75 19.96 -
1.25
DontCare -1 -1 -10 800.38 163.67 825.45 184.07 -1 -1 -1 -1000 -1000 -1000 -10
DontCare -1 -1 -10 859.58 172.34 886.26 194.51 -1 -1 -1 -1000 -1000 -1000 -10
DontCare -1 -1 -10 801.81 163.96 825.20 183.59 -1 -1 -1 -1000 -1000 -1000 -10
DontCare -1 -1 -10 826.87 162.28 845.84 178.86 -1 -1 -1 -1000 -1000 -1000 -10
```

Οι κατηγορίες `DontCare` υποδεικνύουν περιοχές που πρέπει να αγνοηθούν κατά την εκπαίδευση του μοντέλου, ενώ οι υπόλοιπες κατηγορίες (π.χ., `Car`) χρησιμοποιούνται για την εκπαίδευση και την αξιολόγηση.
Γίνεται αντιληπτό το γεγονός ότι χρειάζεται να γίνει περικοπή στα αντικείμενα ενδιαφέροντος, καθώς για να γίνει αναγνώριση αντικειμένων σε αυτού του είδους τις εικόνες χρειάζονται τεχνικές deep learning (πχ YOLO), κάτι το οποίο είναι πέραν των στόχων της συγκεκριμένης εργασίας.

2.2 Cropping Εικόνων

Για την εξαγωγή των αντικειμένων από τις εικόνες του KITTI dataset, χρησιμοποιήθηκε η βιβλιοθήκη `OpenCV` στο αρχείο κώδικα *cropping.py*. Συγκεκριμένα, πραγματοποιήθηκαν τα εξής βήματα:

- Τα δεδομένα φόρτωσης περιλάμβαναν τις εικόνες από το φάκελο `image_2` και τα αντίστοιχα labels από το φάκελο `label_2`.
- Εφαρμόστηκε φιλτράρισμα ώστε να αγνοηθούν αντικείμενα με κλάσεις που δεν ανήκουν στις κύριες κατηγορίες (`Car`, `Pedestrian`, `Cyclist`, `Tram`).
- Απορρίφθηκαν αντικείμενα με *occlusion* μεγαλύτερο από 1 ή *truncation* μεγαλύτερο από 0.40.
- Πραγματοποιήθηκε περικοπή των αντικειμένων με βάση τις συντεταγμένες του bounding box, ενώ οι cropped εικόνες αποθηκεύτηκαν σε ξεχωριστό φάκελο.
- Δημιουργήθηκε νέο αρχείο labels που περιέχει την αντιστοιχία κάθε cropped εικόνας με την κλάση του αντικειμένου της.

Συνολικά, παράχθηκαν 24,613 cropped εικόνες. Παρακάτω ακολουθούν μερικές από τις περικομμένες εικόνες.

(a) Pedestrian

(b) Car

(c) Car (Occluded)

Figure 2: Παραδείγματα cropped εικόνων από το KITTI dataset.

2.3 Feature Extraction

Για την εξαγωγή χαρακτηριστικών από τις cropped εικόνες του KITTI dataset, χρησιμοποιήθηκε η βιβλιοθήκη `OpenCV` και ο αλγόριθμος HOG (Histogram of Oriented Gradients). Ο αλγόριθμος HOG εξάγει χαρακτηριστικά που βασίζονται στην κατανομή των προσανατολισμών ακμών, επιτυγχάνοντας ανθεκτικότητα σε αλλαγές φωτισμού και μικρές παραμορφώσεις. Ο κώδικας που χρησιμοποιήθηκε βρίσκεται στο αρχείο *HOG_feature_extraction.py*.
Η διαδικασία περιλάμβανε τα εξής βήματα:

- Μετατροπή των εικόνων σε grayscale.
- Resizing κάθε εικόνας σε σταθερό μέγεθος 128×64 pixels.
- Εξαγωγή χαρακτηριστικών HOG με τις εξής παραμέτρους:
  - **Μέγεθος παραθύρου (Window Size)**: 128×64 pixels.
  - **Μέγεθος μπλοκ (Block Size)**: 16×16 pixels.
  - **Βήμα μπλοκ (Block Stride)**: 8×8 pixels.
  - **Μέγεθος κυψελών (Cell Size)**: 8×8 pixels.
  - **Αριθμός bins (Nbins)**: 9 bins.
- Αποθήκευση των χαρακτηριστικών σε αρχείο `CSV`, όπου κάθε γραμμή περιέχει τα χαρακτηριστικά μιας εικόνας και την ετικέτα της (`class`).

Συνολικά, το αρχείο `visible_hog_feature.csv` περιέχει όλα τα χαρακτηριστικά που εξήχθησαν (3780 features και 24614 instances) και αποτέλεσαν τη βάση για την εκπαίδευση και την αξιολόγηση των μοντέλων.

3 Μοντέλα και διαδικασία εκπαίδευσης

3.1 Εισαγωγή στα μοντέλα

Για την ταξινόμηση των αντικειμένων στις εικόνες, επιλέχθηκαν τέσσερα μοντέλα machine learning: Support Vector Machine (SVM), k-Nearest Neighbors (k-NN), Gaussian Naive Bayes (GNB), και Random Forests. Η επιλογή αυτών των μοντέλων βασίστηκε στην καταλληλότητά τους για classification προβλήματα και στην ικανότητά τους να διαχειρίζονται δεδομένα διαφορετικών κατανομών και πολυπλοκότητας. Τα SVM και k-NN είναι αλγόριθμοι που βασίζονται σε γεωμετρικές ιδιότητες των δεδομένων, ενώ τα GNB και Random Forests είναι πιθανοτικά μοντέλα. Αυτή η ποικιλία επιτρέπει μια συγκριτική ανάλυση απόδοσης σε διαφορετικές συνθήκες.

3.2 Θεωρητικό υπόβαθρο κάθε μοντέλου

3.2.1 Support Vector Machine (SVM)

Θεωρητική Περιγραφή του Μοντέλου: Ο Support Vector Machine (SVM) είναι ένας διαχωριστικός αλγόριθμος που βασίζεται στην εύρεση ενός υπερεπιπέδου (*hyperplane*) που μεγιστοποιεί το περιθώριο (*margin*) μεταξύ των δεδομένων διαφορετικών κατηγοριών. Αν τα δεδομένα δεν είναι γραμμικά διαχωρίσιμα, χρησιμοποιούνται πυρήνες (*kernels*) για την προβολή τους σε υψηλότερη διάσταση, όπου μπορούν να διαχωριστούν.
Πώς λειτουργεί: Το μοντέλο εκπαιδεύεται με τα δεδομένα ώστε να εντοπίσει το υπερεπίπεδο που διαχωρίζει τις κατηγορίες με τον καλύτερο τρόπο. Στην παρούσα εργασία επιλέχθηκε ο **RBF Kernel** Η απόφαση λαμβάνεται με βάση τη θέση του κάθε σημείου σε σχέση με το υπερεπίπεδο και σκοπός είναι η ελαχιστοποίηση της παρακάτω αντικειμενικής συνάρτησης [1] [2] :

$$\min \frac{1}{2}\|w\|^2 + C \sum_i \max(0, 1 - y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b))$$

Όπου:

- $\frac{1}{2}\|w\|^2$: Ο όρος κανονικοποίησης (*regularization term*), ο οποίος ελαχιστοποιεί το μέγεθος του διανύσματος βαρών w. Αυτό βοηθά στη μεγιστοποίηση του margin μεταξύ των κατηγοριών, ενώ ταυτόχρονα αποτρέπει το overfitting.
- C: Υπερπαράμετρος που ελέγχει το *trade-off* μεταξύ του όρου κανονικοποίησης και του κόστους σφαλμάτων ταξινόμησης.
  - Μεγάλο C: Το μοντέλο δίνει μεγαλύτερη έμφαση στην ελαχιστοποίηση των σφαλμάτων ταξινόμησης.
  - Μικρό C: Το μοντέλο εστιάζει περισσότερο στη μεγιστοποίηση του margin.
- $\max(0, 1 - y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b))$: Ο όρος *Hinge Loss*, ο οποίος μετρά το κόστος για το δείγμα i.
  - Αν $y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) \geq 1$: Το δείγμα ταξινομήθηκε σωστά και το κόστος είναι 0.
  - Αν $y^{(i)}(\mathbf{w}^T \mathbf{x}^{(i)} + b) < 1$: Το δείγμα είναι είτε λανθασμένα ταξινομημένο είτε εντός του margin, και το κόστος αυξάνεται ανάλογα.
- $\mathbf{w}$: Το διάνυσμα βαρών που καθορίζει το υπερεπίπεδο ταξινόμησης.
- b: Ο bias όρος που μετατοπίζει το υπερεπίπεδο.

- $y^{(i)}$: Η ετικέτα του δείγματος i, όπου $y^{(i)} \in \{-1, 1\}$.
- $\mathbf{x}^{(i)}$: Το διάνυσμα χαρακτηριστικών του δείγματος i.

3.2.2 k-Nearest Neighbors (k-NN)

Θεωρητική Περιγραφή του Μοντέλου: Το k-Nearest Neighbors (k-NN) είναι ένας αλγόριθμος ταξινόμησης που βασίζεται στην ομοιότητα των δεδομένων [3] [1]. Η Ευκλίδεια απόσταση υπολογίζεται όπως παρακάτω:

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{i=1}^{d} (x_{1,i} - x_{2,i})^2}$$

Όπου:

- $d(\mathbf{x}_1, \mathbf{x}_2)$: Η Ευκλείδεια απόσταση μεταξύ των δύο σημείων.
- d: Ο αριθμός των διαστάσεων (features) του χώρου.
- $x_{1,i}$: Η τιμή του i-ου χαρακτηριστικού του πρώτου σημείου $\mathbf{x}_1$.
- $x_{2,i}$: Η τιμή του i-ου χαρακτηριστικού του δεύτερου σημείου $\mathbf{x}_2$.

