 ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

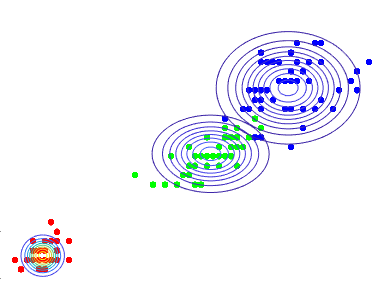
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

Αναγνώριση Προτύπων – Εργασία 1η

***Εξέταση της απόδοσης ταξινομητών (kNN, Bayes, Naive Bayes) και χρήση δικτύων Bayes για συμπερασμό πιθανοτήτων***



*Ον/μο: Δοξαστάκης Γεώργιος*

*Α.Μ.: Μ1613*

*Ον/μο: Μέτσης Γεώργιος*

*Α.Μ.: Μ16--*

Περιεχόμενα

[1. Εισαγωγή 3](#_Toc503701052)

[2. Δεδομένα ΚΑι Μέθοδοι 5](#_Toc503701053)

[2.1 Ταξινομητές Bayes 5](#_Toc503701054)

[2.2 Ταξινομητές KNN 5](#_Toc503701055)

[2.3 Δίκτυα Bayes 7](#_Toc503701056)

[2.4 Σύνολα Δεδομένων 13](#_Toc503701057)

[2.4.1 Fisher Iris 13](#_Toc503701058)

[3. Πειραματικά Αποτελέσματα 16](#_Toc503701059)

[6. Συμπεράσματα 19](#_Toc503701060)

[Βιβλιογραφία 20](#_Toc503701061)

[Παράρτημα A. Κώδικας 21](#_Toc503701062)

[1. KNN Classifier 21](#_Toc503701063)

[2. Bayes Classifier 22](#_Toc503701064)

[3. Naïve Bayes Classifier 28](#_Toc503701065)

[4. Bayesian Net 29](#_Toc503701066)

[Εικόνα 1: Διαθέσιμες εμπορικές λύσεις EthoVisionXT (πάνω) και MedAssociates TST (κάτω) 4](#_Toc503023757)

[Εικόνα 2: IntelliJ IDEA 6](#_Toc503023758)

[Εικόνα 3: JavaFX 6](#_Toc503023759)

[Εικόνα 6: Λειτουργικό διάγραμμα εφαρμογής 7](#_Toc503023760)

[Εικόνα 7: Περιβάλλον χρήστη 8](#_Toc503023761)

[Εικόνα 11: Επιλογή περιοχής ανάλυσης 10](#_Toc503023762)

[Εικόνα 12: Ένδειξη προόδου ανάλυσης 11](#_Toc503023763)

[Εικόνα 13: Διαγράμματα αποτελεσμάτων 12](#_Toc503023764)

[Εικόνα 14:. Οπτικοποίηση σκόρ κατά την αναπαραγωγή 13](#_Toc503023765)

[Εικόνα 11: Ανίχνευση οπτικής ροής μέσω του αλγορίθμου Farneback 14](#_Toc503023766)

[Εικόνα 12: Φιλτράρισμα διανυσμάτων (πράσινο: αποδεκτά, κόκκινο: απορριπτέα) 15](#_Toc503023767)

[Εικόνα 13: Δεδομένα κίνησης 15](#_Toc503023768)

[Εικόνα 14: Απόδοση αλγορίθμου συναρτήσει των παραμέτρων 16](#_Toc503023769)

[Εικόνα 15: Σύγκριση πραγματικού (πορτοκαλί) με προβλεπόμενο (μπλέ) σκορ 17](#_Toc503023770)

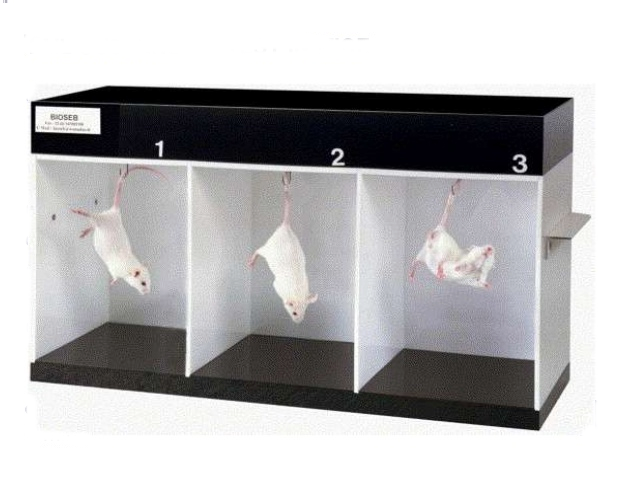
[Εικόνα 17: Διάγραμμα αποτελεσμάτων αυτοματοποιημένου συστήματος (VL) σε σύγκριση με των βαθμολογητών (NK,CD) 18](#_Toc503023771)

# Εισαγωγή

Κατά την διαλογή νέων φαρμάκων χρησιμοποιούνται διάφορα πειραματόζωα τα οποία είναι στην πλειοψηφία τρωκτικά στα οποία πραγματοποιούνται δοκιμές και αξιολογείται η πιθανή επίδραση των φαρμάκων. Για την μελέτη πιθανών αντικαταθλιπτικών ουσιών πραγματοποιούνται πειράματα σε μικρά τρωκτικά τα οποία θέτονται σε κατάσταση αυξημένου άγχους και παρακολουθείται η διαφορά στις συμπεριφορές που παρουσιάζει ένα πειραματικό group σε σχέση με ένα group αναφοράς. Μια τέτοια μέθοδος είναι το Τail Suspension Test (TST) κατά το οποίο τα πειραματόζωα αναρτώνται μέσω της ουράς σε σημείο από το οποίο δεν μπορούν να φτάσουν κοντινές επιφάνειες για τυπικό διάστημα έξι λεπτών (Can et al., 2011). Κατά την διάρκεια του πειράματος το τρωκτικό εμφανίζει προσπάθειες διαφυγής κουνώντας το σώμα του και σε κάποια διαστήματα οι προσπάθειες παύουν. Ο συνολικός χρόνος κίνησης του πειραματόζωου μεταβάλλεται ανάλογα με την αποτελεσματικότητα των υποψήφιων αντικαταθλιπτικών (Lad et al., 2007).

