Τσιατούρας Βαγγέλης, 1115201200185

Ιούνιος 10, 2018

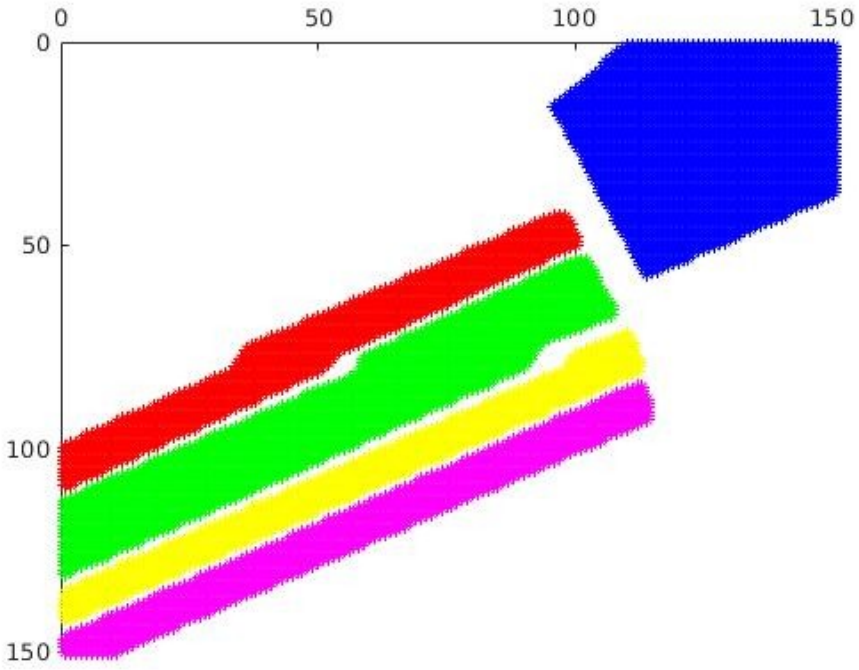
ΑΝΑΦΟΡΑ PROJECT ΑΝΑΓΝΩΡΙΣΗΣ ΠΡΟΤΥΠΩΝ & ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ

Εισαγωγικά

Το πρόγραμμα που έχει υλοποιηθεί καλύπτει πλήρως τα ζητούμενα της παρούσας εργασίας και δεν έχουν εντοπιστεί προβλήματα. Για την εκτέλεση του προγράμματος εκτελούμε το αρχείο project.m. Κατά την εκτέλεση ο χρήστης είναι υποχρεωμένος να επιλέξει ανάμεσα στους 3 ταξινομητές. Κάθε εκτέλεση ολοκληρώνεται αφού ολοκληρωθεί η ταξινόμηση από τον ταξινομητή που επιλέχθηκε αρχικά και για εκ νέου ταξινόμηση με άλλον ταξινομητή θα πρέπει να γίνει επανεκτέλεση του προγράμματος. Η ανάπτυξη πραγματοποιήθηκε σε Matlab R2018a σε λειτουργικό Linux Mint 18.3 (64bit) με υλικό CPU: i7 2600 @ 3.4 Ghz, RAM: 8GB DDR3.

Απεικόνιση Dataset

Το dataset το οποίο χρησιμοποιήθηκε για την υλοποίηση της εφαρμογής έχει την ακόλουθη μορφή:



Όπως φαίνεται στο παραπάνω διάγραμμα τα σύνολα έχουν χωριστεί σε 5 κατηγορίες. Με μπλε χρώμα παρουσιάζεται η 1η κατηγορία καλλιέργειας, με κόκκινο η 2η,

με πράσινο η 3η, με κίτρινο η 4η και τέλος με ροζ η 5η κατηγορία. Με βάση αυτά τα σύνολα οι κατηγοριοποιητές θα εκπαιδευτούν, θα τεσταριστούν και θα είναι σε θέση να προβλέψουν αν ένα εικονοστοιχείο ανήκει σε μια από τις 5 κατηγορίες καλλιεργειών. Όσον αφορά την επίδοση του κάθε κατηγοριοποιητή και λεπτομέρειες που παρατηρήθηκαν θα αναφερθούν στις αμέσως επόμενες ενότητες.