Πώς λειτουργεί: Υπολογίζεται η ευκλείδεια (ή άλλη) απόσταση μεταξύ του δείγματος και όλων των σημείων στο dataset. Η ετικέτα του δείγματος καθορίζεται από την πλειοψηφία των k πλησιέστερων γειτόνων. Η υπερπαράμετρος k τίθεται προς εύρεση.

3.2.3 Gaussian Naive Bayes (GNB)

Θεωρητική Περιγραφή του Μοντέλου: Το Gaussian Naive Bayes (GNB) βασίζεται στον θεώρημα του Bayes και υποθέτει ότι τα χαρακτηριστικά κάθε κατηγορίας ακολουθούν κανονική κατανομή (*Gaussian distribution*). Εφαρμόζει την υπόθεση της ανεξαρτησίας μεταξύ των χαρακτηριστικών, ακόμα κι αν αυτό δεν ισχύει πάντα.
Πώς λειτουργεί: Υπολογίζει την πιθανότητα κάθε δείγματος να ανήκει σε μια κατηγορία, με βάση τις πιθανότητες των χαρακτηριστικών και της κατανομής της κατηγορίας. Το δείγμα ταξινομείται στην κατηγορία με τη μεγαλύτερη πιθανότητα [4] [1].
Το θεώρημα του Bayes υποδεικνύει ότι:

$$P(y_k|\mathbf{x}) = \frac{P(\mathbf{x}|y_k)P(y_k)}{P(\mathbf{x})}$$

Όπου:

- $P(y_k|\mathbf{x})$: Η πιθανότητα το δείγμα $\mathbf{x}$ να ανήκει στην κατηγορία y_k.
- $P(\mathbf{x}|y_k)$: Η πιθανότητα να παρατηρηθεί το δείγμα $\mathbf{x}$, δεδομένου ότι ανήκει στην κατηγορία y_k.
- $P(y_k)$: Η πιθανότητα της κατηγορίας y_k (εκ των προτέρων πιθανότητα).
- $P(\mathbf{x})$: Η συνολική πιθανότητα του δείγματος $\mathbf{x}$ (θεωρείται σταθερή για όλες τις κατηγορίες και δεν επηρεάζει τη σύγκριση).

Επίσης, η κανονική (Gaussian) κατανομή δίνεται από τη σχέση:

$$P(x_i|y_k) = \frac{1}{\sqrt{2\pi\sigma_{k,i}^2}} \exp\left(-\frac{(x_i - \mu_{k,i})^2}{2\sigma_{k,i}^2}\right)$$

Όπου:

- $\mu_{k,i}$: Η μέση τιμή του χαρακτηριστικού x_i για την κατηγορία y_k.
- $\sigma^2_{k,i}$: Η διακύμανση του χαρακτηριστικού x_i για την κατηγορία y_k.
- x_i: Η τιμή του χαρακτηριστικού x_i στο δείγμα.

Λαμβάνοντας υπόψιν την υπόθεση της ανεξαρτησίας (Naive Assumption for Independence):

$$P(\mathbf{x}|y_k) = \prod_{i=1}^{d} P(x_i|y_k)$$

καταλήγουμε στην τελική απόφαση:

$$y_{\text{pred}} = \arg\max_{y_k} P(y_k) \prod_{i=1}^{d} P(x_i|y_k)$$

3.2.4 Random Forests

Θεωρητική Περιγραφή του Μοντέλου: Τα Random Forests είναι ένα σύνολο (*ensemble*) αλγορίθμων που αποτελείται από πολλά δέντρα απόφασης. Κάθε δέντρο εκπαιδεύεται σε διαφορετικό υποσύνολο των δεδομένων, ενώ η απόφαση λαμβάνεται με πλειοψηφική ψήφο. Ελαχιστοποιούν το πρόβλημα του υπερπροσδιορισμού (*overfitting*) που παρατηρείται συχνά στα μεμονωμένα δέντρα απόφασης.
Πώς λειτουργεί: Κατά την εκπαίδευση, κάθε δέντρο δημιουργείται από ένα τυχαίο υποσύνολο δεδομένων (*bagging*). Για ένα νέο δείγμα, κάθε δέντρο προβλέπει μια κατηγορία και η τελική πρόβλεψη είναι η πλειοψηφία των προβλέψεων όλων των δέντρων. Τα βασικά βήματα του αλγορίθμου για ταξινόμηση είναι τα εξής [5] [1]:

- **Δημιουργία Δέντρων (Bagging):** Κάθε δέντρο εκπαιδεύεται σε ένα διαφορετικό υποσύνολο δεδομένων S_t, το οποίο δημιουργείται μέσω *bootstrap sampling* (δειγματοληψία με επανατοποθέτηση) από το αρχικό dataset S. Αυτό μειώνει την αλληλεξάρτηση μεταξύ των δέντρων.
- **Τυχαία Επιλογή Χαρακτηριστικών:** Κατά την εκπαίδευση κάθε δέντρου, σε κάθε διαχωρισμό κόμβου, επιλέγεται τυχαία ένα υποσύνολο χαρακτηριστικών m από τα συνολικά d. Συνήθως, $m = \sqrt{d}$ για ταξινομικά προβλήματα.
- **Πλειοψηφική Ψήφος:** Κατά την πρόβλεψη, κάθε δέντρο επιστρέφει μια κατηγορία, και η τελική κατηγορία για το δείγμα **x** υπολογίζεται ως η κατηγορία με τις περισσότερες ψήφους:

$$y_{\text{pred}} = \arg\max_{y_k} \sum_{t=1}^{T} \mathbb{I}(h_t(\mathbf{x}) = y_k)$$

  Όπου:

  - y_{pred}: Η τελική προβλεπόμενη κατηγορία.
  - y_k: Η k-οστή πιθανή κατηγορία.
  - $h_t(\mathbf{x})$: Η πρόβλεψη του t-οστού δέντρου για το δείγμα **x**.
  - $\mathbb{I}(h_t(\mathbf{x}) = y_k)$: Η ένδειξη που είναι ίση με 1 αν το t-οστό δέντρο προβλέπει την κατηγορία y_k, και 0 διαφορετικά.
  - T: Ο συνολικός αριθμός των δέντρων στο δάσος.

Δείκτης Gini (Gini Index): Κατά την εκπαίδευση κάθε δέντρου, η επιλογή του βέλτιστου διαχωρισμού σε έναν κόμβο γίνεται με βάση τον *Δείκτη Gini*, ο οποίος μετρά την "καθαρότητα" ενός κόμβου. Ο Δείκτης Gini ορίζεται ως [6]:

$$\text{Gini} = 1 - \sum_{k=1}^{K} p_k^2$$

Όπου:

- p_k: Η πιθανότητα ένα δείγμα να ανήκει στην κατηγορία k στον συγκεκριμένο κόμβο.
- K: Ο συνολικός αριθμός των κατηγοριών.

Τα Random Forests βελτιώνουν τη γενίκευση σε σχέση με τα μεμονωμένα δέντρα απόφασης, μειώνοντας τον κίνδυνο υπερπροσαρμογής (*overfitting*), χάρη στη χρήση της τυχαίας επιλογής δεδομένων και χαρακτηριστικών.

4 Διαδικασία Εκπαίδευσης

Σε αυτό το κεφάλαιο, περιγράφεται η διαδικασία εκπαίδευσης των μοντέλων ταξινόμησης που χρησιμοποιήθηκαν για την αναγνώριση αντικειμένων σε σενάρια αυτόνομης οδήγησης. Το κεφάλαιο χωρίζεται σε τρεις βασικές ενότητες:

- **Προεπεξεργασία δεδομένων:** Πώς διαμορφώνεται το dataset (train-validation-test), ποιες τεχνικές κανονικοποίησης και μείωσης διαστατικότητας εφαρμόζονται.
- **Επιλογή υπερπαραμέτρων:** Περιγραφή της διαδικασίας εύρεσης των βέλτιστων τιμών υπερπαραμέτρων για κάθε μοντέλο μέσω k-folds cross-validation.
- **Τελική εκπαίδευση και αξιολόγηση:** Πώς τα μοντέλα εκπαιδεύονται στην πλήρη dataset και αξιολογούνται σε άγνωστα δεδομένα (test set).

Όλα τα αποτελέσματα των μοντέλων, μαζί με συγκρίσεις μεταξύ τους, παρουσιάζονται αναλυτικά στο Κεφάλαιο 5.

4.1 Βιβλιοθήκες

Για την εκπαίδευση και αξιολόγηση των μοντέλων στο πλαίσιο της παρούσας εργασίας, χρησιμοποιήθηκαν διάφορες βιβλιοθήκες της Python. Κάθε βιβλιοθήκη εξυπηρετεί διαφορετικούς στόχους, όπως διαχείριση δεδομένων, επεξεργασία εικόνων, εξαγωγή χαρακτηριστικών, μηχανική μάθηση και οπτικοποίηση αποτελεσμάτων. Παρακάτω φαίνονται οι βιβλιοθήκες που χρησιμοποιήθηκαν.

Διαχείριση Αρχείων και Δεδομένων	
os	Διαχείριση αρχείων και λειτουργικού συστήματος
csv	Ανάγνωση και εγγραφή αρχείων CSV
pandas	Διαχείριση και προετοιμασία δεδομένων
numpy	Υπολογισμοί με πίνακες και αριθμητικές πράξεις
Επεξεργασία και Ανάλυση Εικόνας	
cv2 (OpenCV)	Επεξεργασία εικόνων και εξαγωγή HoG χαρακτηριστικών
Μηχανική Μάθηση και Προεπεξεργασία Δεδομένων	
sklearn	Μηχανική μάθηση και προεπεξεργασία δεδομένων
imblearn	Αντιμετώπιση μη ισορροπημένων datasets
Οπτικοποίηση Δεδομένων	
matplotlib	Οπτικοποίηση δεδομένων
seaborn	Προηγμένη οπτικοποίηση δεδομένων

Table 1: Βιβλιοθήκες και βασική τους χρήση

4.2 Προ-επεξεργασία Δεδομένων

Για την προεπεξεργασία των δεδομένων υλοποιήθηκαν τα παρακάτω βήματα:

1. **Φόρτωση των χαρακτηριστικών:** Το αρχείο `visible_hog_feature.csv` που περιέχει τα χαρακτηριστικά των εικόνων (HOG features) φορτώνεται μέσω της βιβλιοθήκης `pandas`.

