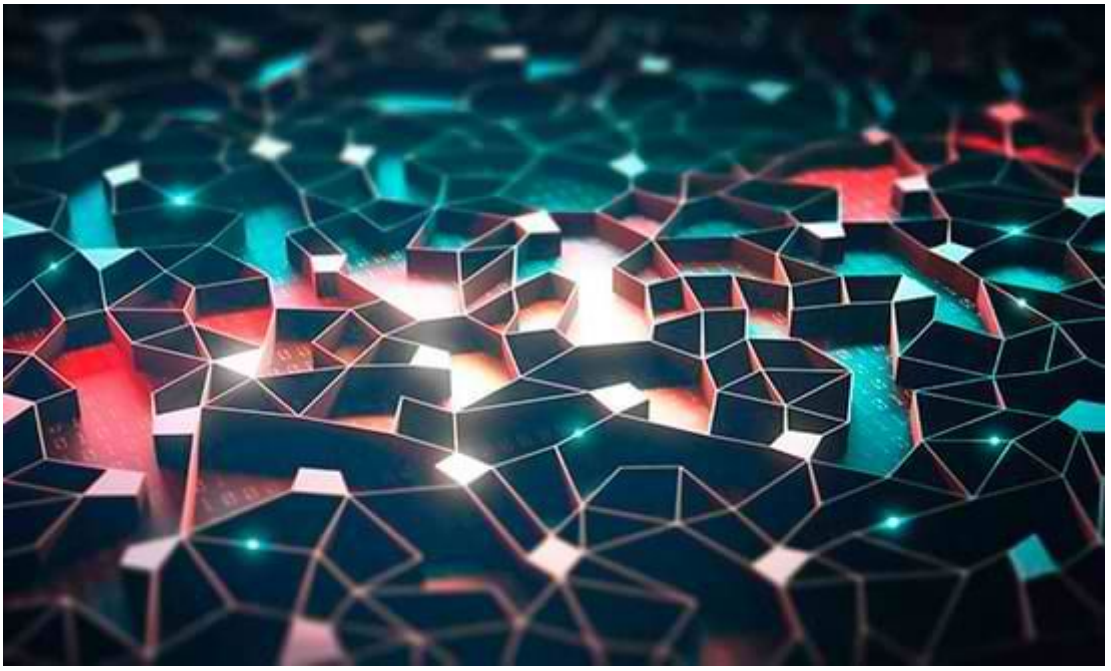
9/12/19

Πανεπιστήμιο δυτικής Αττικής
Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Πληροφορικής



Αναγνώριση προτύπων & Μηχανική Μάθηση

Σφυριδάκη Αγγελική
cs151036
cs151036@uniwa.gr

Περιεχόμενα

Αρχεία Κώδικα:

InputData:

Dataset2Use_Assignment2.xlsx:αρχείο που περιέχει όλα τα δεδομένα εισόδου για την εφαρμογή.

OutputData:

Results.xlsx:Τα αποτελέσματα που προκύπτουν από όλα τα μοντέλα του MainScript1

Results2.xlsx:Τα αποτελέσματα που προκύπτουν από όλα τα μοντέλα του MainScript2

MainScript1.py: Ο βασικός κώδικας της εφαρμογής.

MainScript2.py: Παρόμοιος με τον παραπάνω, με την διαφορά ότι κάνει downsampling.

(Άλλα αρχεία που αφορούν το μέρος 2.)

1ο Μέρος

Σκοπός εργασίας:

Ο σκοπός της εργασίας είναι, με συγκεκριμένα δεδομένα, να εκπαιδεύσουμε ένα σύνολο μοντέλων και στο τέλος να επιλέξουμε τον καταλληλότερο ταξινομητή με την καλύτερη απόδοση. Ειδικότερα, θα πρέπει να εντοπίζει με όσο το δυνατόν μεγαλύτερη ακρίβεια, τις εταιρείες εκείνες που θα χρεοκοπήσουν.

Συναρτήσεις:

def **MakeScoreLists1**(className,y_train,y_pred_train): προσθέτει στην λίστα που θα αποθηκευτεί στο excel, μια ακόμα εγγραφή που αφορά το train set.

def **MakeScoreLists2**(className,y_test,y_pred_test):προσθέτει στην λίστα που θα αποθηκευτεί στο excel, μια ακόμα εγγραφή που αφορά το test set.

def **CheckBest**(classname,recall,tn,fp): ελέγχει την καταλληλότητα των ταξινομητών.

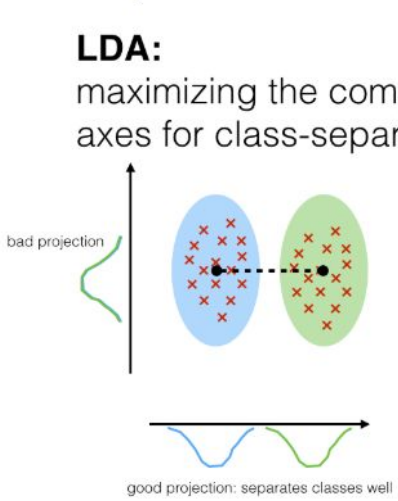
Επεξήγηση κώδικα:

Τα δεδομένα που θα διαβάσουμε βρίσκονται στο αρχείο 'Dataset2Use_Assignment2.xlsx'. Αρχικά, τα χωρίζουμε σε inputData και outputData, όπου αποθηκεύουμε τα χαρακτηριστικά και τα αντίστοιχα αποτελέσματα. Εν συνεχεία, κατηγοριοποιούμε περαιτέρω τα δεδομένα σε train και test set και τα συναφή αποτελέσματα τους. Εφαρμόζουμε μια διαδικασία κανονικοποίησης και τώρα είμαστε πλέον έτοιμοι να εκπαιδεύσουμε τα μοντέλα. Τα αποτελέσματα θα αποθηκευτούν στην λίστα transpose, της οποίας κάθε στοιχείο θα είναι και μια στήλη του excel.

Τα μοντέλα που επιλέξαμε είναι τα Linear Discriminant Analysis, Logistic Regression, Decision Trees, k-Nearest Neighbors, Naïve Bayes, Support Vector Machines και Neural Networks. Να σημειωθεί ότι για κάθε μοντέλο προβλέπουμε και το train και το test set. Εν τέλη, αποθηκεύουμε την πληροφορία σε ένα excel με όνομα Results.xlsx και εκτυπώνουμε τα μοντέλα που εντοπίζουν, τις εταιρείες που θα πτωχεύσουν με ποσοστό επιτυχίας τουλάχιστον 62% και τις εταιρείες που δεν θα πτωχεύσουν με ποσοστό επιτυχίας τουλάχιστον 70% , εάν υπάρχουν.