Για τον υπολογισμό του συνολικού χρόνου κίνησης ένας ερευνητής παρακολουθεί το βίντεο του κάθε πειράματος και βαθμολογεί την κατάσταση κίνησης του πειραματόζωου για κάθε χρονική στιγμή (Can et al., 2011). Η διαδικασία αυτή της βαθμολόγησης πρέπει να πραγματοποιείται με ακρίβεια και από πολλαπλούς βαθμολογητές για κάθε ένα από τα δεκάδες πειραματόζωα του κάθε group. Για την καταγραφή των καταστάσεων κίνησης χρησιμοποιούνται χρονόμετρα ή λογισμικά καταγραφής καταστάσεων μέσω πληκτρολογίου τα οποία δεν ενσωματώνουν την αναπαραγωγή του βίντεο καθιστώντας την διαδικασία δύσκολη στην τυποποίηση και χρονοβόρα.

Για την λύση του προβλήματος βαθμολόγησης υπάρχουν διαθέσιμες στο εμπόριο ολοκληρωμένες λύσεις διεξαγωγής του πειράματος όπως η διάταξη TST της εταιρίας Med Associartes Ιnc. η οποία παρέχει θαλάμους ανάρτησης με ενσωματωμένους αισθητήρες βάρους οι οποίοι με την βοήθεια του συνοδευτικού λογισμικού ανιχνεύουν την κίνηση του πειραματόζωου (Juszczak et al., 2006). Μια άλλη εμπορική λύση είναι το λογισμικό Ethovision XT της εταιρίας Noldus το οποίο πραγματοποιεί συμπεριφορική ανάλυση πολλών τύπων πειραμάτων με χρήση υπολογιστικής όρασης και διατίθεται με τιμή εκκίνησης 5.850 $.



Εικόνα 1: Διαθέσιμες εμπορικές λύσεις EthoVisionXT (πάνω) και MedAssociates TST (κάτω)

Η παρούσα εργασία έχει ως σκοπό την σχεδίαση ενός αυτοματοποιημένου συστήματος βαθμολόγησης πειραμάτων TST μέσω των βίντεο καταγραφής τους το οποίο θα χρησιμοποιεί αλγορίθμους υπολογιστικής όρασης για την εξαγωγή των δεδομένων κίνησης και θα παρέχει στον χρήστη ένα πλήρες περιβάλλον ανάλυσης πειραμάτων. Η χρήση ενός τέτοιου συστήματος απαλλάσσει τον χρήστη από την διαδικασία παρακολούθησης των βίντεο με αποτέλεσμα να μπορεί να χρησιμοποιηθεί μεγαλύτερος αριθμός πειραματόζωων σε κάθε μελέτη.

Παρόμοιες μελέτες έχουν πραγματοποιηθεί για την εξέταση της καταλληλόλητας των διαθέσιμων εμπορικών λύσεων βαθμολόγησης πειραμάτων. Από τα πειράματα TST που πραγματοποιήθηκαν με το σύστημα MedAssociates και το λογισμικό EthoVision XT και συγκρίθηκαν με πραγματικούς βαθμολογητές φάνηκε ότι τα αυτοματοποιημένα συστήματα παρουσιάζουν απόκλιση μερικών μόνο δευτερολέπτων και είναι κατάλληλα για την αξιόπιστη βαθμολόγηση του πειράματος (Juszczak et al., 2006).

# Δεδομένα ΚΑι Μέθοδοι

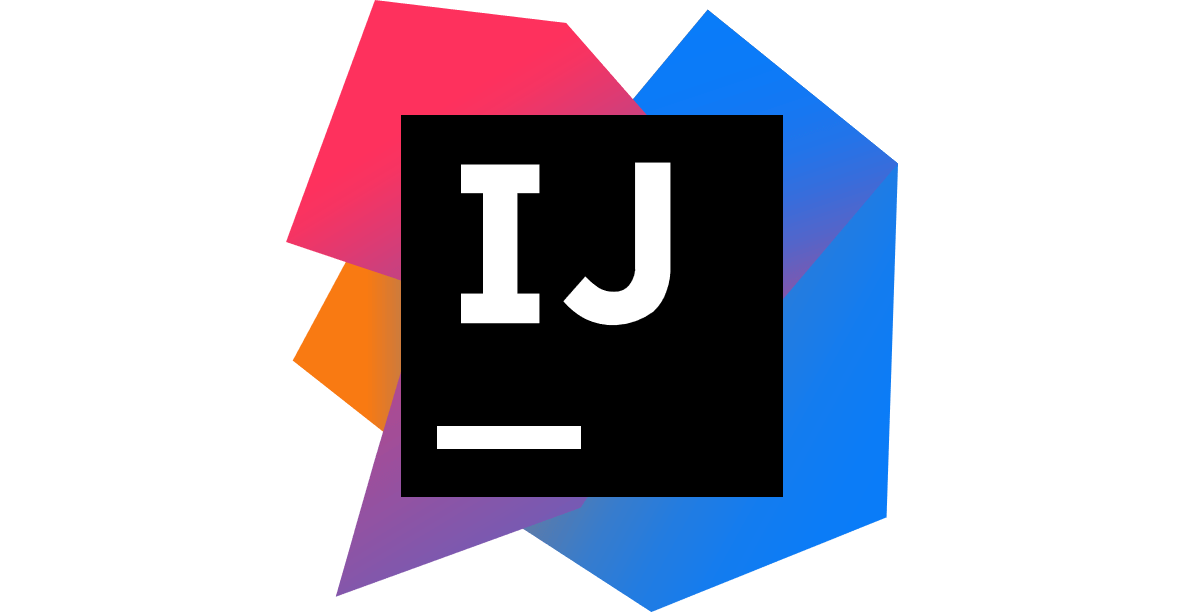
## 2.1 Ταξινομητές Bayes

Το σύστημα αυτοματοποιημένης συμπεριφορικής ανάλυσης πειραμάτων με χρήση υπολογιστικής όρασης σχεδιάζεται με τους παρακάτω στόχους.