Κατηγοριοποιητής Naive Bayes

Ο κατηγοριοποιητής Naive Bayes είναι ο πιο βέλτιστος ταξινομητής από άποψης ακρίβειας ταξινόμησης και από άποψη ταχύτητας εκτέλεσης. Πιο συγκεκριμένα όπως φαίνεται στη συνέχεια μέσω των μητρώων συγχύσεως των test και operational dataset πηγάζει εύκολα το συμπέρασμα ότι πρόκειται για έναν ταξινομητή υψηλής ακρίβειας. Να σημειωθεί ότι για την υλοποίηση του ταξινομητή έγινε η χρήση του παρακάτω τύπου:

$$\hat{y} = \underset{k \in \{1,\dots,|C|\}}{argmax}\, p(C_k|x_1,\dots,x_n) = \underset{k \in \{1,\dots,|C|\}}{argmax}\, p(C_k)\prod_{i=1}^{n} p(x_i|C_k)$$

Ο παραπάνω τύπος όμως παρουσιάζει ένα σημαντικό πρόβλημα. Λόγω της υψηλής διάστασης των διανυσμάτων (204 διαστάσεις) παρουσιάζεται υψηλή ανακρίβεια στην εκτίμηση αποτελεσμάτων καθώς οι πυκνότητες αρκετές φορές έχουν πολύ μικρές τιμές (πάνω από 4 μηδενικά δεκαδικά ψηφία). Επομένως το γινόμενο αυτό τείνει πολλές φορές προς το 0 με αποτέλεσμα να γίνεται λάθος ταξινόμηση. Το πρόβλημα αυτό μπορεί να αντιμετωπιστεί εύκολα αν απλά λογαριθμιστεί ο παραπάνω τύπος.

$$\hat{y} = \underset{k \in \{1,\dots,|C|\}}{argmax} \log\left(p(C_k|x_1,\dots,x_n)\right)$$

$$\hat{y} = \underset{k \in \{1,\dots,|C|\}}{argmax} \log\left(p(C_k)\prod_{i=1}^{n} p(x_i|C_k)\right)$$

$$\hat{y} = \underset{k \in \{1,\dots,|C|\}}{argmax}\left(\log\left(p(C_k)\right) + \sum_{i=1}^{n} \log\left(p(x_i|C_k)\right)\right)$$

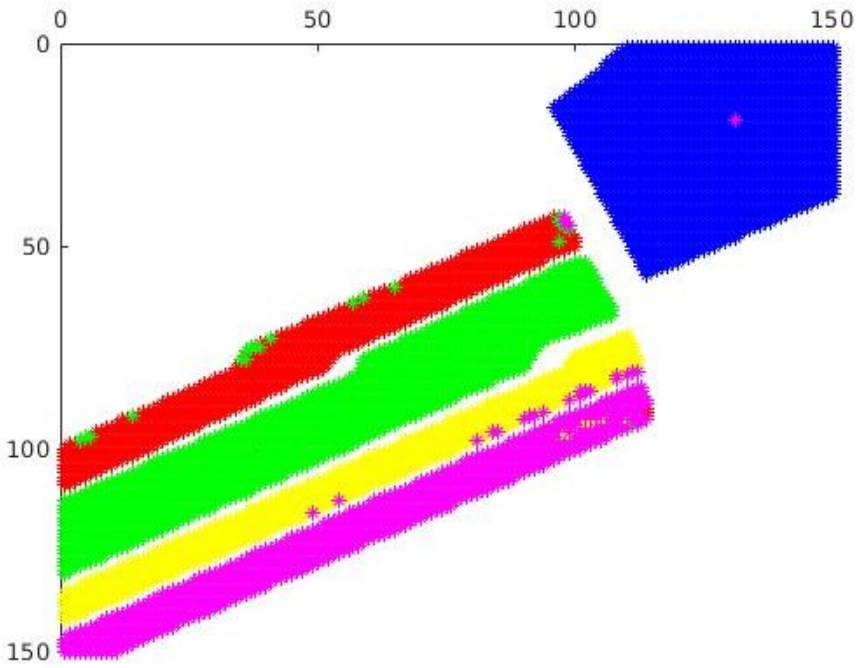
Έτσι με τη χρήση του νέου τύπου ο ταξινομητής επιτυγχάνει αξιόλογα αποτελέσματα.