Μοντέλα:

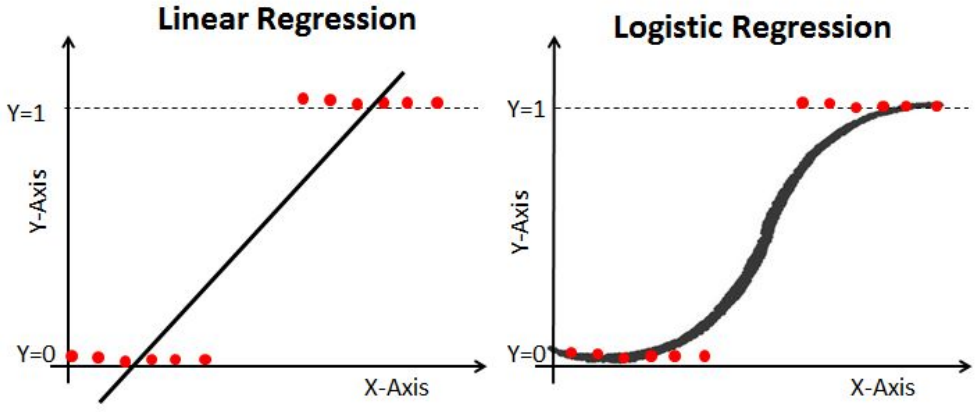
Linear Discriminant Analysis:



Η LDA χρησιμοποιείται συνήθως ως <u>τεχνική μείωσης των διαστάσεων</u> στο στάδιο της προεπεξεργασίας για εφαρμογές αναγνώρισης προτύπων και μηχανικής μάθησης. Ο στόχος είναι, από ένα σύνολο δεδομένων, να μεταφερθούμε σε ένα χώρο μικρότερων διαστάσεων με καλή διαχωρισιμότητα των κλάσεων, προκειμένου να αποφευχθεί το overfitting και επίσης να μειωθεί το υπολογιστικό κόστος.

Η γενική προσέγγιση LDA είναι αρκετά παρόμοια με την PCA, αλλά εκτός από την εύρεση των συνιστωσών των αξόνων που μεγιστοποιούν τη διακύμανση των δεδομένων μας (PCA), ενδιαφερόμαστε επιπλέον στους άξονες που μεγιστοποιούν τον διαχωρισμό μεταξύ των πολλαπλών κλάσεων (LDA). Το LDA είναι "εποπτευόμενο" και υπολογίζει τις κατευθύνσεις ("γραμμικές διαφοροποιήσεις") που θα αντιπροσωπεύουν τους άξονες που μεγιστοποιούν τον διαχωρισμό μεταξύ των πολλαπλών τάξεων.

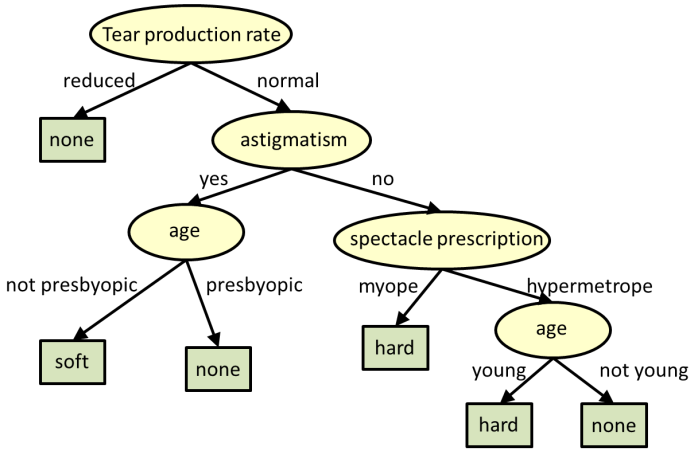
Logistic Regression:



Αποτελεί ένα στατιστικό μοντέλο που στη βασική του μορφή χρησιμοποιεί μια λογική λειτουργία για να μοντελοποιήσει μια δυαδική εξαρτώμενη μεταβλητή. Περαιτέρω μπορούμε να χωρίσουμε την μέθοδο Logistic Regression σε τρεις κατηγορίες. Η Binary Logistic Regression όπου η κατηγορική απάντηση έχει μόνο δύο πιθανά αποτελέσματα. Παράδειγμα: Spam ή Not. Η Multinomial Logistic Regression όπου έχει τρεις ή περισσότερες κατηγορίες χωρίς ταξινόμηση. Παράδειγμα: Προβλέψτε ποια τρόφιμα προτιμάτε περισσότερο (Veg, Non-Veg, Vegan). Η Ordinal Logistic Regression όπου έχει τρεις ή περισσότερες κατηγορίες με ταξινόμηση. Παράδειγμα: Βαθμολογία ταινίας από 1 έως 5. Προκειμένου να

προβλεφθεί σε ποια κλάση ένα δεδομένο ανήκει, μπορεί να οριστεί ένα όριο. Με βάση το όριο αυτό, η εκτιμώμενη πιθανότητα που προκύπτει κατηγοριοποιείται σε κλάσεις.Το όριο απόφασης μπορεί να είναι γραμμικό ή μη γραμμικό. Η σειρά πολυωνύμων μπορεί να αυξηθεί για να πάρει περίπλοκο όριο απόφασης.

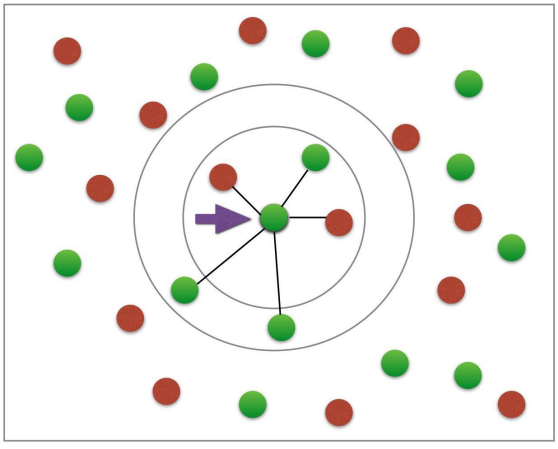
Decision Trees:



Ένα δέντρο απόφασης χρησιμοποιεί ένα γράφημα τύπου δέντρου αποφάσεων που μοιάζει με flowchart και τα πιθανά αποτελέσματα τους. Είναι ένας τρόπος αποτύπωσης ενός αλγορίθμου που περιέχει μόνο υποθετικές δηλώσεις.
Σε ένα δέντρο απόφασης, κάθε εσωτερικός κόμβος αντιπροσωπεύει μια "δοκιμή" σε ένα χαρακτηριστικό, κάθε κλάδος αντιπροσωπεύει το αποτέλεσμα της δοκιμής, και κάθε κόμβος φύλλων αντιπροσωπεύει μια ετικέτα κλάσης. Οι διαδρομές από ρίζα σε φύλλο αντιπροσωπεύουν κανόνες ταξινόμησης.

Οι αλγόριθμοι μάθησης που βασίζονται σε δέντρα θεωρούνται μια από τις καλύτερες μεθόδους μάθησης με επίβλεψη που χρησιμοποιούνται, ως επί το πλείστον. Οι μέθοδοι που βασίζονται σε δέντρα ενισχύουν τα προγνωστικά μοντέλα με υψηλή ακρίβεια, σταθερότητα και ευκολία ερμηνείας. Σε αντίθεση με τα γραμμικά μοντέλα, χαρτογραφούν τις μη γραμμικές σχέσεις αρκετά καλά. Προσαρμόζονται στην επίλυση κάθε είδους προβλήματος που παρουσιάζεται (κατηγοριοποίηση ή παλινδρόμηση).

k-Nearest Neighbors:



Ο KNN Αλγόριθμος βασίζεται στην ομοιότητα των χαρακτηριστικών: Το πόσο στενά ένα δεδομένο εισόδου μοιάζει με το set μας καθορίζει τον τρόπο με τον οποίο το ταξινομούμε. Για την ταξινόμηση ενός αντικειμένου, ελέγχουμε μεταξύ των πλησιέστερων γειτόνων, την τάξη που εχει τα περισσότερα δείγματα.