1. Βαθμολόγηση πειραμάτων TST χωρίς παρουσία χρήστη
2. Έλεγχος διαδικασίας μέσω γραφικού περιβάλλοντος χρήστη (GUI)
3. Κατανοητή ροή εργασίας και εύκολη αποθήκευση αποτελεσμάτων
4. Δυνατότητα αλλαγής παραμέτρων αλγορίθμου ανάλυσης
5. Πλήρης απεικόνιση δεδομένων και οπτικοποίηση κατά την αναπαραγωγή βίντεο πειραμάτων

## Ταξινομητές KNN

Το ολοκληρωμένο περιβάλλον ανάπτυξης IntelliJ IDEA παρέχει την δυνατότητα ανάπτυξης εφαρμογών Java ενσωματώνοντας όλα τα απαραίτητα εργαλεία για κάθε στάδιο δημιουργίας και δοκιμών (JetBrains, 2017). Το IntelliJ IDEA περιέχει δυνατότητες γρήγορης δημιουργίας project, άμεσης εκτέλεσης και διαγνωστικών debugging, διαχείρισης αρχείων και βιβλιοθηκών, δημιουργίας εκτελέσιμων αρχείων, εύκολης μετονομασίας, αυτόματης συμπλήρωσης και έξυπνων προτάσεων βελτιστοποίησης κώδικα και ανίχνευσης λογικών λαθών.



Εικόνα 2: IntelliJ IDEA

Το περιβάλλον επίσης ενσωματώνει διεπαφή με συστήματα version control (github) και υποστηρίξει επεξεργασία συνοδευτικών αρχείων κώδικα (XML,CSS).

H βιβλιοθήκη JavaFX αποτελεί το σύγχρονο σύστημα σχεδιασμού γραφικού περιβάλλοντος χρήστη (GUI) της Oracle (Docs.oracle.com, 2017). To JavaFX API (Application Programming Interface) περιέχει εύχρηστα αντικείμενα για τον έλεγχο των στοιχείων του γραφικού περιβάλλοντος.



Εικόνα 3: JavaFX

H σχεδίαση του γραφικού περιβάλλοντος πραγματοποιείται σε τροποποιημένη γλώσσα XML και η εμφάνιση των στοιχείων μπορεί να τροποποιηθεί με χρήση αρχείων CSS (Cascading Style Sheets). Στην βιβλιοθήκη JavaFX περιλαμβάνεται επίσης λογισμικό σχεδιασμού και τροποποίησης γραφικού περιβάλλοντος Αλγόριθμοι βαθμονόμησης στερεοσκοπικής όρασης (calib3d)

## Δίκτυα Bayes

Η εφαρμογή υλοποιείται ως ένα εκτελέσιμο πρόγραμμα σε μορφή JAR, ένα εκτελέσιμο αρχείο Java που περιλαμβάνει τον εκτελέσιμο κώδικα και τα απαραίτητα συνοδευτικά αρχεία. Η εφαρμογή βασίζεται στο γραφικό περιβάλλον και περιλαμβάνει δυο επιπλέον νήματα (threads) εκτέλεσης τα οποία εκτελούν τα υπολογιστικά απαιτητικά μέρη της εφαρμογής. Το κυρίως thread διαχειρίζεται το περιβάλλον χρήστη ενώ τα αλλά δυο διαχειρίζονται την αναπαραγωγή και την ανάλυση των πειραμάτων (Εικόνα 6).



Εικόνα 6: Λειτουργικό διάγραμμα εφαρμογής

Το thread του περιβάλλοντος χρήστη είναι το κύριο thread της εφαρμογής και αναλαμβάνει την διαχείριση των γραφικών στοιχείων ελέγχου όπως η απόκριση σε click κουμπιών, αλλαγής επιλογών ή αλλαγής θέσης συρόμενων ρυθμιστών. Στο ίδιο thread επίσης εκτελούνται οι απαραίτητες μετατροπές για την απεικόνιση δεδομένων (εικόνες και διαγράμματα) τα οποία προέρχονται από τα thread επεξεργασίας και αναπαραγωγής.