2. **Διαχωρισμός μεταβλητών:** Δημιουργείται το σύνολο χαρακτηριστικών `X` (όλες οι στήλες εκτός από την τελευταία) και το σύνολο ετικετών `y` (η τελευταία στήλη).

3. **Διαχωρισμός σε Train+Validation και Test**: Χρησιμοποιώντας συναρτήσεις διαχωρισμού (*train_test_split*), το αρχικό σύνολο χωρίζεται σε δύο μέρη:

   - **Train+Validation:** Περιλαμβάνει το 80% των δεδομένων.
   - **Test:** Περιλαμβάνει το υπόλοιπο 20% των δεδομένων.

   Επιλέγεται *stratified sampling* ώστε η αναλογία των κλάσεων να παραμένει σταθερή και στα δύο υποσύνολα. Στο Σχήμα 3 φαίνεται η κατανομή των κλάσεων.

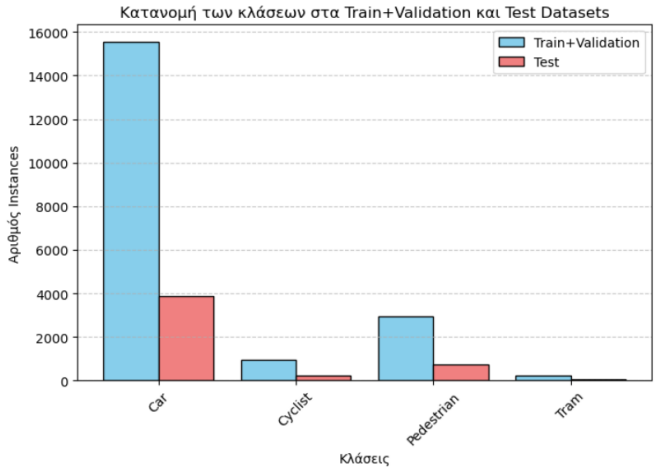


Figure 3: Κατανομή των κλάσεων στα Train+Validation και Test Datasets

4. **Αποθήκευση νέων συνόλων**: Τα καινούργια αρχεία `Train+Validation` και `Test` αποθηκεύονται σε ξεχωριστά `CSV`, διατηρώντας την ίδια δομή μεταβλητών (`X` και `y`).

Με τα παραπάνω βήματα, διασφαλίζεται σωστή οργάνωση των δεδομένων, καθώς και η δυνατότητα επανάληψης της διαδικασίας σε μελλοντικά στάδια ανάπτυξης ή βελτίωσης των μοντέλων.

4.3 Επιλογή Υπερπαραμέτρων

Η επιλογή των υπερπαραμέτρων (*hyperparameter tuning*) αποτελεί κρίσιμο στάδιο στη διαδικασία ανάπτυξης μοντέλων μηχανικής μάθησης, καθώς επιτρέπει τη βελτιστοποίηση της απόδοσής τους. Στην παρούσα εργασία, ακολουθήθηκε μια συστηματική μεθοδολογία με χρήση *K-Fold Cross Validation* για τον καθορισμό των βέλτιστων υπερπαραμέτρων στους τέσσερις ταξινομητές.

Default και Δοκιμαζόμενες Τιμές: Στον Πίνακα 2 συνοψίζονται οι τυπικές (default) τιμές που ορίζει η *scikit-learn* για κάθε αλγόριθμο, καθώς και το πλέγμα τιμών που εξερευνήθηκε στην εργασία αυτή:

Ταξινομητής	Default Τιμές	Πλέγμα Τιμών
SVM	`C=1.0`	$[0.1, 0.3, 0.5, 1.0, 2.0, 5.0]$
KNN	`n_neighbors=5`	$[3, 5, 7, 9, 11, 13, 15, 17, 19]$
GNB	`var_smoothing=1e-9`	`logspace(-5, -14, num=10)`
Random Forest	`n_estimators=100`	$[100, 150, 200, 250, 300]$

Table 2: Τυπικές (default) τιμές και εύρος τιμών για το *hyperparameter tuning* σε κάθε ταξινομητή.

Σημείωση: Για να διασφαλιστεί η επαναληψιμότητα των αποτελεσμάτων, **όλες** οι διαδικασίες που περιλαμβάνουν τυχαίο παράγοντα (π.χ. *train-test split*, *SMOTE*, *RandomUnderSampler*, *cross-validation*) εκτελέστηκαν ορίζοντας σταθερή τιμή `random_state` (εδώ.random_state= 42). Έτσι, τα πειράματα μπορούν να αναπαραχθούν παρόμοια στο μέλλον, διασφαλίζοντας ότι οι διαφορές στις επιδόσεις δεν οφείλονται σε κάποια τυχαία διαδικασία, αλλά μόνο στον εκάστοτε συνδυασμό υπερπαραμέτρων.

Βήματα και Κριτήρια Αξιολόγησης

1. **Ορισμός Πλέγματος Υπερπαραμέτρων:** Το εύρος τιμών που δοκιμάστηκε στον Πίνακα 2.
2. **Pipeline:**
   - *Scaling:* Γίνεται τυποποίηση (με *StandardScaler*) ώστε όλες οι μεταβλητές να έχουν παρόμοια κλίμακα.
   - *SMOTE & Undersampling:* Χρησιμοποιούνται τεχνικές εξισορρόπησης κλάσεων. Πιο συγκεκριμένα, εφαρμόστηκε η μέθοδος SMOTE στις κλάσεις tram και cyclist και under-sampling στην κλάση car. Μετά την εξισορρόπηση, οι παραπάνω 3 κλάσεις είχαν από 3500 instances η κάθε μία, ενώ, η κλάση pedestrian αφέθηκε ως έχει (2935 instances) για να μην προστεθεί θόρυβος στη συγκεκριμένη κατηγορία, διότι όπως θα φανεί και στα αποτελέσματα, η κλάση cyclist τείνει να ”μπερδεύεται” με την κλάση pedestrian. Οπότε είναι κομβικό να κρατήσουμε άρτια όλη την πληροφορία για την κλάση pedestrian εφόσον ο αριθμός των instances της είναι κοντά στο 3500.
   - *PCA:* Εφαρμόζεται *PCA* για μείωση της διαστατικότητας (ορίστηκε `n_components=300`).
   - *Εκπαίδευση Μοντέλου:* Στο τελικό βήμα, κάθε μοντέλο εκπαιδεύεται με τις υπό δοκιμή τιμές υπερπαραμέτρων.

3. **K-Fold Cross Validation ($k = 5$):** Εφαρμόζεται *5-Fold Cross Validation* στο *Training + Validation* σετ. Σε κάθε επανάληψη, ένα *fold* χρησιμοποιείται ως σύνολο επικύρωσης και τα υπόλοιπα 4 ως σύνολο εκπαίδευσης. Η διαδικασία αυτή επαναλαμβάνεται για όλες τις υποψήφιες ρυθμίσεις υπερπαραμέτρων.

4. **Μετρική Απόδοσης:** Ως κύρια μετρική ορίστηκε η *Macro F1-score*, η οποία δίνει ίσο βάρος σε κάθε κλάση. Με αυτόν τον τρόπο αξιολογούνται δίκαια και οι κλάσεις που έχουν λιγοστά δείγματα.

5. **Αξιολόγηση & Επιλογή Καλύτερης Τιμής Υπερπαραμέτρων:** Για κάθε συνδυασμό τιμών υπερπαραμέτρων, υπολογιζόταν η μέση τιμή (*mean*) της *Macro F1* σε όλα τα *folds*, καθώς και η τυπική απόκλιση (*std*). Επιλέγαμε ως βέλτιστη την παραμετροποίηση που παρουσίασε την υψηλότερη *mean* και ταυτόχρονα χαμηλή *std*.

Ολοκληρώνοντας την παραπάνω διαδικασία, καταλήξαμε στις βέλτιστες τιμές υπερπαραμέτρων για κάθε μοντέλο.

4.4 Τελική Εκπαίδευση και Αξιολόγηση των Μοντέλων

Σε αυτό το υποκεφάλαιο περιγράφεται η τελική διαδικασία εκπαίδευσης και αξιολόγησης των μοντέλων. Αξιοποιήθηκαν οι βέλτιστες υπερπαράμετροι όπως προέκυψαν από το *hyperparameter tuning*.

Διαδικασία Τελικής Εκπαίδευσης:

- **Ενοποίηση Συνόλων:** Όλοι οι ταξινομητές εκπαιδεύονται σε ολόκληρο το σύνολο *Training + Validation* ώστε να αξιοποιηθεί το μέγιστο πλήθος δειγμάτων.

- **Preprocessing (StandardScaler & PCA):**

  1. **StandardScaler:** Εφαρμόζεται σε όλα τα μοντέλα, ώστε τα χαρακτηριστικά να έχουν μέση τιμή 0 και τυπική απόκλιση 1.
  2. **PCA:** Μειώσαμε τη διάσταση των δεδομένων σε 300 συνιστώσες,

- **Διαχείριση Ανισορροπίας (SMOTE & Under-sampling):**

  – *SVM:* Για το μοντέλο *SVM* πραγματοποιήθηκε *SMOTE* (over-sampling) και *RandomUnderSampler* (under-sampling) κατά την τελική εκπαίδευση, καθώς οι αρχικές δοκιμές έδειξαν βελτίωση των δεικτών *Macro F1* κυρίως σε μειοψηφικές κλάσεις.

  – *KNN, GNB, Random Forests:* **Δεν** εφαρμόστηκαν *SMOTE* και *Under-sampling* στο τελικό στάδιο, αφού τα αποτελέσματα έδειξαν ότι χωρίς εξισορρόπηση ο *Macro F1* ήταν υψηλότερος.