Πιο συγκεκριμένα επιλέγουμε k δεδομένα από το dataset που είναι πιο κοντά στο νέο δείγμα και απο αυτα βρίσκουμε την κλάση που υπερισχύει.

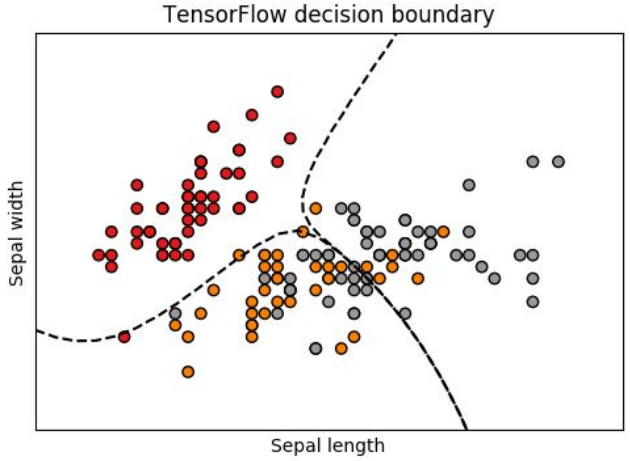
Ενα βασικό πλεονέκτημα αποτελεί η υψηλή ακρίβεια του αλγορίθμου, αν και δεν είναι ανταγωνιστική σε σύγκριση με καλύτερα ελεγχόμενα μοντέλα μάθησης. Επίσης είναι αρκετά ευέλικτος. Παρ' όλα αυτα είναι υπολογιστικά ακριβός, καθώς ο αλγόριθμος αποθηκεύει όλα τα δεδομένα εκπαίδευσης, και ως εκ τούτου έχει υψηλή απαίτηση μνήμης. Επιπλέον αποθηκεύει όλα (ή σχεδόν όλα) τα δεδομένα εκπαίδευσης και τα ανακαλεί σε κάθε εμφάνιση καινούργιου δεδομένου και ως εκ τούτου το στάδιο πρόβλεψης μπορεί να είναι αργό (με μεγάλο N).

Naïve Bayes

Bayes' Theorem: $P(h|d) = (P(d|h) * P(h)) / P(d)$

Όπου

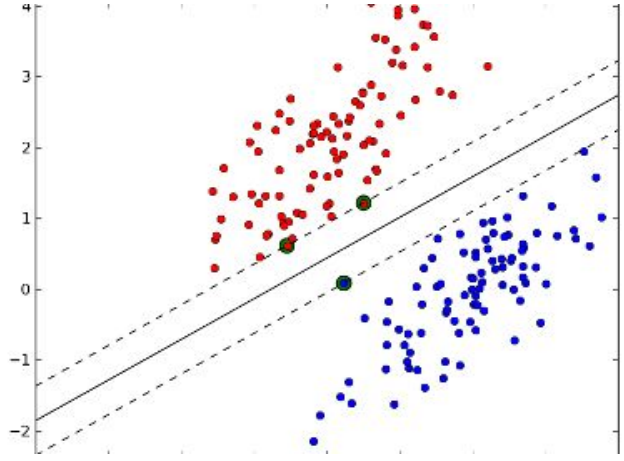
- P(h|d) είναι η πιθανότητα του h δεδομένου του d.
- P(d|h) είναι η πιθανότητα του d δεδομένου του h.
- P(h) είναι η πιθανότητα το h να είναι αληθής ασχέτως των δεδομένων.
- P(d) είναι η πιθανότητα το d να είναι αληθής ασχέτως των δεδομένων.



Ένας κατηγοριοποιητής Naive Bayes είναι ένα πιθανοτικό μοντέλο μηχανικής μάθησης που χρησιμοποιείται για κατηγοριοποίηση και βασίζεται στο θεώρημα Bayes. Χρησιμοποιώντας το θεώρημα Bayes, μπορούμε να βρούμε την πιθανότητα να συμβεί το A, δεδομένου ότι έχει συμβεί το B. Κάνουμε την παραδοχή πως οι παράγοντες/ χαρακτηριστικά είναι ανεξάρτητα, άρα η παρουσία ενός συγκεκριμένου χαρακτηριστικού δεν επηρεάζει το άλλο. Μια άλλη υπόθεση, είναι ότι όλοι οι παράγοντες έχουν ίση επίδραση στο αποτέλεσμα. Υπάρχουν 3 είδη όπως αναλύονται παρακάτω:

Multinomial Naive Bayes: Χρησιμοποιείται κυρίως για το πρόβλημα της ταξινόμησης των εγγράφων, δηλ. Αν ένα έγγραφο ανήκει στην κατηγορία του αθλητισμού, της πολιτικής, της τεχνολογίας κλπ. Τα χαρακτηριστικά /παράγοντες που χρησιμοποιούνται από τον ταξινομητή είναι η συχνότητα των λέξεων που υπάρχουν στο έγγραφο.
Bernoulli Naive Bayes: Παρόμοιο με το multinomial naive bayes αλλά οι παράγοντες είναι boolean μεταβλητές. Οι παράμετροι που χρησιμοποιούμε για την πρόβλεψη της κλάσης παίρνουν μόνο τιμές ναι ή όχι, για παράδειγμα, αν υπάρχει μια λέξη στο κείμενο ή όχι.
Gaussian Naive Bayes:Όταν οι προγνωστικοί παίρνουν μια συνεχή τιμή και δεν είναι διακριτοί, υποθέτουμε ότι αυτές οι τιμές προκύπτουν από μια Gaussian κατανομή. Οι αλγόριθμοι Naive Bayes χρησιμοποιούνται ως επί το πλείστον στην ανάλυση συναισθημάτων, το φιλτράρισμα spam μηνυμάτων, τα συστήματα προτάσεων κ.λπ. Είναι ταχύτατα και εύκολα στην εφαρμογή τους, αλλά το μεγαλύτερο μειονέκτημα τους είναι η απαίτηση των προγνωστικών να είναι ανεξάρτητες. Στις περισσότερες ρεαλιστικές περιπτώσεις, τα χαρακτηριστικά είναι εξαρτημένα, άρα μειώνεται η απόδοση του ταξινομητή.