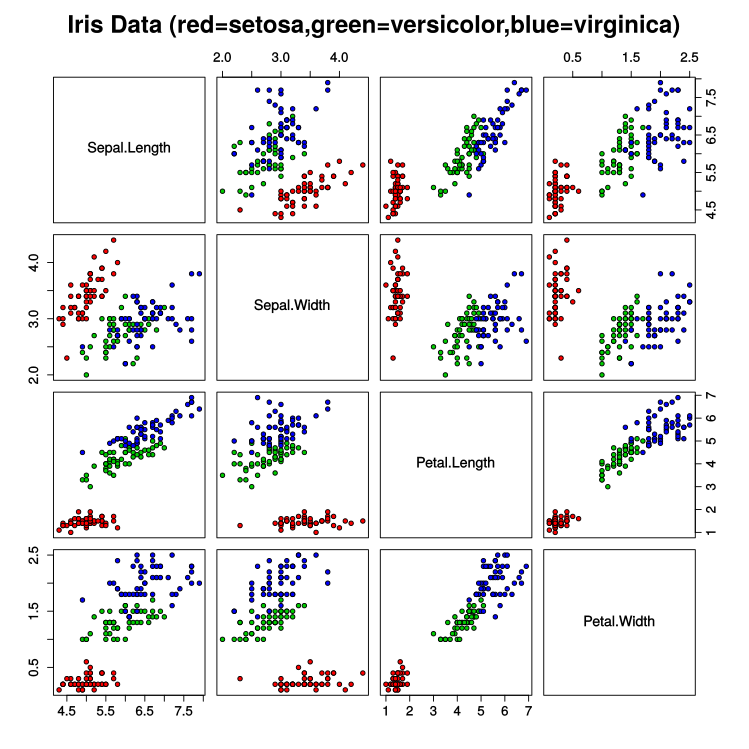


Εικόνα 6: Λειτουργικό διάγραμμα εφαρμογής

## Σύνολα Δεδομένων

### Fisher Iris

Το dataset περιλαμβάνει μετρήσεις μορφολογικών χαρακτηριστικών για τρία είδη ανθών ίριδας. Συνολικά υπάρχουν 50 δείγματα για κάθε είδος (Iris setosa, Iris virginica and Iris versicolor). Το dataset συνήθως χρησιμοποιείται για την δοκιμή στατιστικών τεχνικών μηχανικής μάθησης.



Εικόνα 10:. Απεικόνιση του Fisher’s Iris flower dataset

### Pima Indians Diabetes

Το pima Indians diabetes dataset περιλαμβάνει χαρακτηριστικά για γυναίκες άνω των 21 οι οποίες κατάγονται από την φυλή pima και ο στόχος του είναι η πρόβλεψη για το εάν η ασθενής πάσχει από διαβήτη.

|  |  |
| --- | --- |
| 1 | Number of times pregnant |
| 2 | Plasma glucose concentration a 2 hours in an oral glucose tolerance test |
| 3 | Diastolic blood pressure (mm Hg) |
| 4 | Triceps skin fold thickness (mm) |
| 5 | 2-Hour serum insulin (mu U/ml) |
| 6 | Body mass index (weight in kg/(height in m)^2) |
| 7 | Diabetes pedigree function |
| 8 | Age (years) |
| 9 | Class variable (0 or 1) |

Εικόνα 11:. Χαρακτηριστικά του Pima Indians Diabetes Dataset

### Census Income

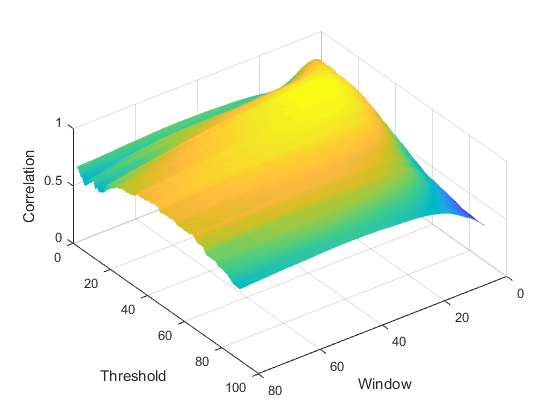
To census income dataset περιλαμβάνει δεδομένα απογραφής ενηλίκων του 1994 από το U.S. Census Bureau. Το σύνολο δεδομένων έχει ως στόχο την πρόβλεψη της τάξης εισοδήματος (<=50Κ ή >50K) με βάση αλλά χαρακτηριστικά των απογραφέντων.

|  |  |
| --- | --- |
| 1 | workclass |
| 2 | education |
| 3 | maritalstatus |
| 4 | occupation |
| 5 | race |
| 6 | sex |
| 7 | nativecountry |
| 8 | class |

Εικόνα 10:. Χαρακτηριστικά που χρησιμοποιούνται από το Census Income Dataset

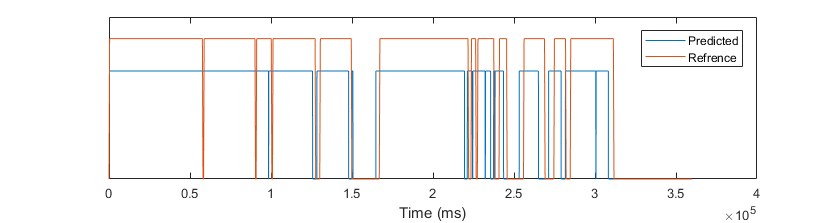
# Πειραματικά Αποτελέσματα

Για τις δυο παραμέτρους του αλγορίθμου (μέγεθος παραθύρου, κατώφλι κίνησης), επιλέγονται οι τιμές τους εξετάζοντας έναν μεγάλο αριθμό συνδυασμών και υπολογίζοντας την συσχέτιση με τους πραγματικούς βαθμολογητές στον συνολικό χρόνο κίνησης. Όπως φαίνεται στην εικόνα 14 η βέλτιστη συσχέτιση επιτυγχάνεται για μέγεθος παραθύρου 16 και για κατώφλι 23.



Εικόνα 14: Απόδοση αλγορίθμου συναρτήσει των παραμέτρων

Στην εικόνα 15 απεικονίζεται η προβλεπόμενη βαθμολόγηση του πειράματος σε σύγκριση με την πραγματική και παρατηρείται ότι το σύστημα επιτυγχάνει να ανιχνεύει σωστά την διάρκεια κάθε κίνησης εμφανίζοντας μια μικρή προήγηση στον χρόνο που πιθανώς οφείλεται στον χρόνο αντίδρασης του βαθμολογητή.