Ακολουθούν οι πίνακες συγχύσεως καθώς και η οπτικοποίηση του συνόλου δεδομένων μετά την ταξινόμηση.

```
##########################
        TEST SET
##########################
Accuracy: 98.43%
Errors: 34
   652     0     0     0     0
     0   333     8     0     1
     0     1   548     0     0
     0     0     0   267     8
     0     2     0    14   331

##########################
      OPERATIONAL SET
##########################
Accuracy: 98.23%
Errors: 64
  1156     0     0     0     1
     0   521    11     0     2
     0     8   950     0     0
     0     0     0   440    11
     0     4     3    24   478

Classification Execution Time: 0.911583 seconds
```



Αξίζει να σημειωθεί ότι ο χρόνος εκτέλεσης είναι ο χαμηλότερος από τους 3 ταξινομητές.

ΕΡΩΤΗΜΑ C

Για την υλοποίηση ενός πολυδιάστατου κατηγοριοποιητή τύπου Bayes θα πρέπει να γίνει η χρήση του παρακάτω τύπου:

$$p(x)=\frac{1}{2\,\pi^{l/2}|\Sigma|^{1/2}}\exp\left(\frac{1}{2}(x-\mu)^T\,\Sigma^{-1}(x-\mu)\right)$$

Συγκεκριμένα για να εξαχθεί η παραπάνω πιθανότητα θα πρέπει να υπολογιστεί ο πίνακας Σ (covariance matrix) ο οποίος θα είναι 204x204 και θα είναι της μορφής:

$$\Sigma=\begin{vmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1204} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2204} \\ \vdots & \vdots & \sigma_i^2 & \vdots \\ \sigma_{2041} & \sigma_{2042} & \cdots & \sigma_{204}^2 \end{vmatrix}$$

Ουσιαστικά ο αλγόριθμος θα αποτελείται συνολικά από 5 κατανομές, όσες και οι κλάσεις, όπου κάθε μία θα παράγεται από τους παραπάνω τύπους (σε αντίθεση με τον Naive Bayes όπου υλοποιούμε 204x5 μονοδιάστατες normal pdfs). Αρχικά ο αλγόριθμος αυτός για να εκπαιδευτεί θα απαιτεί ικανό σύνολο Ν ώστε να εκτιμήσει όσο το δυνατόν πιο σωστά τους πίνακες Σ και στη συνέχεια τη κατανομή στην οποία ανήκουν τα στοιχεία της κάθε κατηγορίας. Στην περίπτωση όπου ο Σ είναι διαγώνιος τότε τα attributes είναι στατιστικά ανεξάρτητα μεταξύ τους και επομένως η κατανομή που θα υλοποιηθεί θα είναι η βέλτιστη. Για να επιτευχθεί αυτό στο συγκεκριμένο μοντέλο των 204 διαστάσεων (αρκετά μεγάλης διάστασης πρόβλημα) το σύνολο εκπαίδευσης όπως προαναφέρθηκε θα πρέπει να είναι αρκετά μεγάλο και συγκεκριμένα της τάξης N^{204}, αν Ν είναι το βέλτιστο πλήθος σημείων εκπαίδευσης μιας μονοδιάστατης κανονικής κατανομής. Αυτό μπορεί να συγκριθεί με τον ήδη υλοποιημένο ταξινομητή Naive Bayes ο οποίος χρησιμοποιεί μονοδιάστατες κατανομές και έχει ένα βέλτιστο Train Set του πλήθους 1444 σημείων με το οποίο επιτυγχάνεται ακρίβεια 98%. Θεωρητικά για να επιτευχθεί το ίδιο στον πολυδιάστατο ταξινομητή Bayes θα απαιτηθεί ένα σύνολο εκπαίδευσης με πλήθος σημείων 1444^{204}, στην καλύτερη περίπτωση, το οποίο είναι ένα εξωφρενικά μεγάλο ποσό. Επιπρόσθετα, έστω ότι με κάποιον τρόπο υπήρχε διαθέσιμο ένα τέτοιο σύνολο, η υπολογιστική ισχύς που θα απαιτούνταν για την εφαρμογή αυτού του αλγορίθμου θα ήταν εκθετικά πολύ μεγαλύτερη από την ισχύ που απαιτείται τώρα έτσι ώστε να επιτευχθούν αντίστοιχοι χρόνοι εκτέλεσης. Επομένως η εφαρμογή ενός τέτοιου μοντέλου θα ήταν μια αστοχία.