Support Vector Machines

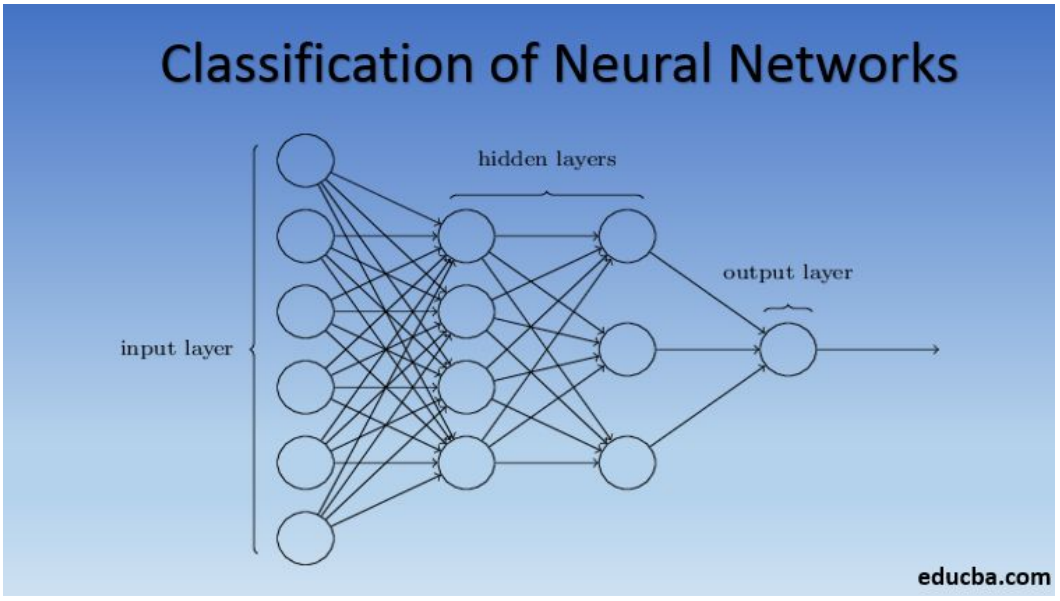


Ο στόχος του αλγόριθμου SVM είναι να βρεθεί ένα υπερεπιπεδο σε ένα χώρο Ν-διαστάσεων που διαχωρίζει ευκρινώς τα σημεία δεδομένων. Για να διαχωρίσουμε τις δύο κατηγορίες, υπάρχουν πολλά πιθανά επίπεδα που θα μπορούσαν να επιλεγούν. Στόχος μας είναι να βρούμε εκείνο το επίπεδο που έχει το μέγιστο περιθώριο, δηλαδή τη μέγιστη απόσταση μεταξύ των σημείων των δύο κατηγοριών. Η μεγιστοποίηση της απόστασης περιθωρίου παρέχει αντοχή στο μοντέλο ώστε μελλοντικά σημεία δεδομένων να μπορούν να ταξινομηθούν με μεγαλύτερη ακρίβεια. Επίσης, η διάσταση του υπερεπιπεδου εξαρτάται από τον αριθμό των χαρακτηριστικών. Εάν ο αριθμός των χαρακτηριστικών εισόδου είναι 2, τότε το επίπεδο είναι μόνο μια γραμμή. Εάν ο αριθμός των χαρακτηριστικών εισόδου είναι 3,

τότε το υπερεπιπεδο γίνεται ένα δισδιάστατο επίπεδο. Οταν υπερβαίνει τις 3 διαστάσεις γίνεται δύσκολο να το φανταστεί κανείς.

Εξελίσσοντας τον αλγόριθμο μπορούμε να εφαρμόσουμε την ίδια λογική και στην περίπτωση που τα δεδομένα είναι εγγενής μη διαχωρίσιμα. Σε αυτή την περίπτωση αρκει να μετασχηματίσουμε τα δεδομένα μας σε ενα χωρο μεγαλύτερης διάστασης ώστε να είναι γραμμικά διαχωρισιμα. Για την εύρεση της συνάρτησης μετασχηματισμού των μεταβλητών μας βοηθάνε οι συναρτήσεις kernel.

Neural Networks



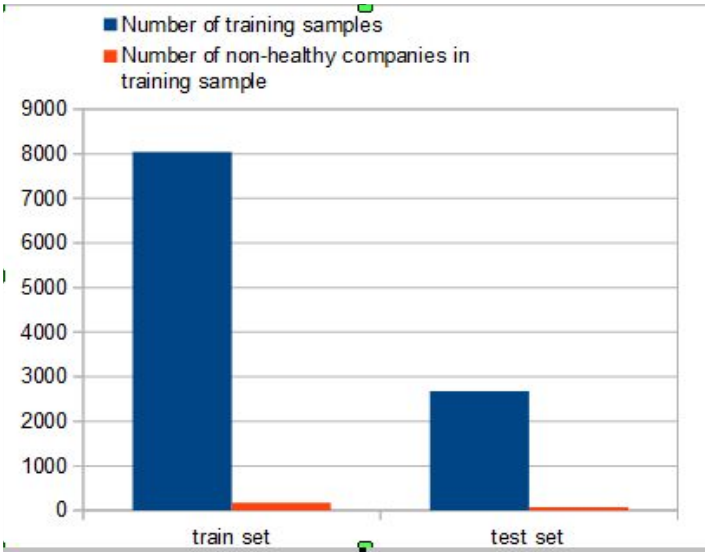
Υπάρχουν τρεις μέθοδοι μάθησης: η επιβλεπόμενη, η χωρίς επίβλεψη και η ενισχυτική μάθηση. Στην επιβλεπόμενη μάθηση, δίνονται στο νευρικό δίκτυο οι ετικέτες εισόδου και στη συνέχεια χρησιμοποιούνται για να παράγουν γενικευμένους κανόνες που μπορούν να εφαρμοστούν σε καινούργια δεδομένα. Είναι η απλούστερη μέθοδος εκμάθησης, αφού επιτρέπει στο δίκτυο να συγκρίνει τις προβλέψεις του με τα πραγματικά και επιθυμητά αποτελέσματα. Οι μη επιβλεπόμενες μέθοδοι δεν απαιτούν αρχικές εισόδους με ετικέτα δηλαδή δεν παρέχεται κάποια εμπειρία στον αλγόριθμο μάθησης αλλά πρέπει να βρει την δομή των δεδομένων εισόδου. ανακαλύπτοντας κρυμμένα μοτίβα στα δεδομένα. Στην δικιά μας εργασία χρησιμοποιούμε το πρώτο είδος νευρωνικών δικτύων.