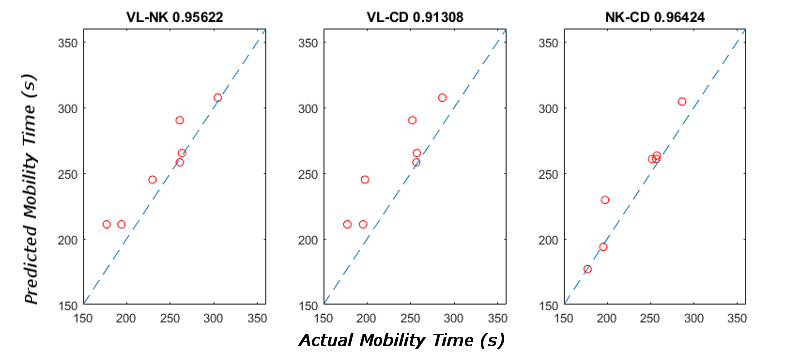
Εικόνα 15: Σύγκριση πραγματικού (πορτοκαλί) με προβλεπόμενο (μπλέ) σκορ

Για την εκτίμηση της απόδοσης του τελικού συστήματος εξετάζεται η συσχέτιση των εκτιμήσεων του συνολικού χρόνου κίνησης του κάθε πειραματόζωου που υπολογίζει το αυτοματοποιημένο σύστημα (VL) με αυτόν που προκύπτει από τους δυο βαθμολογητές (NK, CD). Στον πίνακα 1 φαίνονται οι συντελεστές συσχέτισης ανά δυο βαθμολογητές και φαίνεται ότι η συσχέτιση του συστήματος με κάθε βαθμολογητή πλησιάζει την συσχέτιση μεταξύ των δυο βαθμολογητών αναφοράς (ΝΚ,CD).

|  |  |  |  |
| --- | --- | --- | --- |
|  | Αυτόματο Σύστημα (VL) | Βαθμολογητής 1 (NK) | Βαθμολογητής 2 (CD) |
| Αυτόματο Σύστημα (VL) | 1 | 0.95622 | 0.91308 |
| Βαθμολογητής 1 (NK) | 0.95622 | 1 | 0.96424 |
| Βαθμολογητής 2 (CD) | 0.91308 | 0.96424 | 1 |

Πίνακας 1: Συντελεστές συσχέτισης μεταξύ κάθε ζεύγους βαθμολογητών (γραμμή, στήλη)

Στην εικόνα 17 φαίνονται οι προβλεπόμενες τιμές συναρτήσει των πραγματικών μεταξύ του συστήματος και κάθε βαθμολογητή και μεταξύ των δυο βαθμολογητών. Από το σχήμα φαίνεται ότι το αυτόματο σύστημα παρουσιάζει σφάλματα μερικών δευτερολέπτων τα οποία όμως δεν υπερβαίνουν σε διάρκεια τα σφάλματα που εμφανίζονται μεταξύ βαθμολογητών.



Εικόνα 17: Διάγραμμα αποτελεσμάτων αυτοματοποιημένου συστήματος (VL) σε σύγκριση με των βαθμολογητών (NK,CD)

# Συμπεράσματα

Από την χρήση του τελικού λογισμικού φαίνεται ότι το σύστημα προσφέρει ένα φιλικό προς τον χρήστη περιβάλλον με κατανοητή ροή εργασίας και απλό σύστημα διαχείρισης αρχείων. Το λογισμικό επίσης δεν απαιτεί εγκατάσταση με αποτέλεσμα να μην απαιτεί συγκεκριμένο σταθμό εργασίας και να είναι προσιτό σε οποιονδήποτε χρήστη. Επίσης η χρονική επίδοση του συστήματος εξαρτάται από τους διαθέσιμους υπολογιστικούς πόρους με αποτέλεσμα η ανάλυση να μπορεί να ολοκληρωθεί σε μερικά δευτερόλεπτα σε σύγχρονους επεξεργαστές.

# Βιβλιογραφία

[1] Can, A., Dao, D., Terrillion, C., Piantadosi, S., Bhat, S. and Gould, T. (2011). The Tail Suspension Test. Journal of Visualized Experiments, (58).

[2] Juszczak, G., Sliwa, A., Wolak, P., Tymosiakzueliniska, A., Lisowski, P. and Swiergiel, A. (2006). The usage of video analysis system for detection of immobility in the tail suspension test in mice. Pharmacology Biochemistry and Behavior, 85(2), pp.332-338.

[3] Lad, H., Liu, L., Payá-Cano, J., Fernandes, C. and Schalkwyk, L. (2007). Quantitative traits for the tail suspension test: automation, optimization, and BXD RI mapping. Mammalian Genome, 18(6-7), pp.482-491.

[4] JetBrains. (2017). IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains. [online] Available at: https://www.jetbrains.com/idea/ [Accessed 15 Oct. 2017].

[5] Docs.oracle.com. (2017). JavaFX 2 Tutorials and Documentation. [online] Available at: http://docs.oracle.com/javafx/2/overview/jfxpub-overview.htm [Accessed 15 Oct. 2017].

[6] Opencv.org. (2017). OpenCV library. [online] Available at: https://opencv.org/ [Accessed 15 Oct. 2017].