- **Εκπαίδευση (`fit`):** Στο τελευταίο βήμα, καλείται η συνάρτηση εκπαίδευσης (`fit`) στο σύνολο *Train+Val*, με τις βέλτιστες υπερπαραμέτρους που βρέθηκαν στο προηγούμενο στάδιο.

Αξιολόγηση στο Testing Σετ: Μόλις ολοκληρωθεί η εκπαίδευση, κάθε μοντέλο αξιολογείται ανεξάρτητα στο *Testing* σετ (το οποίο δεν έχει χρησιμοποιηθεί καθόλου σε *training* ή *tuning*):

- **Πρόβλεψη (Predict):** Γίνεται πρόβλεψη των ετικετών στο *Testing* σετ.

- **Classification Report:** Εξάγονται βασικές μετρικές (*precision*, *recall*, *F1-score*) ανά κλάση.

- **Confusion Matrix:** Χρησιμοποιήθηκε για να αναδείξει τον αριθμό των true/false positives και true/false negatives.

- **Macro F1-score:** Χρησιμοποιείται ως συνολικός δείκτης απόδοσης για όλες τις κλάσεις.
- **ROC Curves & AUC:** Παρουσιάζονται οι *One-vs-Rest* ROC καμπύλες και υπολογίζεται η AUC (Area Under the Curve) για κάθε κλάση.

5 Πειραματικά Αποτελέσματα

Στο παρόν κεφάλαιο, παρουσιάζονται τα αποτελέσματα της αξιολόγησης των μοντέλων ταξινόμησης που εκπαιδεύτηκαν. Η ανάλυση περιλαμβάνει τις μετρικές επίδοσης των τεσσάρων ταξινομητών, καθώς και συγκριτική αξιολόγηση μεταξύ τους.
Το κεφάλαιο χωρίζεται στις ακόλουθες ενότητες:

- **Αποτελέσματα hyperparameter tuning:** Παρουσιάζονται οι βέλτιστες υπερπαράμετροι που προέκυψαν από τη διαδικασία βελτιστοποίησης, καθώς και η επίδρασή τους στην απόδοση των μοντέλων.
- **Μετρικές Σύγκρισης:** Γίνεται ανάλυση των βασικών μετρικών (*Precision, Recall, Macro F1-score*) για κάθε ταξινομητή, καθώς confusion matrices.
- **ROC Curves και AUC:** Εξετάζεται η ικανότητα των μοντέλων να διαχωρίζουν τις κλάσεις μέσω των *ROC* καμπυλών και του *Area Under the Curve (AUC)*.

Τα αποτελέσματα που παρουσιάζονται βασίζονται στο Testing Σετ, το οποίο δεν χρησιμοποιήθηκε κατά την εκπαίδευση των μοντέλων.

5.1 Hyperparameter tuning-Αποτελέσματα

Σε αυτήν την ενότητα παρουσιάζονται τα πειραματικά αποτελέσματα της διαδικασίας hyperparameter tuning για τους τέσσερις ταξινομητές που εξετάστηκαν στην παρούσα εργασία (SVM, KNN, GNB και Random Forests). Τα πειράματα διενεργήθηκαν με δύο διαφορετικές προσεγγίσεις:

1. **Με την εφαρμογή SMOTE και Undersampling:** Στην περίπτωση αυτή, εφαρμόστηκαν τεχνικές εξισορρόπησης (SMOTE για τις κλάσεις με λιγότερα δείγματα και undersampling για την πλειοψηφία) ώστε να επιτευχθεί ένα πιο ισορροπημένο training set.
2. **Χωρίς την εφαρμογή SMOTE και Undersampling:** Εδώ, τα δεδομένα παραμένουν στην αρχική τους (ανισορροπημένη) μορφή.

Παρακάτω παρουσιάζονται τα αποτελέσματα για κάθε μία από τις δύο προσεγγίσεις.

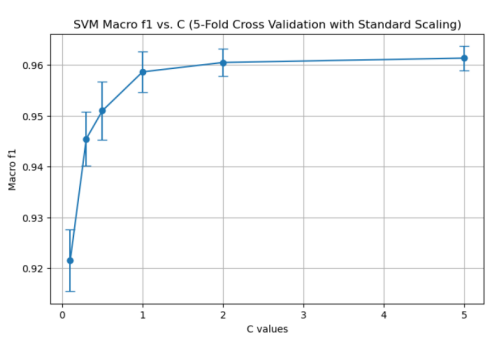
A. Αποτελέσματα με SMOTE και Undersampling

Πίνακες Αποτελεσμάτων: Οι παρακάτω πίνακες συνοψίζουν τις τιμές του μέσου Macro F1 score και της τυπικής απόκλισης (*std*) για κάθε δοκιμαζόμενη τιμή υπερπαραμέτρου, ανά μοντέλο.

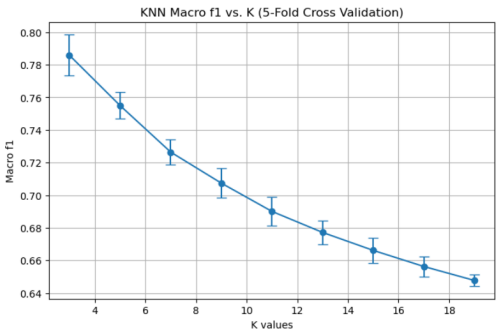
Table 3: Αποτελέσματα Hyperparameter Tuning με SMOTE και Undersampling

Μοντέλο	**Υπερπαράμετρος**	**Mean Macro F1**	**Std**
SVM	$C = 0.1$	0.922	0.000037
	$C = 0.3$	0.945	0.000028
	$C = 0.5$	0.951	0.000033
	$C = 1.0$	0.959	0.000016
	$C = 2.0$	0.960	0.000007
	$C = 5.0$	0.961	0.000006
KNN	$n\_neighbors = 3$	0.786	0.000159
	$n\_neighbors = 5$	0.755	0.000065
	$n\_neighbors = 7$	0.726	0.000058
	$n\_neighbors = 9$	0.707	0.000083
	$n\_neighbors = 11$	0.690	0.000078
	$n\_neighbors = 13$	0.677	0.000052
	$n\_neighbors = 15$	0.666	0.000061
	$n\_neighbors = 17$	0.656	0.000037
	$n\_neighbors = 19$	0.648	0.000012
GNB	`var_smoothing` $= 1 \times 10^{-5}$	0.580	0.001723
	`var_smoothing` $= 1 \times 10^{-6}$	0.567	0.001074
	`var_smoothing` $= 1 \times 10^{-7}$	0.574	0.002769
	`var_smoothing` $= 1 \times 10^{-8}$	0.580	0.002424
	`var_smoothing` $= 1 \times 10^{-9}$	0.573	0.002341
	`var_smoothing` $= 1 \times 10^{-10}$	0.581	0.002957
	`var_smoothing` $= 1 \times 10^{-11}$	0.578	0.001521
	`var_smoothing` $= 1 \times 10^{-12}$	0.579	0.002538
	`var_smoothing` $= 1 \times 10^{-13}$	0.579	0.001669
	`var_smoothing` $= 1 \times 10^{-14}$	0.585	0.002460
Random Forests	$n\_estimators = 100$	0.9751	0.000019
	$n\_estimators = 150$	0.9778	0.000014
	$n\_estimators = 200$	0.9794	0.000008
	$n\_estimators = 250$	0.9803	0.0000073
	$n\_estimators = 300$	0.9809	0.0000109

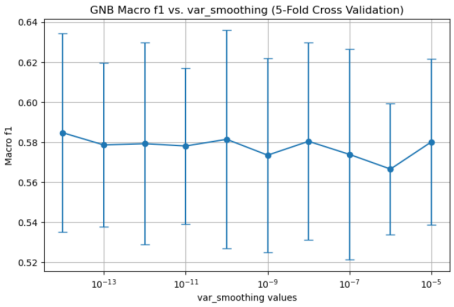
Γραφικές Παραστάσεις: Τα παρακάτω plots απεικονίζουν την πορεία του μέσου Macro F1 score σε σχέση με τις δοκιμαζόμενες τιμές υπερπαραμέτρων για κάθε μοντέλο, συνοδευόμενα από error bars που αντιπροσωπεύουν τη διακύμανση.



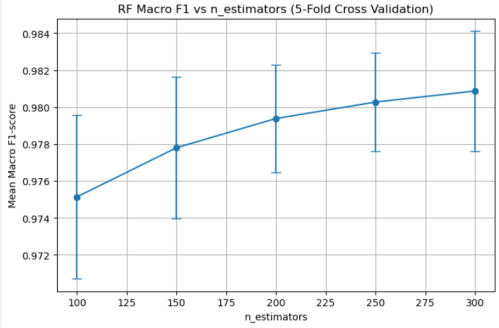
(a) SVM (Balanced)



(b) KNN (Balanced)



(c) GNB (Balanced)



(d) Random Forests (Balanced)

Figure 4: Αποτελέσματα Hyperparameter Tuning (Balanced): Macro F1 score σε σχέση με τις δοκιμαζόμενες τιμές υπερπαραμέτρων για τους 4 ταξινομητές.

Συμπεράσματα (με SMOTE/Undersampling): Από τη μελέτη των πειραματικών αποτελεσμάτων στον Πίνακα 3 αλλά και στα διαγράμματα του Σχήματος 4, μπορούν να εξαχθούν τα εξής συμπεράσματα:

- **SVM:** Η τιμή του μέσου Macro F1 αυξάνεται σταθερά καθώς το C μεγαλώνει (από 0.922 σε $C = 0.1$ έως και 0.961 σε $C = 5.0$). Η διαφορά μεταξύ $C = 2.0$ και $C = 5.0$ είναι πολύ μικρή (από 0.960 σε 0.961), υποδεικνύοντας ότι η περαιτέρω αύξηση του C μετά το 2.0 προσφέρει οριακή μόνο βελτίωση με κίνδυνο overfitting. Συνολικά, η απόδοση του SVM είναι εξαιρετικά υψηλή, με πολύ χαμηλή τυπική απόκλιση.