Απλοποιημένα ένα νευρωνικό δίκτυο είναι ένα κλειστό κουτί που παίρνει εισόδους και φτάνει σε μια έξοδο. Οι ενδιάμεσες λειτουργίες, οι οποίες πραγματοποιούνται από τους διαστρωματωμένους νευρώνες, συνήθως δεν είναι εμφανής. Τα μαθηματικά πίσω από την σύνδεση των νευρώνων είναι τόσο ενδιαφέροντα όσο είναι πολύπλοκα, και συνήθως είναι αυτοματοποιημένα. Όπως προαναφέρθηκε, οι νευρώνες στο ένα στρώμα αλληλεπιδρούν με τους νευρώνες στο επόμενο στρώμα, με κάθε έξοδο να λειτουργεί ως είσοδος για την επόμενη. Κάθε συνάρτηση, συμπεριλαμβανομένου του αρχικού νευρώνα, λαμβάνει μια

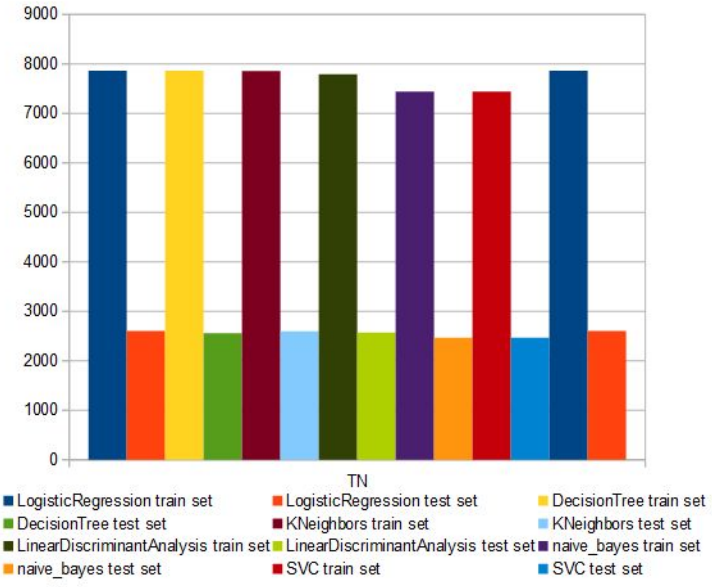
αριθμητική είσοδο και παράγει μια αριθμητική έξοδο βασισμένη σε μια εσωτερικοποιημένη λειτουργία, όπου κύριο μέρος της αφορά το βάρος του νευρώνα και το bias. Εν τέλη θα παραχθεί μια μοναδική έξοδος για το δίκτυο.

Η δυσκολία έγκειται στον προσδιορισμό της βέλτιστης τιμής bias για κάθε όρο, καθώς και στην εύρεση του βέλτιστου βάρους για κάθε κύκλο στο νευρικό δίκτυο. Ως εκ τούτου, επιστρατεύουμε τις συναρτήσεις κόστους, όπου εκεί υπολογίζουμε το πόσο κοντά μια συγκεκριμένη λύση είναι στην καλύτερη δυνατή λύση. Ένας απλός τρόπος βελτιστοποίησης των βαρών και του bias, είναι η απλή εκτέλεση του δικτύου αρκετές φορές. Στην πρώτη προσπάθεια, οι προβλέψεις θα είναι αναγκαστικά τυχαίες. Μετά από κάθε επανάληψη, με την βοήθεια της συνάρτηση κόστους, θα προσδιορίσουμε την επίδοση του μοντέλου και πώς αυτό μπορεί να βελτιωθεί. Οι πληροφορίες που αποκτήθηκαν από την συνάρτηση κόστους μεταβιβάζονται στην συνάρτηση βελτιστοποίησης, η οποία υπολογίζει νέες τιμές βάρους, καθώς και νέες τιμές bias, τις οποίες ενσωματώνουμε στο μοντέλο και επαναλαμβάνουμε την διαδικασία. Αυτό συνεχίζεται μέχρις ότου καμία αλλαγή να μην βελτιώσει την συνάρτηση κόστους.

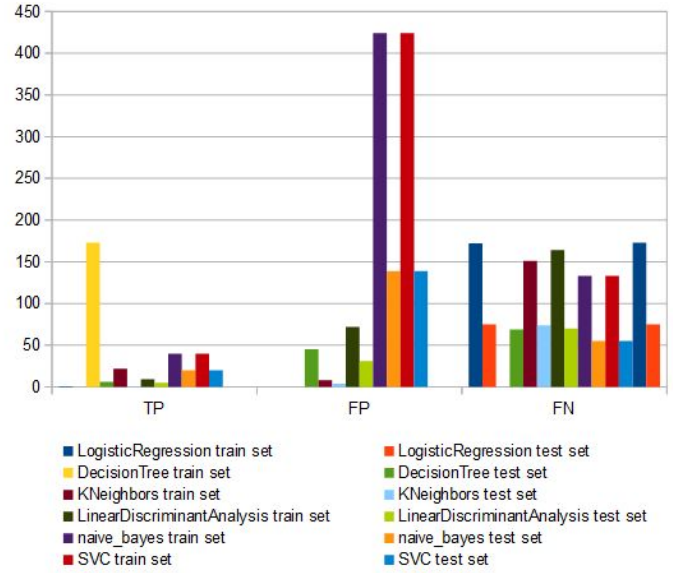
Αποτελέσματα μοντέλων:



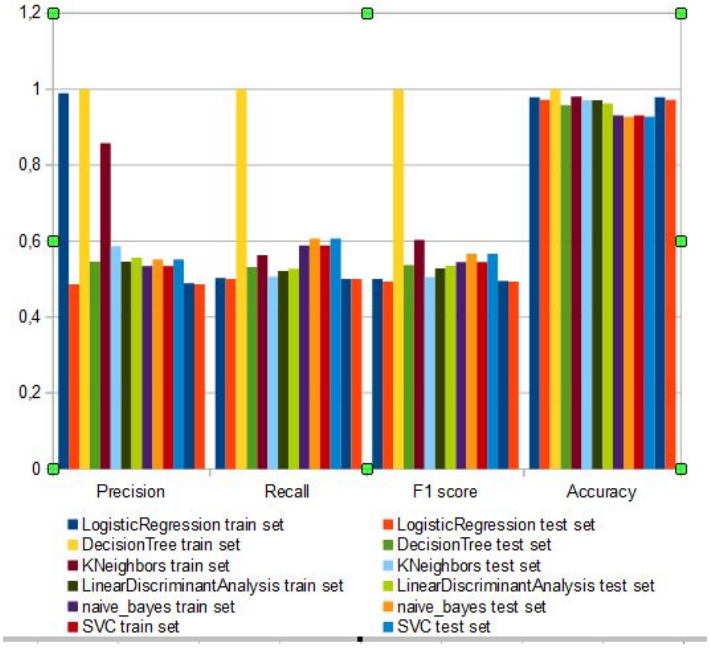
Εδώ παρατηρούμε την ανισορροπία που εμφανίζεται στις δύο κλάσεις των δεδομένων. Ως εκ τούτου θα επηρεαστούν αρκετα αρνητικά τα μοντέλα μάθησης.



Όπως φαίνεται στο γράφημα τα tn(true negative: αληθώς αρνητικά, δηλαδή οι υγιεις επιχειρήσεις) είναι αρκετά υψηλά. Παρ' όλα αυτά αρκετά χαμηλά αποτελέσματα βλέπουμε στα test set, εξού και συνειδητοποιούμε ότι δεν μπόρεσε κανένα να γενικεύσει αρκετά.