[7] Commons.apache.org. (2017). Commons Math: The Apache Commons Mathematics Library. [online] Available at: http://commons.apache.org/proper/commons-math/ [Accessed 15 Oct. 2017]. (Commons.apache.org, 2017)

# Κώδικας

## KNN Classifier

**nfoldDataset.m**

**function** **[** dataset **]** **=** nFoldDataset**(** database**,** folds **)**

%NFOLDDATASET Summary of this function goes here

% Detailed explanation goes here

**[**m**,**n**]** **=** size**(**database**);**

database**(**randperm**(**m**),:)=**database**;**

seg**=**uint32**(**round**(**m**/**folds**));**

**for** i**=**1**:**folds

validx**=((**folds**-**i**)\***seg**)+**1**:(**folds**-**i**+**1**)\***seg**;**

dataset**(**i**).**train**=**database**;**

dataset**(**i**).**train**(**validx**,:)=[];**

dataset**(**i**).**val**=**database**(**validx**,:);**

**end**

**end**

**knnclassifier.m**

%KNN Classifier function

**function** accur **=** knnclassifier**(**traindata**,** testdata**,** K**)**

%Find distance with all training datapoints, sort and poll

**for** i **=** 1 **:** size**(**testdata**)**

x **=** testdata**(**i**,:);**

dist **=** sqrt**((**traindata**(:,** 1**)** **-** x**(**1**))** **.^** 2 **+** **(**traindata**(:,** 2**)** **-** x**(**2**))** **.^** 2 **+** **(**traindata**(:,** 3**)** **-** x**(**3**))** **.^** 2 **+** **(**traindata**(:,** 4**)** **-** x**(**4**))** **.^** 2**);**

classes **=** traindata**(:,** 5**);**

dist**(:,** 2**)** **=** classes**;**

poll **=** sortrows**(**dist**,** 1**);**

%For tiebreak in case of even K

**if** **(**mod**(**K**,** 2**)** **==** 1**)**

expclass**(**i**)** **=** mode**(**poll**(**1 **:** K**,** 2**));**

**else**

temp **=** poll**(**1 **:** K**,** 2**);**

uniq **=** unique**(**temp**);**

p **=** size**(**uniq**);**

bincounts **=** histc**(**temp**,** uniq**);**

q **=** max**(**bincounts**);**

%if number of unique elements = 2 && highest frequency is K/2, then there is tie

M **=** **(**p **==** 2**)** **&** **(**q **==** K**/**2**);**

%Alloted the class which is at closest distance

expclass**(**i**)** **=** mode**(**poll**(**1 **:** K **-** M**,** 2**));**

**end**

**end**

%Error percentage calculation

error **=** transpose**(**expclass**)** **-** testdata**(:,**5**);**

accur **=** **((**size**(**error**,** 1**)** **-** nnz**(**error**))/**size**(**error**,** 1**));**

**end**

**nfold\_iris.m**

clear**;**

iris **=** datatable2mat**(**dataset2table**(**dataset**(**'File'**,** 'iris.data'**,**...

'Delimiter'**,** ','**,** 'ReadVarNames'**,** false**)));**

folds **=** 10**;**

iris **=** nFoldDataset**(**iris**,**folds**);**

**for** K**=**1**:**9

**for** f**=**1**:**folds

err**(**f**)=**knnclassifier**(**iris**(**f**).**train**,**iris**(**f**).**val**,** K**);**

**end**

sub **=** subplot**(**3**,**3**,** K**);**

plot**(**err**);**

title**([**'Plot for K = '**,** num2str**(**K**)])**

**end**

**function** dat **=** datatable2mat**(**datatable**)**

%Converts table with numerical and categorical data to matrix

c**=** grp2idx**(**table2array**(**datatable**(:,end)));**

dat **=** table2array**(**datatable**(:,**1**:end-**1**));**

dat**(:,end+**1**)=**c**;**

**end**

## Bayes Classifier

**bayes\_classifier.m**

% Bayes Classifier

% Assumes last column of input TRAIN and TEST matrices is the class

%

% set covarianceMode to 1 for common diagonal covariance matrix

% set covarianceMode to 2 for common covariance matrix

% set covarianceMode to 3 for a seperate covariance matrix for each class

**function** testval **=** bayes\_classifier**(**XTRAIN**,** XTEST**,** covarianceMode**)**

**if** nargin **<** 3

fprintf**(**'No covarianceMode specified : '**);**

covarianceMode **=** 3**;**

**end**

fprintf**(**'Fitting Bayes Classifier :\n'**);**

trainDataSize **=** size**(**XTRAIN**,**1**);**

testDataSize **=** size**(**XTEST**,**1**);**

% Separate all the training samples into separate classes

**[**C**,~,**idx**]** **=** unique**(**XTRAIN**(:,end));**

classes **=** accumarray**(**idx**,**1**:**trainDataSize**,[],@(**r**){**XTRAIN**(**r**,**1**:end-**1**)});**

% Generate the summary map from all the classes

classMap **=** containers**.**Map**(**C**,** classes**);**

covMap **=** containers**.**Map**(**'KeyType'**,**'int32'**,**'ValueType'**,**'any'**);**

meanMap **=** containers**.**Map**(**'KeyType'**,**'int32'**,**'ValueType'**,**'any'**);**

% -------------------Calculate covariance and mean-------------------------

**switch** covarianceMode

**case** 1 % Use a common DIAGONAL covariance regardless of class

fprintf**(**'using COMMON DIAGONAL covariance matrix for all classes :\n'**);**

% First gather all features

allClassesFeatures **=** **[];**

**for** i **=** 1**:**size**(**C**,**1**)**

c **=** C**(**i**);**

allClassesFeatures **=** **[**allClassesFeatures**;** classMap**(**c**)];**

**end**

% Calculate one covariance matrix regardless of class and multiply

% it by the diagonal matrix

covariance **=** var**(**allClassesFeatures**(:))** **\*** eye **(**size**(**allClassesFeatures**,**2**));**