- **KNN:** Το καλύτερο σκορ επιτυγχάνεται για $n\_neighbors = 3$ (Macro F1 ≈ 0.786), ενώ όσο αυξάνεται ο αριθμός των γειτόνων (π.χ. μέχρι $n\_neighbors = 19$) η απόδοση πέφτει σταδιακά (0.648). Αυτό δείχνει ότι ο KNN αξιοποιεί καλύτερα τις τοπικές πληροφορίες σε μικρό αριθμό γειτόνων. Επίσης, η τυπική απόκλιση είναι αρκετά μικρή, υποδηλώνοντας σταθερή συμπεριφορά στα διαφορετικά folds.

- **Gaussian Naive Bayes (GNB):** Οι τιμές του Macro F1 κυμαίνονται περίπου από 0.567 έως 0.585, με σχετικά μεγαλύτερες τυπικές αποκλίσεις (π.χ. 0.0029) σε σύγκριση με άλλα μοντέλα. Η καλύτερη τιμή σημειώνεται στο $\texttt{var\_smoothing} = 10^{-14}$ (0.585), ωστόσο οι διακυμάνσεις δεν είναι αμελητέες. Γενικά, το GNB φαίνεται να μην μπορεί να προσφέρει με το balanced dataset.

- **Random Forests:** Η μέση τιμή του Macro F1 αυξάνεται βαθμιαία όσο μεγαλώνει ο αριθμός των δέντρων, από 0.9751 στα 100 trees έως 0.9809 στα 300 trees. Η βελτίωση είναι μεν υπαρκτή αλλά τείνει προς κορεσμό. Οι τυπικές αποκλίσεις παραμένουν πολύ χαμηλές, τονίζοντας την ιδιαίτερη σταθερότητα του Random Forest σε συνθήκες εξισορροπημένου δείγματος.

Γενικά, η εφαρμογή τεχνικών SMOTE/Undersampling βελτίωσε σημαντικά την ισορροπία των δεδομένων. Οι **SVM** και τα **Random Forests** εμφάνισαν τις καλύτερες επιδόσεις, ξεπερνώντας σε

Macro F1 το 0.96 και 0.98 αντίστοιχα, ενώ το **KNN** παρουσίασε αξιοσημείωτο εύρος επίδοσης ανάλογα με τον αριθμό γειτόνων. Παράλληλα, το **GNB** είχε συγκριτικά χαμηλότερο σκορ.

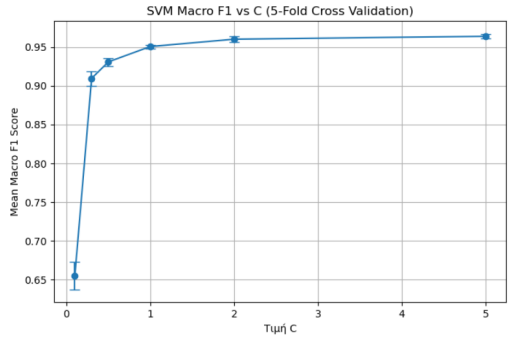
Β. Αποτελέσματα χωρίς SMOTE και Undersampling

Πίνακες Αποτελεσμάτων: Ο πίνακας 4 παρουσιάζει τις τιμές του μέσου Macro F1 score και της τυπικής απόκλισης για κάθε δοκιμαζόμενη τιμή υπερπαραμέτρου, όταν τα δεδομένα παραμένουν ανεξισορρόπητα.

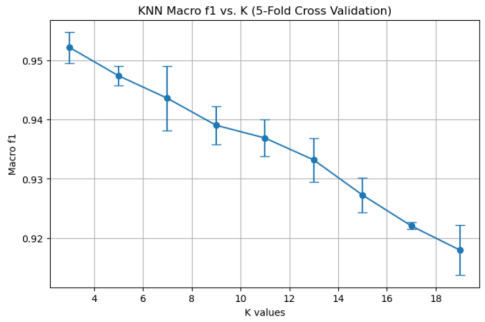
Table 4: Αποτελέσματα Hyperparameter Tuning χωρίς SMOTE και Undersampling

Μοντέλο	Υπερπαράμετρος	Mean Macro F1	Std
SVM	$C = 0.1$	0.654891	0.000310
	$C = 0.3$	0.909462	0.000090
	$C = 0.5$	0.930696	0.000027
	$C = 1.0$	0.950487	0.000006
	$C = 2.0$	0.959960	0.000014
	$C = 5.0$	0.963708	0.000010
KNN	$k = 3$	0.952	0.000007
	$k = 5$	0.947	0.000003
	$k = 7$	0.944	0.000029
	$k = 9$	0.939	0.000010
	$k = 11$	0.937	0.000010
	$k = 13$	0.933	0.000014
	$k = 15$	0.927	0.000008
	$k = 17$	0.922	0.000000
	$k = 19$	0.918	0.000018
GNB	$\texttt{var\_smoothing} = 10^{-5}$	0.881	0.000032
	$\texttt{var\_smoothing} = 10^{-6}$	0.883	0.000048
	$\texttt{var\_smoothing} = 10^{-7}$	0.881	0.000050
	$\texttt{var\_smoothing} = 10^{-8}$	0.882	0.000042
	$\texttt{var\_smoothing} = 10^{-9}$	0.884	0.000080
	$\texttt{var\_smoothing} = 10^{-10}$	0.881	0.000045
	$\texttt{var\_smoothing} = 10^{-11}$	0.883	0.000038
	$\texttt{var\_smoothing} = 10^{-12}$	0.881	0.000051
	$\texttt{var\_smoothing} = 10^{-13}$	0.883	0.000034
	$\texttt{var\_smoothing} = 10^{-14}$	0.882	0.000065
Random Forests	$n\_estimators = 100$	0.5484	0.0185
	$n\_estimators = 150$	0.5379	0.0119
	$n\_estimators = 200$	0.5433	0.0134
	$n\_estimators = 250$	0.5432	0.0155
	$n\_estimators = 300$	0.5447	0.0152

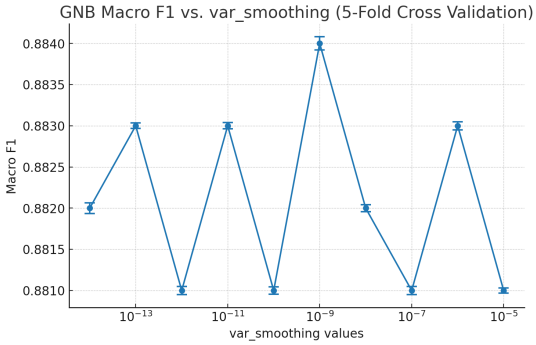
Γραφικές Παραστάσεις: Παρακάτω παρατίθενται τα plots που απεικονίζουν την πορεία του μέσου Macro F1 score σε σχέση με τις δοκιμαζόμενες τιμές υπερπαραμέτρων για κάθε μοντέλο, όταν δεν εφαρμόζεται εξισορρόπηση των δεδομένων.



(a) SVM (Unbalanced)



(b) KNN (Unbalanced)



(c) GNB (Unbalanced)

(d) Random Forests (Unbalanced)

Figure 5: Αποτελέσματα Hyperparameter Tuning (Unbalanced): Macro F1 score σε σχέση με τις δοκιμαζόμενες τιμές υπερπαραμέτρων για τους 4 ταξινομητές.

Συμπεράσματα (χωρίς SMOTE/Undersampling): Από τη μελέτη των πειραματικών αποτελεσμάτων στον Πίνακα 4 μπορούν να εξαχθούν τα εξής συμπεράσματα:

- **SVM:** Η τιμή του μέσου Macro F1 αυξάνεται σημαντικά όσο μεγαλώνει το C (από 0.6549 για $C = 0.1$ έως 0.9637 για $C = 5.0$). Ειδικά η μετάβαση από $C = 0.1$ σε $C = 0.3$ παρουσιάζει **μεγάλη άνοδο** (από ~ 0.65 σε ~ 0.91), υποδηλώνοντας ότι η χαμηλή τιμή C περιορίζει δραστικά το μοντέλο. Στη συνέχεια, η αύξηση του C οδηγεί σε πιο *βαθμιαία* βελτίωση. Οι τυπικές αποκλίσεις είναι ιδιαίτερα χαμηλές, δείχνοντας **σταθερή** απόδοση σε όλα τα folds. Συνολικά, το SVM καταγράφει *πολύ υψηλά* σκορ σε αυτό το ανισόρροπο περιβάλλον.

- **KNN:** Το καλύτερο σκορ (0.952) επιτυγχάνεται για $k = 3$, ενώ καθώς αυξάνεται ο αριθμός των γειτόνων. Το Macro F1 μειώνεται σταδιακά φτάνοντας το 0.918. Αυτό δείχνει ότι ο KNN παραμένει *αποτελεσματικός* σε μικρό αριθμό γειτόνων, αξιοποιώντας τοπική πληροφορία στο unbalanced σετ. Η τυπική απόκλιση είναι *πάρα πολύ χαμηλή* (κυρίως στην κλίμακα 10^{-5}), υποδηλώνοντας υψηλή **σταθερότητα** στα διαφορετικά δείγματα που δημιουργούνται με 5-Fold Cross Validation.

- **Gaussian Naive Bayes (GNB):** Το Macro F1 κυμαίνεται σε ένα σχετικά στενό εύρος (0.881–0.884), με τις διαφορές ανάμεσα στις διάφορες τιμές `var_smoothing` να είναι *πολύ μικρές*. Αυτό σημαίνει ότι ο GNB **δεν επηρεάζεται** έντονα από την εν λόγω υπερπαράμετρο στο συγκεκριμένο unbalanced σετ. Επίσης, η διακύμανση παραμένει χαμηλή, αντανακλώντας *ομοιόμορφη* απόδοση ανάμεσα στα folds.