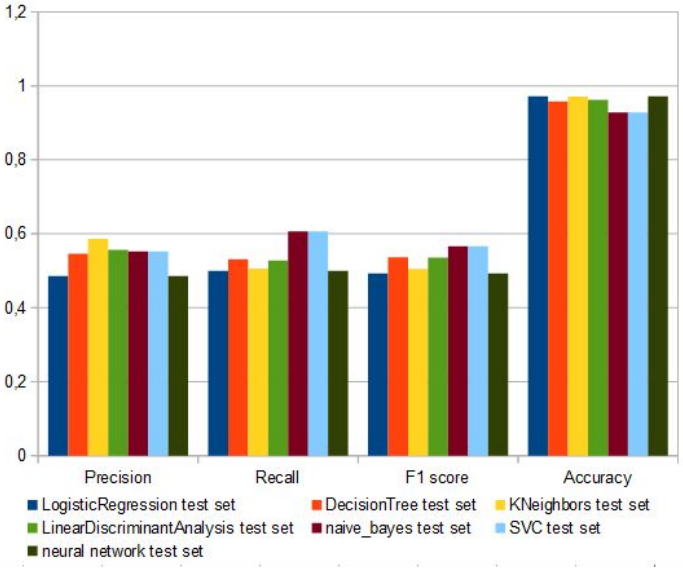


Όσον αφορά τα tp(true positive/μη υγιή) το decision tree έκανε υπερεκπαίδευση αλλά δεν γενίκευσε. Τα πιό πολλά βρήκαν τα naive bayes ,svc. Όμως βλέπουμε ότι δεν είναι αποδοτικοί ταξινομητές καθώς βρήκαν πολλά fp(false positive/ψευδώς μη υγιή).
Τα fn (false negative/ψευδώς υγιή) κυμάνθηκαν στα ίδια αποτελέσματα.



Precision=tp/(tp+fp)
Recall=tp/(tp+fn)
F1 score=(p*r)/(p+r)
Accuracy=(tp+tn)/(tp+tn+fp+fn)

Τα μοναδικά που διαφοροποιήθηκαν είναι όλα train test, με τέλεια αποτελέσματα το decision tree. Όπως προείπαμε δυστυχώς είναι αποτέλεσμα υπερ εκπαίδευσης χωρίς δυνατότητα γενίκευσης.



Όσον αφορά μόνο το test set, βλέπουμε από το γράφημα που προκύπτει από τον παρακάτω πίνακα ότι γενικά είχαν όλα παρόμοιες αποδόσεις. Εάν θα διάλεγα κάποιο μοντέλο θα ήταν το naive bayes ή το svc καθώς είχαν με μικρή διαφορά τα καλύτερα αποτελέσματα.

Classifier Name		training samples	non-healthy	TP	TN	FP	FN	Precision	Recall	F1 score	Accuracy
LogisticRegression	test set	2679	75	0	2604	0	75	0,48	0,5	0,49	0,97
DecisionTree	test set	2679	75	6	2559	45	69	0,54	0,53	0,53	0,95
KNeighbors	test set	2679	75	1	2600	4	74	0,58	0,5	0,5	0,97
LinearDiscriminantAnalysis	test set	2679	75	5	2573	31	70	0,55	0,52	0,53	0,96
naive_bayes	test set	2679	75	20	2465	139	55	0,55	0,6	0,56	0,92
SVC	test set	2679	75	20	2465	139	55	0,55	0,6	0,56	0,92
neural network	test set	2679	75	0	2604	0	75	0,48	0,5	0,49	0,97

Απόδοση:

Στην εργασία διευκρινίζεται ότι αποδεκτά αποτελέσματα είναι μόνο εκείνα που το μοντέλο βρίσκει με ποσοστό επιτυχίας τουλάχιστον 62% τις εταιρείες που θα πτωχεύσουν και με ποσοστό επιτυχίας τουλάχιστον 70% τις εταιρείες που δεν θα πτωχεύσουν. Ο πρώτος περιορισμός αφορά τις εταιρείες που εντόπισα ότι πτώχευσαν και όντων πτώχευσαν προς όλες όσες πτώχευσαν: tp/(tp+fn))=recall. Ο δεύτερος περιορισμός αφορά τις εταιρείες που εντόπισα ότι δεν πτώχευσαν και όντων δεν πτώχευσαν προς όλες όσες δεν πτώχευσαν=tn/(tn+fp)).

Τους παραπάνω περιορισμούς δεν καλύπτει κανένα μοντέλο.(όπως εκτυπώνει και το πρόγραμμα πριν τελειώσει.)

Downsampling:

Downsampling σημαίνει εκπαίδευση σε ένα μικρότερο υποσύνολο των παραδειγμάτων της τάξης πλειοψηφίας. Καθώς διαθέτουμε τόσο λίγα παραδείγματα που πτωχεύουν σε σχέση με τα υγιή, το μοντέλο εκπαίδευσης θα περάσει τον μεγαλύτερο μέρος του χρόνου του σε υγιή παραδείγματα και δεν θα μάθει αρκετά από αυτά που πτωχεύουν. Κατά τη διάρκεια της εκπαίδευσης, παρατηρούμε πιο συχνά την μειονεκτική τάξη, γεγονός που θα βοηθήσει το μοντέλο να συγκλίνει ταχύτερα. Με την μείωση των δειγμάτων της πλειοψηφούμενης τάξης ξοδεύουμε λιγότερο χώρο για την αποθήκευση τους, άρα περισσότερο χώρο στο δίσκο για τη μειονεκτική τάξη, ώστε να συλλέξουμε περισσότερα με ευρύτερο φάσμα παραδειγμάτων από αυτή την τάξη.

Επεξήγηση κώδικα:

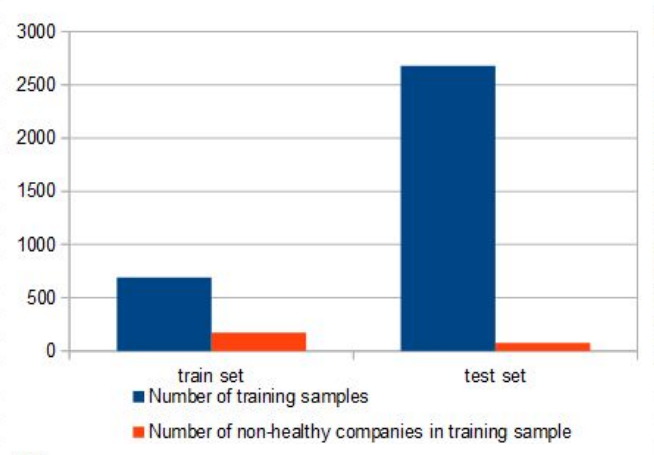
Η διαφορά του πρώτου από τον δεύτερο κώδικα εντοπίζεται στις γραμμές 114-140. Υπενθυμίζω μέχρι εκείνο το σημείο έχουμε διαβάσει από το αρχείο όλα τα παραδείγματα και τα έχουμε χωρίσει σε training είσοδο-έξοδο και σε test είσοδο-έξοδο.

Αρχικά θέλουμε να μετρήσουμε τα παραδείγματα που πτωχεύουν ώστε να υπολογίσουμε πόσα παραδείγματα Ν επιθυμούμε να διατηρήσουμε απο τα υγιή. Στην συνέχεια χωρίζουμε σε 2 λίστες τα παραπάνω δύο είδη καθώς θέλουμε να παραμείνουν όλα όσα πτωχεύουν. Απο τα υγιή, τυχαία επιλέγουμε τα Ν δείγματα όπως υπολογίσαμε πρίν.