**for** i **=** 1**:**size**(**C**,**1**)**

c **=** C**(**i**);**

covMap**(**c**)** **=** covariance**;** % Use the same covariance matrix for all classes

meanMap**(**c**)** **=** mean**(**classMap**(**c**),**1**);**

**end**

**case** 2 % Use a common covariance matrix regardless of class

fprintf**(**'using COMMON covariance matrix for all classes :\n'**);**

allClassesFeatures **=** **[];**

**for** i **=** 1**:**size**(**C**,**1**)**

c **=** C**(**i**);**

allClassesFeatures **=** **[**allClassesFeatures**;** classMap**(**c**)];**

**end**

% Calculate one covariance matrix regardless of class

covariance **=** cov**(**allClassesFeatures**);**

**for** i **=** 1**:**size**(**C**,**1**)**

c **=** C**(**i**);**

covMap**(**c**)** **=** covariance**;**

meanMap**(**c**)** **=** mean**(**classMap**(**c**),**1**);**

**end**

**case** 3

% Use seperate covariance matrices for each class

fprintf**(**'using SEPERATE covariance matrix for each class :\n'**);**

**for** i **=** 1**:**size**(**C**,**1**)**

c **=** C**(**i**);**

**if** size**(**classMap**(**c**),**1**)** **~=** 0

covMap**(**c**)** **=** cov**(**classMap**(**c**));** % Calculate the covariance matrix of THIS class

meanMap**(**c**)** **=** mean**(**classMap**(**c**),**1**);**

**end**

**end**

**end**

% -------------Find the accuracy of training and testing data--------------

% First get the a priori probabilities for each class

priors **=** **[];** % Initialise some space to store the priors

**for** i **=** 1**:**size**(**C**,**1**)** % Loop over all classes

c **=** C**(**i**);** % Get the current class

% Get the a priori probability of this class

priorProb **=** size**(**classMap**(**c**),**1**)/**trainDataSize**;**

% Store it for later use

priors **=** cat**(**2**,** priors**,** priorProb**);**

**end**

% Training ----------------------------------------------------------------

% Initialise some space to store the estimated class for each x

calculatedY **=** **[];**

% Do the estimate!

**for** trainIndex**=**1**:**trainDataSize % Loop over all x in the train set

prob **=** **[];** % Initialise some space to store the resulting probabilities

% Calculate the a posteriori probabilities for all classes

% for current x

**for** i **=** 1**:**size**(**C**,**1**)** % Loop over all classes

c **=** C**(**i**);** % Get the current class

% Get the current class' a posteriori probability given current x

% using mean and covariance for this class.

% (Note: depending on the covarianceMode chosen, common and diagonal,

% common or seperate covariance matrices might be used. See above.)

% (Note: no need to normalize by p(x), as its the same everywhere)

condProb **=** mvnpdf**(**XTRAIN**(**trainIndex**,**1**:end-**1**),** meanMap**(**c**),** covMap**(**c**))\***priors**(**i**);**

% Put the a posteriori and a priori probabilities of current class

% for current x side by side for

% later comparison

prob **=** cat**(**2**,** prob**,** condProb**);**

**end**

% Get the greatest a posteriori probability (estimate of which class the

% x falls under)

**[**maxval**,**argmax**]** **=** max**(**prob**);**

% Store the estimated class for later use

calculatedY **=** **[**calculatedY**;**argmax**];**

**end**

% Now count how many estimates were right by comparing them with the labels

accuratePredictions **=** **(**calculatedY **==** XTRAIN**(:,end));**

% Get the accuracy percentage

trainingAccuracy **=** sum**(**accuratePredictions**)\***100**/**size**(**accuratePredictions**,**1**);**

fprintf**(**'Training Accuracy = %4.2f\n'**,**trainingAccuracy**);**

%Testing ------------------------------------------------------------------

% Initialise some space to store the estimated class for each x

calculatedY **=** **[];**

% Do the estimate!

**for** testIndex**=**1**:**testDataSize % Loop over all x in the train set

prob **=** **[];** % Initialise some space to store the resulting probabilities

% Calculate the a posteriori probabilities for all classes

% for current x

**for** i **=** 1**:**size**(**C**,**1**)** % Loop over all classes

c **=** C**(**i**);** % Get the current class

% Get the current class' a posteriori probability given current x

% using mean and covariance for this class.

% (Note: depending on the covarianceMode chosen, common and diagonal,

% common or seperate covariance matrices might be used. See above.)

% (Note: no need to normalize by p(x), as its the same everywhere)

condProb **=** mvnpdf**(**XTEST**(**testIndex**,**1**:end-**1**),** meanMap**(**c**),** covMap**(**c**))\***priors**(**i**);**

% Put the a posteriori and a priori probabilities of current class

% for current x side by side for

% later comparison

prob **=** cat**(**2**,** prob**,** condProb**);**

**end**

% Get the greatest a posteriori probability (estimate of which class the

% x falls under)

**[**maxval**,**argmax**]** **=** max**(**prob**);**

% Store the estimated class for later use

calculatedY **=** **[**calculatedY**;**argmax**];**

**end**

% Now count how many estimates were right by comparing them with the labels

accuratePredictions **=** **(**calculatedY **==** XTEST**(:,end));**

% Get the accuracy percentage

testAccuracy **=** sum**(**accuratePredictions**)\***100**/**size**(**accuratePredictions**,**1**);**

fprintf**(**'Test Accuracy = %4.2f\n\n'**,**testAccuracy**);**

% return the accuracy percentage to the caller

testval **=** testAccuracy**;**

**gaussian\_bayes.m**

clear**;**

close all**;**

tic

rng**(**8**);** % For reproducibility

load**(**'../datasets.mat'**);**

% Randomize dataset

dataSetSize **=** size**(**pimaindiansdiabetes**,**1**);**

data\_rand **=** table2array**(**pimaindiansdiabetes**(**randperm**(**dataSetSize**),:));**

data\_rand**(:,end)** **=** data\_rand**(:,end)** **+** 1**;** %Shift class one value up to help with later calculations

accuracies **=** **[];**

% Calculate accuracy rates for all different covariance modes using

% 10fold cross-validation method

**for** i **=** 1**:**3 % Loop over all covariance modes

% mode 1 for common diagonal covariance matrix

% mode 2 for common covariance matrix

% mode 3 for a seperate covariance matrix for each class

covarianceMode **=** i**;**

% Wrap the classifier function inside another so we can pass the mode as a

% parameter and keep matlab happy.