- **Random Forests:** Το Mean Macro F1 κυμαίνεται από 0.5379 έως 0.5484 καθώς αυξάνεται ο αριθμός των δέντρων από 100 έως 300. Η *σχετικά περιορισμένη* αυτή διακύμανση υποδεικνύει ότι, σε ανισόρροπα δεδομένα, το RF δεν βελτιώνεται ιδιαίτερα με την αύξηση των `estimators`,

σε αντίθεση με όσα συνήθως συμβαίνουν σε πιο ισορροπημένα σετ. Επίσης, οι τυπικές αποκλίσεις (0.0119–0.0185) είναι μεγαλύτερες σε σύγκριση με τα υπόλοιπα μοντέλα, υποδηλώνοντας *μεγαλύτερη αστάθεια* στις διάφορες πτυχές του cross-validation.

Γενικά, οι **SVM** και **KNN** αποδεικνύονται *πολύ ανταγωνιστικά* ακόμα και χωρίς τη χρήση τεχνικών εξισορρόπησης, επιτυγχάνοντας Macro F1 άνω του 0.90. Αντίθετα, ο **Random Forest** δεν μπορεί να ξεπεράσει το 0.55 περίπου, ενώ ο **GNB** διατηρείται στα επίπεδα του 0.88, παρουσιάζοντας μια σχετικά σταθερή απόδοση.

Τελική Σύγκριση των Μοντέλων: Με ή Χωρίς Εξισορρόπηση Παρακάτω συνοψίζονται οι κυριότερες παρατηρήσεις *ανά μοντέλο*.

SVM: Με εξισορρόπηση, ο SVM αποφεύγει τις *χαμηλές* επιδόσεις σε μικρά C και ξεκινά ήδη από υψηλή τιμή (0.92). Χωρίς εξισορρόπηση, το SVM μπορεί να φτάσει παρόμοια τελικά σκορ (κοντά στο 0.96–0.97), αλλά εμφανίζει *πολύ χαμηλότερο* σημείο εκκίνησης. Σε γενικές γραμμές, η εξισορρόπηση κάνει το μοντέλο πιο *σταθερά* αποδοτικό και η αύξηση των instances για τις κλάσεις cyclist και tram βοηθά στην καλύτερη αντιμετώπιση τους μεμονωμένα.

KNN: Ο KNN δείχνει *αντίθετη* τάση συγκριτικά με τον SVM: αποδίδει *καλύτερα* χωρίς εξισορρόπηση, με Macro F1 έως 0.95+. Με SMOTE/Undersampling, το μέγιστο φτάνει μόλις στο 0.786. Ενδεχομένως η αλλαγή στην κατανομή των δειγμάτων (SMOTE/undersampling) διαφοροποιεί τις "γειτονιές" με τρόπο που δεν ωφελεί τον KNN προσθέτοντας θόρυβο.

Gaussian Naive Bayes (GNB): Σε *ανισόρροπη* κατανομή, το GNB αποδίδει καλύτερα (Macro F1 $\sim$ 0.88) σε σχέση με την περίπτωση SMOTE/Undersampling (Macro F1 $\sim$ 0.58). Η προσθήκη συνθετικών δεδομένων επηρεάζει αρνητικά τον τρόπο που το GNB εκτιμά τις πιθανότητες (*priors* και κατανομές), πιθανώς προσθέτοντας θόρυβο αντί σημαντική πληροφορία.

Random Forests: Από όλα τα μοντέλα, το Random Forest παρουσιάζει τη *μεγαλύτερη* διαφορά απόδοσης ανάμεσα σε balanced και unbalanced σετ (περίπου -0.43 μονάδες στο Macro F1). Φαίνεται ότι η εξισορρόπηση είναι **ζωτικής σημασίας** για το RF, ώστε να μπορέσει να μάθει ικανοποιητικά όλες τις κλάσεις. Επίσης, στο unbalanced dataset ο Random Forests classifier δείχνει να μην επηρεάζεται από την υπερπαράμετρο n_estimators, σε αντίθεση με το balanced dataset, όπου εκεί είχε ξεκάθαρη εξάρτηση.

Τελικη επιλογή Υπερπαραμέτρων: Στο πίνακα 5 φαίνονται οι τιμές που τελικά επιλέχθηκαν για κάθε μοντέλο. Στην τελική επιλογή έπαιξε ρόλο συνδυαστικά η τιμή του mean macro f1 με την αντίστοιχη τυπική απόκλιση.

Μοντέλο	Τελική Τιμή Υπερπαραμέτρου	Balanced
SVM	$C = 2.0$	ΝΑΙ
KNN	$n\_neighbors = 3$	ΟΧΙ
GNB	$var\_smoothing = 10^{-11}$	ΟΧΙ
RF	$n\_estimators = 300$	ΝΑΙ

Table 5: Τελικές τιμές των υπερπαραμέτρων για κάθε μοντέλο.

5.2 Αξιολόγηση Μοντέλων

Μετά την εύρεση των βέλτιστων υπερπαραμέτρων για κάθε μοντέλο, προχωρήσαμε σε **εκπαίδευση** στο train+val dataset και **αξιολόγηση** στο *test dataset*. Παρακάτω παρουσιάζονται τα τελικά *classification*

reports για καθένα από τα τέσσερα μοντέλα, τα αντίστοιχα *confusion matrices* και τα *ROC curves & AUC*.

5.2.1 Classification Reports

Στα παρακάτω tables, κάθε γραμμή αντιστοιχεί σε μία κλάση, ενώ δίνονται οι τιμές *precision, recall* και *F1-score* μαζί με το *support*. Επιπλέον, παρουσιάζονται οι μέσες τιμές **macro avg** και **weighted avg** στο τέλος του report.

Table 6: Classification Report – **SVM**

	Prec.	Rec.	F1	Sup.
Car	1.00	0.99	1.00	3890
Cyclist	0.94	0.91	0.92	241
Pedestrian	0.95	0.99	0.97	734
Tram	1.00	1.00	1.00	58
Accuracy	-	-	0.99	4923
Macro avg	0.97	0.97	0.97	4923
Weighted avg	0.99	0.99	0.99	4923

Table 7: Classification Report – **KNN**

	Prec.	Rec.	F1	Sup.
Car	1.00	0.99	1.00	3890
Cyclist	0.89	0.91	0.90	241
Pedestrian	0.96	0.97	0.96	734
Tram	0.97	1.00	0.98	58
Accuracy	-	-	0.99	4923
Macro avg	0.95	0.97	0.96	4923
Weighted avg	0.99	0.99	0.99	4923

Table 8: Classification Report – **GNB**

	Prec.	Rec.	F1	Sup.
Car	0.98	0.99	0.98	3890
Cyclist	0.80	0.70	0.75	241
Pedestrian	0.91	0.87	0.89	734
Tram	1.00	0.79	0.88	58
Accuracy	-	-	0.96	4923
Macro avg	0.92	0.84	0.88	4923
Weighted avg	0.96	0.96	0.96	4923

Table 9: Classification Report – **RF**

	Prec.	Rec.	F1	Sup.
Car	0.98	1.00	0.99	3890
Cyclist	0.94	0.68	0.79	241
Pedestrian	0.90	0.95	0.93	734
Tram	1.00	0.79	0.88	58
Accuracy	-	-	0.97	4923
Macro avg	0.96	0.86	0.90	4923
Weighted avg	0.97	0.97	0.97	4923

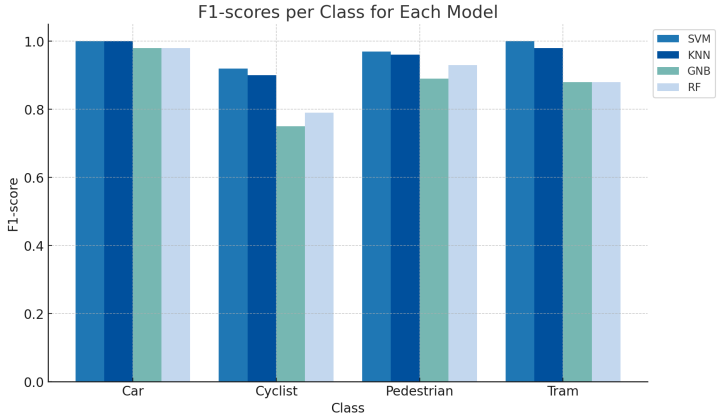


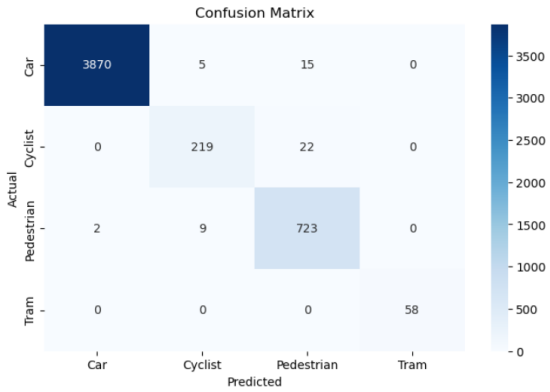
Figure 6: F1-scores ανά κλάση για κάθε μοντέλο. Οι SVM και KNN επιτυγχάνουν τα υψηλότερα σκορ σε όλες τις κατηγορίες, ενώ οι GNB και RF αποδίδουν ελαφρώς χειρότερα στις κατηγορίες Cyclist και Tram.

Με βάση τα Classification Reports των Πινάκων 6–9, προκύπτουν τα ακόλουθα γενικά συμπεράσματα για τα τέσσερα μοντέλα:

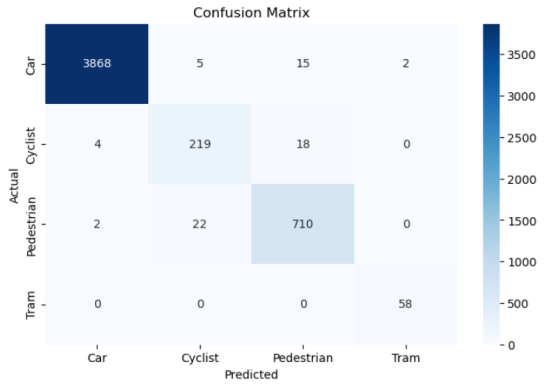
1. **Συνολική Ακρίβεια**
   - **SVM** και **KNN** πετυχαίνουν την υψηλότερη συνολική ακρίβεια (Accuracy = 0.99), διατηρώντας παράλληλα υψηλό *macro F1* (≈ 0.97 και 0.96 αντίστοιχα).