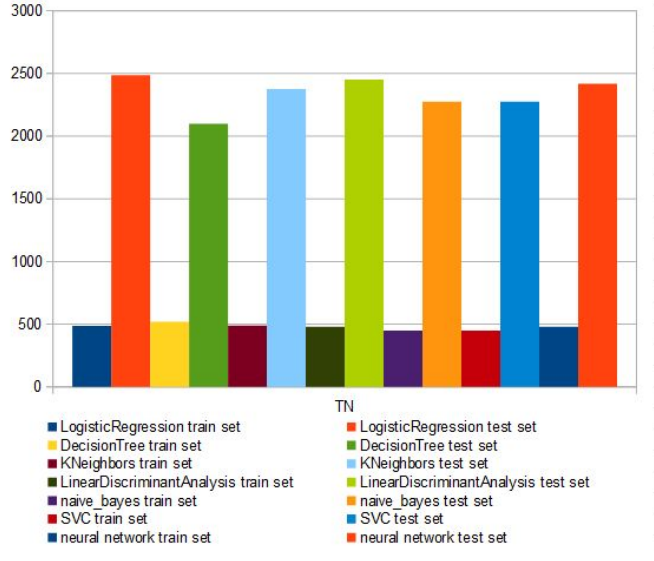
Παρακάτω, επιθυμούμε να ανακατέψουμε τα δύο είδη για το επόμενο βήμα της εκπαίδευσης. Για να το επιτύχουμε, ξαναενώνουμε την λίστα με τα πτωχεύσαντα και την καινούργια με τα υγιή. Καθώς όμως θέλουμε να διατηρήσουμε και την αντιστοιχία χ/τιμες χαρακτηριστικών- y/αποτέλεσμα δημιουργούμε μια λίστα με μηδενικά όσα τα υγιή και άσσους όσα τα μη υγιή. Ακολούθως τα ενώνουμε σε μια δομή zip που μας προσφέρει την δυνατότητα να διατηρήσουμε την αντιστοιχία και με την βοήθεια της shuffle, από τις λίστες, την ανακατεύουμε.

Εν τέλη ξαναχωρίζουμε την συγκεκριμένη δομή σε X_train, y_train και ακολουθούμε ακριβώς τα ίδια βήματα με τον προηγούμενο κώδικα.

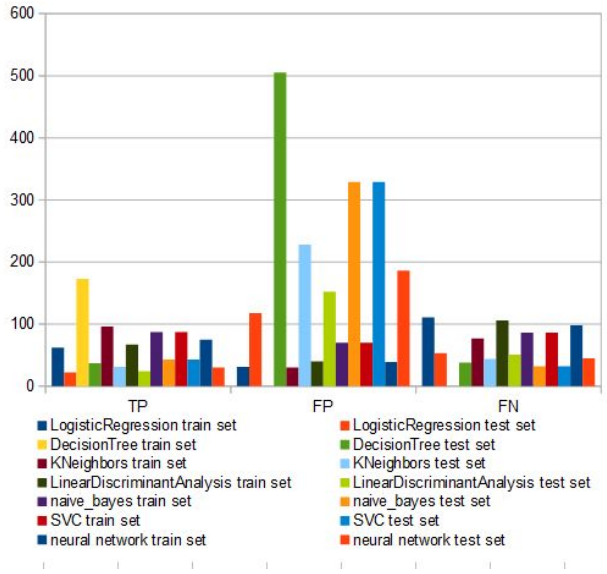
Αποτελέσματα μοντέλων:



Παρατηρούμε την ισορροπία που φέραμε στις δύο κλάσεις των δεδομένων 3 προς 1. Επίσης παρατηρούμε ότι το test set έχει πολλά περισσότερα από ότι το rain set καθώς το downsampling εφαρμόστηκε αποκλειστικά στο train set.

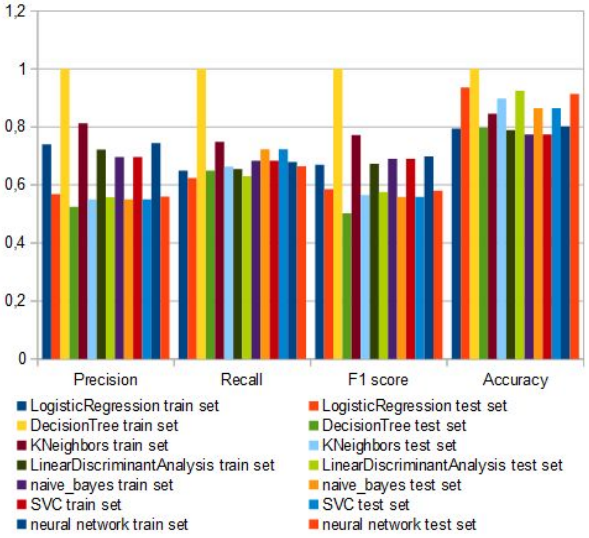


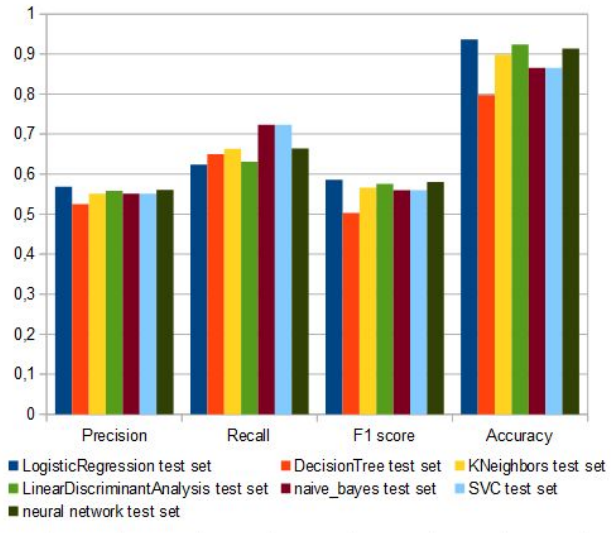
Όπως φαίνεται στο γράφημα τα tn(true negative: αληθώς αρνητικά, δηλαδή οι υγιεις επιχειρήσεις) είναι αρκετά υψηλά δηλαδή τα εντόπισαν σχεδόν όλα.



Αξίζει να σημειώσουμε ότι ο decision tree έκανε πάλι υπερεκπαίδευση(σε μικρότερο βαθμό) και δεν κατάφερε να γενικεύσει αρκετά, εξού και είχε τα περισσότερα fp(false positive/ ψευδώς μη υγιή). Αρκετά fp εμφανίζουν και οι naive bayes,svc. Τα fn (false negative/ψευδώς υγιή) κυμάνθηκαν στα ίδια αποτελέσματα.

Το μοναδικό που διαφοροποιήθηκε με τέλεια αποτελέσματα στο train είναι το decision tree αλλά χειρότερο από τα υπόλοιπα στο test.





Όσον αφορά μόνο το test set, βλέπουμε από το γράφημα ότι γενικά είχαν όλα παρόμοιες αποδόσεις. Εάν θα διάλεγα κάποιο μοντέλο θα ήταν το naive bayes ή το svc καθώς είχαν με μικρή διαφορά τα καλύτερα αποτελέσματα.