% That is because matlab's crossval function takes as first

% parameter a function who's signature is as follows:

% @(XTRAIN, XTEST)(success\_rate)

% The first being the train data set and the seconf the test data set.

% Our bayes\_classifier accepts a third parameter as well.

classifier **=** **@(**XTRAIN**,** XTEST**)(**bayes\_classifier**(**XTRAIN**,** XTEST**,** covarianceMode**));**

% 10fold cross-validate

overallAccuracy **=** mean**(**crossval**(**classifier**,** data\_rand**));**

fprintf**(**'Bayes Overall Accuracy = %4.2f\n'**,**overallAccuracy**);**

accuracies **=** **[**accuracies**;** overallAccuracy**];**

**end**

figure**;**

x **=** categorical**({**'common diagonal covariance matrix'**,**'common covariance matrix'**,**'seperate covariance matrix for each class'**});**

bar**(**x**,** accuracies**);**

title**(**'Comparison of classifier accuracy with different covariance modes'**);**

xlabel**(**'Covariance Mode:'**);** % x-axis label

ylabel**(**'Accuracy %:'**);** % y-axis label

**for** i**=**1**:**3

text**(**x**(**i**),**accuracies**(**i**),**num2str**(**accuracies**(**i**),**'%0.2f'**),**...

'HorizontalAlignment'**,**'center'**,**...

'VerticalAlignment'**,**'bottom'**)**

**end**

toc

## Naïve Bayes Classifier

**naivebayesindians.m**

clear**;**

load**(**'../datasets.mat'**);**

dat **=** datatable2mat**(**pimaindiansdiabetes**);**

folds **=** nFoldDataset**(**array2table**(**dat**),**10**);**

**for** i**=**1**:**10

dat **=** table2array**(**folds**(**i**).**train**);**

class1 **=** dat**(**dat**(:,**9**)** **==** 1**,** 1**:**8**);**

mu1 **=** mean**(**class1**);**

sig1 **=** var**(**class1**).\***eye**(**8**);**

class2 **=** dat**(**dat**(:,**9**)** **==** 2**,** 1**:**8**);**

mu2 **=** mean**(**class2**);**

sig2 **=** var**(**class2**).\***eye**(**8**);**

dat **=** table2array**(**folds**(**i**).**val**);**

p**=[**mvnpdf**(**dat**(:,**1**:**8**),**mu1**,**sig1**)** mvnpdf**(**dat**(:,**1**:**8**),**mu2**,**sig2**)];**

**[**x**,**pred**]** **=** max**(**p**,[],**2**);**

acc**(**i**)** **=**1**-(**sum**(**abs**(**pred**-**dat**(:,**9**)))/**length**(**pred**));**

**end**

disp**([**'Accuracy: ' num2str**(**mean**(**acc**))]);**

**function** dat **=** datatable2mat**(**datatable**)**

%Converts table with numerical and categorical data to matrix

c**=** grp2idx**(**table2array**(**datatable**(:,end)));**

dat **=** table2array**(**datatable**(:,**1**:end-**1**));**

dat**(:,end+**1**)=**c**;**

**end**

## Bayesian Net

**bayesNetIncome.m**

clear**;**

load**(**'income-usable.mat'**);**

N**=**8**;**

**for** i**=**1**:**N

classes**{**i**}=**unique**(**income**(:,**i**));**

classnum**(**i**)=**height**(**classes**{**i**});**

data**(:,**i**)=**grp2idx**(**table2array**(**income**(:,**i**)));**

**end**

% data = data(1:300,:);

%Create network

% 2 3 5 6 7

% \ / \ | /

% 1 4

% \ /

% 8

dag **=** zeros**(**N**,**N**);**

dag**(**2**,**1**)=**1**;** dag**(**3**,**1**)=**1**;**

dag**(**5**,**4**)=**1**;** dag**(**6**,**4**)=**1**;** dag**(**7**,**4**)=**1**;**

dag**(**1**,**8**)=**1**;** dag**(**4**,**8**)=**1**;**

%Create network

onodes **=** **[**2**,**3**,**5**,**6**,**7**];**

node\_sizes **=** classnum**;**

bnet **=** mk\_bnet**(**dag**,** node\_sizes**,** 'observed'**,** onodes**);**

% use random params

**for** i**=**1**:**N

bnet**.**CPD**{**i**}** **=** tabular\_CPD**(**bnet**,** i**);**

**end**

%train network

ncases **=** size**(**data**,** 1**);**

cases **=** num2cell**(**data**');**

engine **=** jtree\_inf\_engine**(**bnet**);**

bnet **=** learn\_params\_em**(**engine**,** cases**);**

%inference forward

engine **=** jtree\_inf\_engine**(**bnet**);**

evidence **=** cell**(**1**,**N**);**

evidence**(**onodes**)** **=** num2cell**([**10 5 5 2 12**]);**

**[**engine**,** ll**]** **=** enter\_evidence**(**engine**,** evidence**);**

p8 **=** marginal\_nodes**(**engine**,** 8**);**

%inference inverse

nodes **=** **[**1**];**

engine **=** jtree\_inf\_engine**(**bnet**);**

evidence **=** cell**(**1**,**N**);**

evidence**(**nodes**)** **=** num2cell**([**3**]);**

**[**engine**,** ll**]** **=** enter\_evidence**(**engine**,** evidence**);**

p2 **=** marginal\_nodes**(**engine**,** 2**);**