   - Το **Random Forest** ακολουθεί με 0.97 σε ακρίβεια και 0.90 σε macro F1.
   - Το **GNB** φτάνει σε Accuracy 0.96, με χαμηλότερο macro F1 (≈ 0.88).
2. **Επίδοση Ανά Κλάση**
   - *Car:* Όλα τα μοντέλα καταγράφουν σχεδόν τέλεια απόδοση (F1 κοντά στο 1.00), εξαιτίας του μεγάλου πλήθους δειγμάτων (3890).
   - *Cyclist:*
     - SVM/KNN: F1 κοντά στο 0.90, δείχνουν ότι οι «ποδηλάτες» αναγνωρίζονται με υψηλή ακρίβεια.
     - RF/GNB: Χαμηλότερα σκορ (F1 ≈ 0.79 ή 0.75), κυρίως λόγω μικρότερου Recall.
   - *Pedestrian:*
     - SVM/KNN: F1 $\approx$ 0.97–0.96, πολύ υψηλή επίδοση.
     - RF: F1 = 0.93, αρκετά καλό, αλλά υπολείπεται ελαφρώς.
     - GNB: F1 = 0.89, ελαφρώς χαμηλότερα.
   - *Tram:*
     - SVM/KNN: Φτάνουν σχεδόν το 1.00 σε F1-score.
     - RF/GNB: F1 ≈ 0.88, λόγω μικρότερου Recall (0.79).
3. **Διαφορά Macro vs. Weighted Avg**
   - Τα *weighted* averages διατηρούνται εξαιρετικά υψηλά ($0.97+$) στα περισσότερα μοντέλα, κυρίως λόγω της κυρίαρχης κλάσης «Car».
   - Τα *macro* averages έχουν μεγαλύτερη ευαισθησία στις «μικρές» κλάσεις (Cyclist, Tram). Έτσι, σε περιπτώσεις χαμηλού Recall σε αυτές τις κλάσεις, το macro F1 πέφτει αισθητά.
4. **Κύριες Παρατηρήσεις**
   - **SVM** & **KNN**: Εξαιρετικά αποτελέσματα με *σχεδόν τέλειες* προβλέψεις σε *Car* & *Tram*, και πολύ υψηλά F1 σε *Cyclist* & *Pedestrian*.
   - **Random Forest**: Παρά το πολύ καλό αποτέλεσμα σε *Car* & *Pedestrian*, εμφανίζει χαμηλότερη επίδοση σε *Cyclist*/*Tram*.
   - **Gaussian Naive Bayes**: Διατηρεί αρκετά καλούς δείκτες ειδικά στην κλάση *Car*, αλλά χάνει σε *Cyclist*/*Tram* (F1 ≈ 0.75 και 0.88 αντίστοιχα).

5.2.2 Confusion Matrices

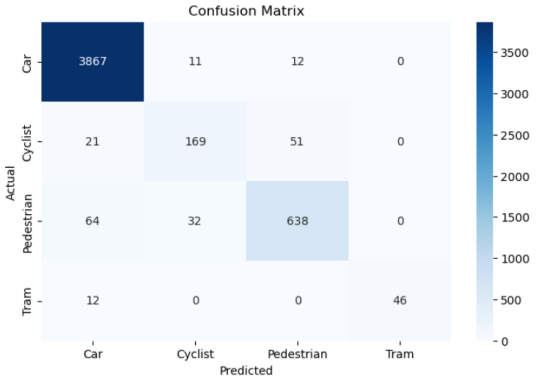
Στα επόμενα διαγράμματα φαίνονται τα *Confusion Matrices* των τεσσάρων μοντέλων, όπου στον άξονα των γραμμών (rows) έχουμε την actual κλάση και στον άξονα των στηλών (columns) την predicted.



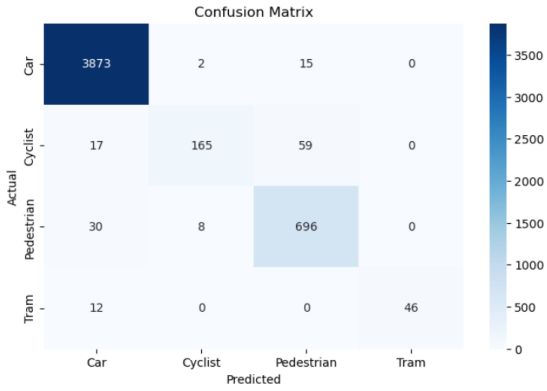
(a) Confusion Matrix – SVM (Test Set)



(b) Confusion Matrix – KNN (Test Set)



(c) Confusion Matrix – GNB (Test Set)



(d) Confusion Matrix – Random Forest (Test Set)

Figure 7: Confusion Matrices για όλα τα μοντέλα (Test Set)

Μελετώντας τα παραπάνω confusion matrices συμπαιρένονται τα εξής:

1. **SVM:**
   - *Car:* Αναγνωρίζεται σχεδόν άψογα (3870 σωστές προβλέψεις σε σύνολο 3890). Τα ελάχιστα λάθη προκύπτουν κυρίως όταν το `Car` προβλέπεται ως `Pedestrian` (15 περιπτώσεις).
   - *Cyclist:* Ελάχιστες λανθασμένες προβλέψεις ως `Car` (5) ή `Pedestrian` (2).
   - *Pedestrian:* Σχεδόν τέλεια αναγνώριση (723/734), με τις 9 λάθος προβλέψεις να είναι ως `cyclist`.
   - *Tram:* Όλα τα δείγματα (58) προβλέπονται σωστά.
2. **KNN:**
   - *Car:* 3868 περιπτώσεις ορθά, με μόλις 5 να διαγιγνώσκονται ως `Cyclist` και 15 ως `Pedestrian`.
   - *Cyclist:* 4 λανθασμένα ως `Car` και 18 ως `Pedestrian`, δείχνοντας μια *μικρή* σύγχυση μεταξύ αυτών των κλάσεων.
   - *Pedestrian:* 710 σωστές προβλέψεις (σε 734 δείγματα), 22 λανθασμένες κυρίως ως `Cyclist`.
   - *Tram:* Αναγνωρίζεται πλήρως (58/58).
3. **GNB:**
   - *Car:* 3867 περιπτώσεις αναγνωρίζονται σωστά, ενώ 12 μπερδεύονται με `Cyclist`.

- *Cyclist:* Πιο εμφανής σύγχυση, καθώς 21 δείγματα προβλέπονται ως `Car` και 51 ως `Pedestrian`.
- *Pedestrian:* 638 ορθές προβλέψεις (από 734), με λάθη κυρίως στην κατηγορία `Cyclist` (32) ή `Car` (64).
- *Tram:* Μόνο 46 σωστές αναγνωρίσεις, με 12 περιπτώσεις που κατηγοριοποιήθηκαν αλλού (κυρίως `Pedestrian`).

4. **Random Forests:**
   - *Car:* 3873 δείγματα σωστά, 2 λανθασμένα ως `Cyclist` και 15 ως `Pedestrian`.
   - *Cyclist:* 17 περιπτώσεις σαν `Car` και 59 σαν `Pedestrian`, υποδεικνύοντας μια σχετική δυσκολία.
   - *Pedestrian:* 696 ορθές προβλέψεις, 59 λανθασμένες σε `Cyclist`.
   - *Tram:* Από τα 58 δείγματα, 46 προβλέπονται σωστά, ενώ 12 εμφανίζονται ως `Pedestrian`.
5. **Κύρια συμπεράσματα:**
   - *SVM* και *KNN* εμφανίζουν τις λιγότερες συγχύσεις (misclassifications), ιδίως στις κλάσεις `Car` και `Tram`.
   - *GNB* και *Random Forest* παρουσιάζουν περισσότερα λάθη στις `Cyclist` και `Tram`, γεγονός που συνάδει και με τα χαμηλότερα F1-scores που είδαμε στα Classification Reports.
   - Συνολικά, η κατηγορία `Cyclist` μπερδεύεται με την `Pedestrian` αρκετές φορές, καθώς υπάρχουν πιθανά κοινά χαρακτηριστικά στα δεδομένα, γεγονός που αναδεικνύεται περισσότερο στους *GNB* και *RF*.

Όπως φαίνεται από τα Classification Reports και τα Confusion Matrices, μπορούμε να διακρίνουμε τις κλάσεις που παρουσιάζουν υψηλότερη *ανακλητικότητα (recall)* ή ακρίβεια (precision), καθώς και τυχόν παρανοήσεις (misclassifications) μεταξύ τους. Η γενική εικόνα επιτρέπει την *ποιοτική* και *ποσοτική* σύγκριση των μοντέλων στην πραγματική κατανομή του test set.

5.2.3 ROC curves και AUC

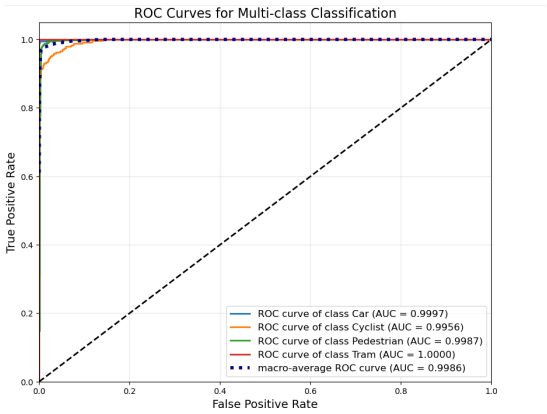
Στην αξιολόγηση μοντέλων ταξινόμησης, η Receiver Operating Characteristic (ROC) curve χρησιμοποιείται για να αναπαραστήσει τη σχέση μεταξύ του True Positive Rate (TPR) και του False Positive Rate (FPR) καθώς μεταβάλλεται το όριο απόφασης (threshold). Η Area Under the Curve (AUC) είναι μια αριθμητική μέτρηση της συνολικής απόδοσης ενός ταξινομητή. Όσο μεγαλύτερη είναι η τιμή AUC (με μέγιστο το 1.0), τόσο καλύτερη είναι η ικανότητα διάκρισης μεταξύ των κατηγοριών.