Κατηγοριοποιητής Ελάχιστης Ευκλείδειας απόστασης

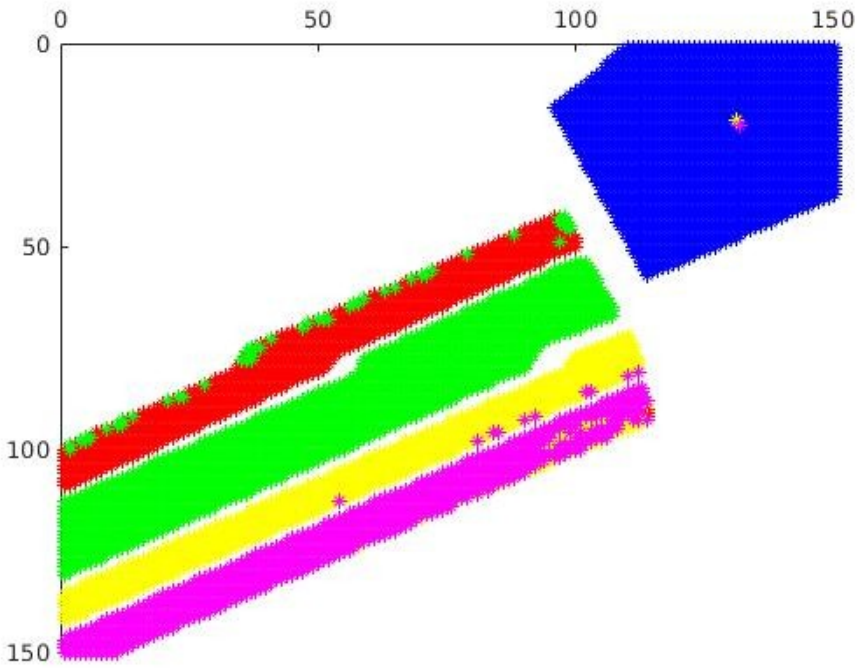
Γενικά ο κατηγοριοποιητής ελάχιστης ευκλείδειας απόστασης πρόσφερε πολύ ικανοποιητικά αποτελέσματα και από άποψης ακρίβειας ταξινόμησης και από άποψης ταχύτητας εκτέλεσης καθώς η πολυπλοκότητα του είναι πολύ πιο χαμηλότερη σε σχέση με του ταξινομητή k-NN που θα αναλυθεί παρακάτω. Παρόλα αυτά ο ταξινομητής αυτός είναι κατώτερος του Naive Bayes και από ακρίβεια και από χρόνο εκτέλεσης (κατά 1 δευτερόλεπτο πιο αργός). Όσον αφορά την υλοποίηση του, για την απόφαση αν κάποιο στοιχείο ανήκει σε μια κατηγορία αρκεί να υπολογιστεί η ευκλείδεια απόσταση του σημείου από το μέσο της εκάστοτε κατηγορίας και η μικρότερη απόσταση από τις 5 είναι η προτιμότερη. Δηλαδή εφαρμόστηκε ο παρακάτω τύπος.

$$\hat{y} = \underset{k \in \{1,\ldots,|C|\}}{argmin} \left(d = \|x - \mu_i\| \right)$$