Απόδοση:

Στην δεύτερη έκδοση με τους ίδιους περιορισμούς τα μοντέλα εκείνα που κατάφεραν να ικανοποιήσουν τα δοθέντα ποσοστά είναι, όπως εμφανίζονται στην κονσόλα, 'Logistic Regression', 'Decision Tree', 'K Neighbors', 'Linear Discriminant Analysis', 'naive_bayes', 'SVC', 'neural network'. Δηλαδή όλα.

2ο Μέρος

B1.

➢ Αρχικά δημιουργούμε 3 vectors που αντιστοιχούν στα σημεία μας και στο τέλος προσθέτουμε ένα επιπλέον στοιχείο. Έτσι προκύπτουν οι:

S1=[0.5 1 1] S2=[1 0.5 1] S3=[1 1.5 1]

➢ Ακολούθους δημιουργούμε τις παρακάτω εξισώσεις όπου ορίζουμε την πρώτη κλάση(S1 S2) ως θετική και την δεύτερη κλάση(S3) ως αρνητική:

a*S1*S1+b*S2*S1+c*S3*S1=1
a*S1*S2+b*S2*S2+c*S3*S2=1
a*S1*S3+b*S2*S3+c*S3*S3=-1

➢ Λύνουμε τις εξισώσεις από όπου και προκύπτει:

a=12 b=2 c=-10

➢ Στη συνέχεια προσπαθούμε να βρούμε τον πίνακα W:

w=a*S1+b*S2+c*S3=[-2 -2 4]

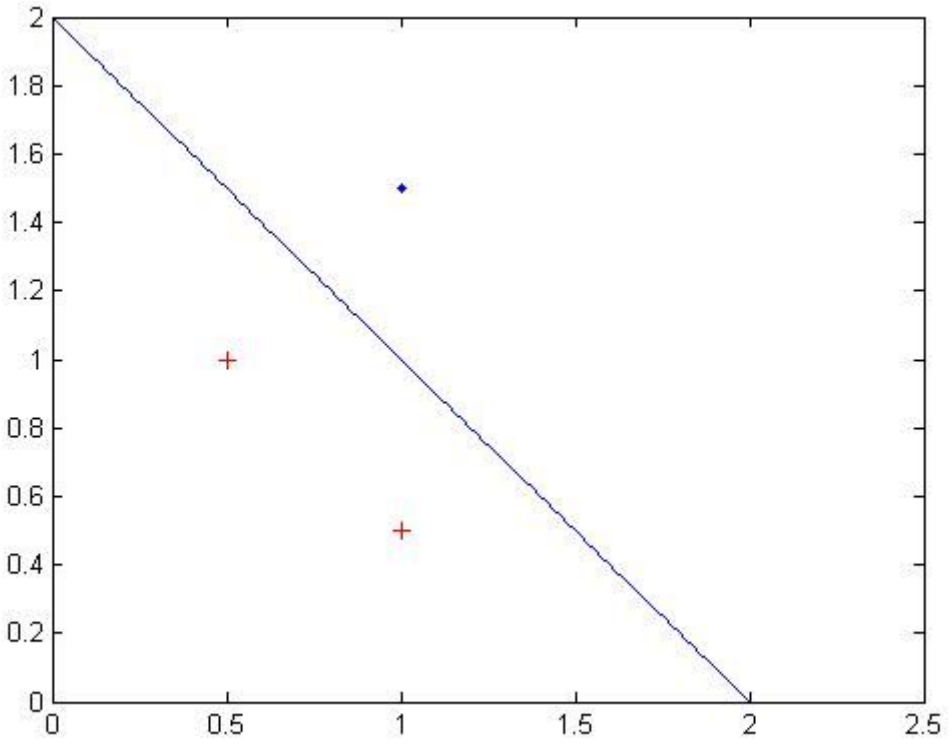
➢ Η τελική εξίσωση προκύπτει ως εξής:

y=(-w1*x-w3)/w2=(2*x-4)/-2=-x+2=2-x

➢ Και για να σχεδιάσουμε την ευθεία βρίσκουμε τα δύο παρακάτω σημεία:

x=0 → y=0+2=2 →(0,2)
y=0 → 0=-x+2→ x=2 →(2,0)

➢ Ετσι προκύπτει η παρακάτω γραφική παράσταση:



B2.

$(\langle X*Z\rangle+\theta)^d = (\langle X*Z\rangle+1)^2 = (X1*Z1+X2*Z2+1)^2 = X1^2*Z1^2 + X1*X2*Z1*Z2 + X1*Z1+X1*X2*Z1*Z2 + X2^2*Z2^2 + X2*Z2+X1*Z1+X2*Z2+1=$

$X1^2*Z1^2 + X2^2*Z2^2 + 2*X1*Z1 + 2*X2*Z2 + 2*X1*X2*Z1*Z2 + 1=$

$Y1^2 \quad Y2^2 \quad \sqrt{2}*Y1 \quad \sqrt{2}*Y2 \quad \sqrt{2}*Y1*Y2 \quad 1$

$=\langle(1, \sqrt{2}*X1, \sqrt{2}*X2, X1^2, X2^2, \sqrt{2}*X1*X2)*(1, \sqrt{2}*Z1, \sqrt{2}*Z2, Z1^2, Z2^2, \sqrt{2}*Z1*Z2)\rangle$

$=\langle\Phi(X)*\Phi(Z)\rangle$

B3.

Στην περίπτωση που εμφανίζεται θόρυβος, τότε υλοποιούμε soft-margin SVM, όπου το υπερεπίπεδο που ψάχνουμε ανέχεται σε κάποιο βαθμό τα λανθασμένα σημεία. Για κάθε τέτοιο σημείο ζημιονόμαστε με ένα κόστος που εξαρτάται από την απόσταση του σε σχέση με τό όριο της κλάσης του.

Η βασική ιδέα είναι να μεταφέρουμε τα μη γραμμικώς διαχωρίσιμα δεδομένα, σε έναν χώρο με περισσότερες διαστάσεις, καθώς έτσι μας δίνετε τη δυνατότητα να τα κάνουμε γραμμικώς διαχωρίσιμα. Για να το πραγματοποιήσουμε αυτό, αντιστοιχίζουμε τα δεδομένα εισόδου από τον χώρο X σε έναν χώρο F μέσω μιας μη γραμμικής συνάρτησης φ, με την βοήθεια των συναρτήσεων kernel. Κατά αυτόν τον τρόπο, μετασχηματίζουμε τα δεδομένα εισόδου σε έναν άλλο χώρο, συνήθως μεγαλύτερης διάστασης, έτσι ώστε να εντοπίσουμε εκείνο το γραμμικό υπερεπίπεδο που θα μπορέσει να οριοθετήσει και να διαχωρήσει τα θετικά από τα αρνητικά δείγματα στον μετασχηματισμένο χώρο. Εφόσον έχουμε πραγματοποιήσει τον μετασχηματισμό στη συνέχεια μπορούμε να χρησιμοποιήσουμε έναν γραμμικό classifier.