Υπολογισμός ROC και AUC - One-vs-Rest (OvR): Δεδομένου ότι το πρόβλημα είναι πολυκλασικό (multi-class classification), χρησιμοποιήθηκε η μέθοδος One-vs-Rest (OvR). Αυτό σημαίνει ότι για κάθε κλάση υπολογίζεται μια δυαδική ROC curve, όπου:

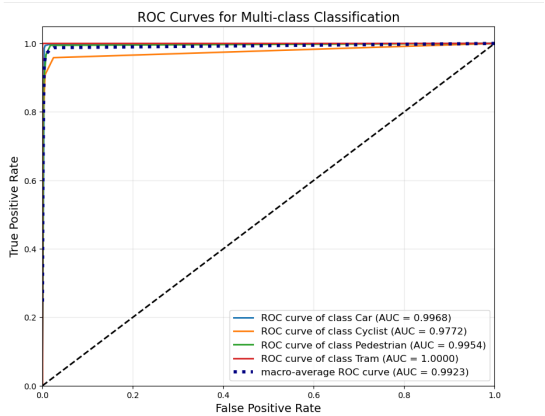
- Η συγκεκριμένη κλάση θεωρείται η θετική.
- Όλες οι υπόλοιπες κλάσεις θεωρούνται αρνητικές.

Με αυτόν τον τρόπο, μπορούμε να υπολογίσουμε μία ROC καμπύλη ανά κλάση και στη συνέχεια να βρούμε τις AUC τιμές.

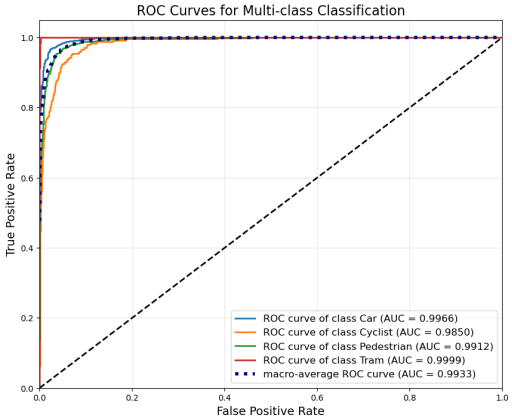
Γραφήματα ROC καμπυλών: Οι παρακάτω καμπύλες ROC παρουσιάζουν την απόδοση των μοντέλων **SVM, KNN, GNB και RF** για κάθε κλάση ξεχωριστά, καθώς και την **μέση ROC καμπύλη (macro-average ROC curve)**.



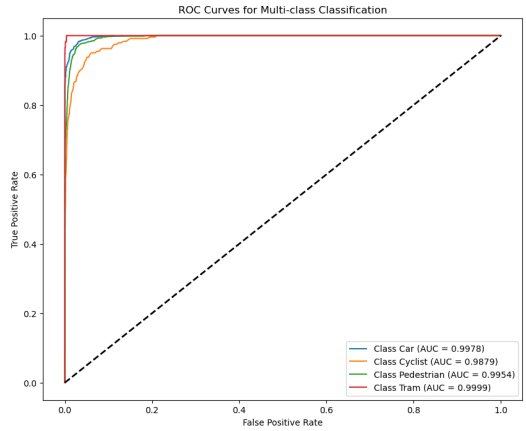
(a) Καμπύλη ROC για το SVM



(b) Καμπύλη ROC για το KNN



(c) Καμπύλη ROC για το GNB



(d) Καμπύλη ROC για το RF

Figure 8: Σύγκριση καμπυλών ROC για τα μοντέλα SVM, KNN, GNB και RF.

Συγκριτικός Πίνακας AUC Scores: Παρακάτω παρουσιάζεται ένας πίνακας με τις AUC τιμές για κάθε μοντέλο.

Μοντέλο	Car	Cyclist	Pedestrian	Tram
SVM	0.9997	0.9956	0.9987	1.0000
KNN	0.9968	0.9772	0.9954	1.0000
GNB	0.9966	0.9850	0.9912	0.9999
RF	0.9978	0.9879	0.9954	0.9999

Table 10: AUC Scores για κάθε κλάση ανά μοντέλο.

Συμπεράσματα: Από τις καμπύλες ROC και τις AUC τιμές, επιβεβαιώνεται και πάλι η καλύτερη συνολική απόδοση ανά κλάση των SVM και KNN αλλά δεν είναι ξεκάθαρη η διαφορά τους, καθώς τα ROC curves (και συνεπώς και τα AUC) υπολογίζονται με τη χρήση των TPR και FPR, ενώ το μεγαλύτερο πρόβλημα όπως φάνηκε και στα confusion matrices είναι το FNR ειδικά για την κλάση cyclist, το οποίο δεν εμφανίζεται στις ROC curves. Για το λόγο αυτό τα classification reports και τα confusion matrices είναι καλύτερες μετρικές για το πρόβλημα μας.

6 Συμπεράσματα

Στο πλαίσιο αυτής της εργασίας, αξιολογήθηκαν διάφοροι ταξινομητές (Support Vector Machine, k-Nearest Neighbors, Gaussian Naive Bayes και Random Forests) για την αναγνώριση αντικειμένων (Car, Pedestrian, Cyclist, Tram) στο KITTI dataset. Χρησιμοποιήθηκαν χαρακτηριστικά HOG έπειτα από crop στις εικόνες του KITTI, ενώ έγινε μελέτη για την επίδραση των τεχνικών εξισορρόπησης (*SMOTE* και *undersampling*) καθώς και της βελτιστοποίησης υπερπαραμέτρων. Από τα πειράματα προέκυψε ότι τα μοντέλα **SVM** και **KNN** πέτυχαν την υψηλότερη συνολική επίδοση. Επιπλέον, αναγνώρισαν με μεγάλη ακρίβεια και recall τις μικρότερες κατηγορίες (Cyclist, Tram), περιορίζοντας τις λανθασμένες ταξινομήσεις (FNR). Αντιθέτως, το **GNB** παρουσίασε δυσκολίες στην κατηγορία Cyclist, καταγράφοντας χαμηλότερο Macro F1, ενώ το **Random Forest** είχε σημαντικά περισσότερα σφάλματα στις κλάσεις με λίγα δείγματα, όπως οι Tram και Cyclist. Παρόλα αυτά, στις δημοφιλέστερες κατηγορίες (Car, Pedestrian) όλα τα μοντέλα έδειξαν εξαιρετική συμπεριφορά, με την κατηγορία Car να αναγνωρίζεται σχεδόν τέλεια, λόγω του μεγάλου όγκου δεδομένων που διέθετε. Η σύγχυση μεταξύ των κατηγοριών Cyclist και Pedestrian, που φάνηκε έντονα στα Confusion Matrices, οφείλεται πιθανώς στις ομοιότητες κάποιων χαρακτηριστικών (ανθρώπινη φιγούρα, συγκεκριμένες γωνίες λήψης). Επίσης, φάνηκε πως η εφαρμογή SMOTE και undersampling οδήγησε σε σημαντικά κέρδη για το Random Forest, αλλά δεν ωφέλησε εξίσου τον KNN και τον GNB, οι οποίοι μάλλον ωφελούνται από την αρχική κατανομή δεδομένων. Ο SVM δεν είχε σημαντικές διαφορές ανάμεσα στο αρχικό και στο ισοκατανεμημένο dataset. Συνολικά, επιβεβαιώθηκε ότι ο συνδυασμός HOG χαρακτηριστικών με αλγορίθμους όπως οι SVM και KNN μπορεί να δώσει υψηλή αξιοπιστία στην αναγνώριση αντικειμένων σε αυτόνομα οχήματα. Παράλληλα, όμως, η ποιότητα και η ποσότητα των δεδομένων φαίνεται καθοριστική, ειδικά για τις κλάσεις που διαθέτουν λίγα δείγματα.

7 Future Work

Μια πιθανή κατεύθυνση μελλοντικής εργασίας θα ήταν η αξιοποίηση επιπρόσθετης πληροφορίας βάθους (LiDAR data) ή χρονικών ακολουθιών εικόνων, ώστε να εμπλουτιστεί η αναπαράσταση και να βελτιωθεί ο διαχωρισμός ιδιαίτερα κοντινών κατηγοριών (π.χ. Pedestrian-Cyclist). Επίσης, θα ήταν ενδιαφέρον να εξεταστεί η ενσωμάτωση τεχνικών Deep Learning, όπως συνελικτικά νευρωνικά δίκτυα (CNNs), είτε για την εξαγωγή χαρακτηριστικών, είτε ως τελικοί ταξινομητές που θα αξιοποιήσουν πλήρως τη χωρική δομή των εικόνων. Τέλος, χρήσιμη θα ήταν και η διερεύνηση εναλλακτικών χαρακτηριστικών, όπως SIFT και LBP.

References

[1] Θεόδωρος Γιαννακόπουλος. *Σημειώσεις Μαθήματος "Μηχανική Μάθηση"*. ΔΠΜΣ Τεχνητή Νοημοσύνη. 2024-2025.

[2] Vikramaditya Jakkula. "Tutorial on support vector machine (svm)". In: *School of EECS, Washington State University* 37.2.5 (2006), p. 3.

[3] Gongde Guo et al. "KNN model-based approach in classification". In: *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings*. Springer. 2003, pp. 986–996.

[4] Carlos Bustamante, Leonardo Garrido, and Rogelio Soto. "Comparing fuzzy naive bayes and gaussian naive bayes for decision making in robocup 3d". In: *Mexican International Conference on Artificial Intelligence*. Springer. 2006, pp. 237–247.

[5] Adele Cutler, D Richard Cutler, and John R Stevens. "Random forests". In: *Ensemble machine learning: Methods and applications* (2012), pp. 157–175.

[6] Weida Gang. *Essential Math for Machine Learning: Gini Index and Entropy*. Retrieved February 11, 2025. 2024. URL: https://medium.com/@weidagang/essential-math-for-machine-learning-gini-index-and-entropy-d23ba9a90593.