Ακολουθούν διαγράμματα και στατιστικά εκτέλεσης του ταξινομητή.

```
##########################
        TEST SET
##########################
Accuracy: 97.55%
Errors: 53
   652     0     0     0     0
     0   321    21     0     0
     0     0   549     0     0
     0     0     0   270     5
     0     2     0    25   320
##########################
      OPERATIONAL SET
##########################
Accuracy: 97.20%
Errors: 101
  1155     0     0     1     1
     0   505    29     0     0
     0     0   958     0     0
     0     0     0   446     5
     0     4     0    61   444

Classification Execution Time: 2.508502 seconds
```



Κατηγοριοποιητής k-πλησιέστερων Γειτόνων

Ο κατηγοριοποιητής k-πλησιέστερων γειτόνων αποδείχθηκε ο πιο ακριβής στην ταξινόμηση των εικονοστοιχείων καθώς η ακρίβεια που επιτυγχάνει αγγίζει το 99% σωστών προβλέψεων. Βέβαια ο αλγόριθμος αυτός πάσχει από ένα σημαντικό πρόβλημα το οποίο είναι το λεγομένο “Curse of Dimensionality” καθώς κάθε εικονοστοιχείο απαρτίζεται από 204 attributes με αποτέλεσμα να αυξάνεται εκθετικά η πολυπλοκότητα του σε μεγάλα datasets. Αξίζει να αναφερθεί ότι για την σωστή επιλογή του k έγινε εφαρμογή του cross validation concept. Πιο συγκεκριμένα το Train set διαχωρίστηκε σε 2 κομμάτια, το ένα το οποίο αποτελείται από το 80% του αρχικού το οποίο θα είναι train set και το υπόλοιπο 20% που είναι το validation set. Το cross validation επαναλαμβάνεται 5 φορές (5 fold cross validation) και σε κάθε γύρο τα set είναι διαφορετικά διατηρώντας τα πλήθη των στοιχείων από κάθε κατηγορία ίσα, έτσι ώστε κάθε validation set να έχει το 1/5 των στοιχείων από κάθε κατηγορία. Με αυτό τον τρόπο η ταξινόμηση κατά το cross validation δίνει αποτελέσματα τα οποία είναι αρκετά κοντά στα αποτελέσματα που θα δώσει μια κανονική εκτέλεση και επίσης λόγω της ταχύτερης εκτέλεσης του cross validation είναι πιο εύκολο (γρήγορο) να γίνουν δοκιμές με διαφορετικά k. Ακολουθούν στατιστικά από εκτελέσεις του 5 fold cross validation για διαφορετικά k.

```
Average Accuracy: 99.04% with k=1
Average Accuracy: 98.97% with k=3
Average Accuracy: 98.91% with k=5
Average Accuracy: 98.70% with k=7
Average Accuracy: 98.70% with k=9
Average Accuracy: 98.22% with k=11
Average Accuracy: 97.81% with k=13
Average Accuracy: 97.95% with k=15
Average Accuracy: 97.82% with k=17
```

Αυτό που παρατηρείται είναι ο ταξινομητής k-NN αποδίδει καλύτερα για μικρότερα k. Αυτό οφείλεται στο ότι τα dataset (Train/Test/Operational) μοιάζουν αρκετά μεταξύ τους καθώς φαίνεται να περιέχουν και κοινά στοιχεία. Επομένως είναι απολύτως λογικό να αποδίδει καλύτερα για μικρότερο k. Όμως για ένα διαφορετικό dataset για k=1 ο ταξινομητής δεν θα απέδιδε τόσο καλά με αποτέλεσμα οι προβλέψεις που να δίνει να βασίζονται σε ένα μοντέλο το οποίο πάσχει από overfitting και έτσι να υπάρχουν αρκετά λανθασμένα αποτελέσματα. Για το παρόν πρόβλημα προτείνεται να επιλεχθεί κάποιο k ανάμεσα στο 3 και στο 7 καθώς και υπάρχει και ένα ικανό πλήθος γειτόνων και υπάρχει και υψηλή ακρίβεια αποτελεσμάτων. Οι εκτελέσεις που ακολουθούν έγιναν για k=3.

5-fold cross validation

```
Fold 1
Accuracy: 99.31%
Errors: 2
    91     0     0     0     0
     0    38     0     0     0
     0     0    84     0     0
     0     0     0    37     1
     0     0     0     1    37
```

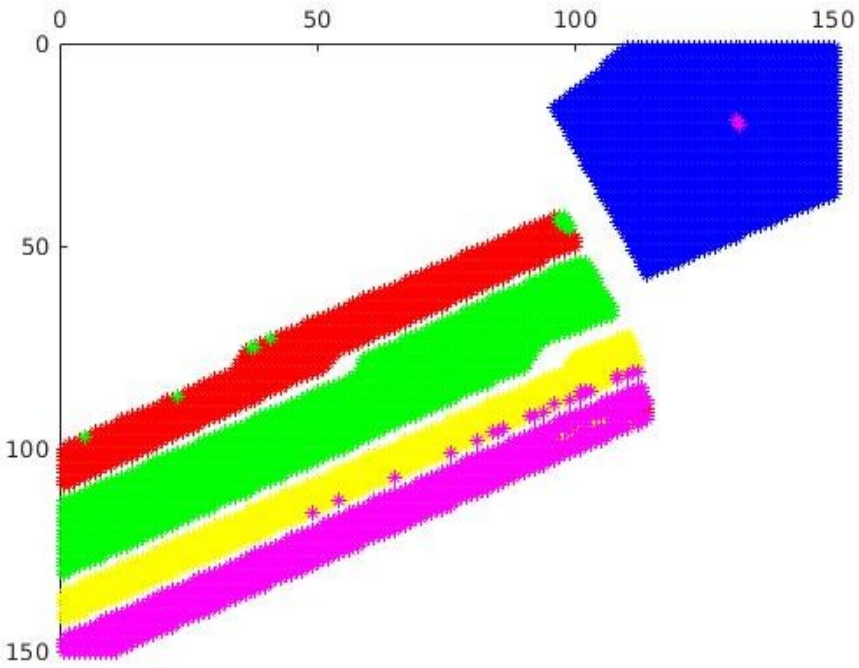
```
Fold 2
Accuracy: 99.32%
Errors: 2
    92     0     0     0     0
     0    38     2     0     0
     0     0    85     0     0
     0     0     0    39     0
     0     0     0     0    38
Fold 3
Accuracy: 98.98%
Errors: 3
    91     0     0     0     0
     0    38     1     0     0
     0     0    85     0     0
     0     0     0    38     1
     0     0     0     1    38
Fold 4
Accuracy: 97.28%
Errors: 8
    91     0     0     0     1
     0    36     4     0     0
     0     0    85     0     0
     0     0     0    39     0
     0     1     0     2    35
Fold 5
Accuracy: 99.32%
Errors: 2
    92     0     0     0     0
     0    39     0     0     0
     0     0    85     0     0
     0     0     0    38     1
     0     0     0     1    38
End of 5-fold cross validation
Average Accuracy: 98.84% with k=3
```

Normal Execution

```
##########################
        TEST SET
##########################
Accuracy: 98.66%
Errors: 29
   652     0     0     0     0
     0   334     8     0     0
     0     1   548     0     0
     0     0     0   266     9
     0     1     0    10   336


##########################
      OPERATIONAL SET
##########################
Accuracy: 98.75%
Errors: 45
  1155     0     0     0     2
     0   530     4     0     0
     0     0   958     0     0
     0     0     0   438    13
     0     4     0    22   483

Classification Execution Time (cross validation execution time is
excluded): 625.136409 seconds
```



Όπως φαίνεται ο ταξινομητής είναι εξαιρετικός και με μικρές αστοχίες. Όμως ο χρόνος εκτέλεσης του είναι αρκετά μεγαλύτερος σε σχέση με τον ταξινομητή ευκλείδειας απόστασης και τον Naive Bayes. Επομένως ίσως η ιδανικότερη λύση για τα παρόντα dataset είναι η χρήση του ταξινομητή Naive Bayes καθώς προσφέρει και υψηλή ταχύτητα εκτέλεσης και υψηλό ποσοστό ακρίβειας.

Ο κώδικας της εργασίας βρίσκεται στον προσωπικό μου λογαριασμό στο Github.