Mnist Dataset

Of Handwritten digits

Χριστίνα Ισακογλου - 2056
christci@csd.auth.gr

Τροπος υλοποιησης

Η μηχανή SVM που επιλύει το πρόβλημα ζυγών-περιττών αριθμών στη βάση δεδομένων της MNIST[6] εκπαιδεύτηκε με τη χρήση των βιβλιοθηκών LibSVM[1] και LibLINEAR[2] (κατά κύριο λόγο των συναρτήσεων της σε $C/C++$ και κάποιων Python bindings που επίσης περιέχει). Ερευνήθηκαν διαφορετικές παράμετροι των συναρτήσεων που εμπεριέχονται σε αυτές και σημειώθηκαν τα αποτελέσματα από τα διαφορετικά πειράματα, η απόδοση αυτών και κάποια συμπεράσματα που προκύπτουν από τη μελέτη τους.

Η δημιουργία των κατάλληλων αρχείων για την εφαρμογή των συναρτήσεων της libsvm γίνεται σε δύο φάσεις. Αρχικά, με το python script $read\_mnist\_images.py$ διαβάζονται οι εικόνες, ενώ στη συνέχεια με το $write\_libsvm\_format.py$ δημιουργούνται δύο νέα αρχεία, ένα για το training set και ένα για το test, όπου περιέχουν το καθένα τα labels και τα διανύσματα χαρακτηριστικών των εικόνων του εκάστοτε σετ. Τα αρχεία θα πρέπει να έχουν την εξής μορφή :

$$< label >< index1 >:< value1 >< index2 >:< value2 > \cdots$$

Και επειδή το πρόβλημα είναι δϋαδικό η μορφή αυτή προσαρμόζεται σε :

$$+1 < index1 >:< value1 >< index2 >:< value2 > \cdots ,$$

όταν η εικόνα αναπαριστά ζυγό αριθμό και

$$-1 < index1 >:< value1 >< index2 >:< value2 > \cdots ,$$

όταν η εικόνα αναπαριστά περιττό αριθμό.

Μαθηματικο υποβαθρο

Η εκπαίδευση μιας μηχανής SVM βασίζεται στην εύρεση του καταλληλότερου διαχωριστικού επιπέδου $\mathbf{w}$ ανάμεσα σε δύο κλάσεις, επιλύοντας με τετραγωνικό προγραμματισμό το εξής πρόβλημα βελτιστοποίησης:

$$J(\mathbf{w}, w_0) = \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_{i=1}^{P} \xi_i$$

υπό τους περιορισμούς

$$d_i(\mathbf{w}^\tau \mathbf{x}_i + w_0) \geq 1 - \xi_i$$

και

$$\xi_i \geq 0,$$

όπου $i = 1, \cdots, P$.

Τα x_i αποτελούν τα δείγματα εκπαίδευσης, τα οποία όπως θα φανεί μπορούν να αντικατασταθούν από δείγματα σε υψηλότερη διάσταση χάρη σε κάποια συνάρτηση πυρήνα. Τα xi_i αποτελούν τους συντελεστές χαλαρότητας, καθένας από τους οποίος αν είναι μεγαλύτερος από ένα θα σημαίνει ότι το αντίστοιχο δείγμα ταξινομείται στη λάθος κλάση. Η παράμετρος C αποτελεί το κόστος της εσφαλμένης ταξινόμησης και το P αποτελεί το πλήθος των προτύπων.

Περιγραφη διαδικασιας

Τα βήματα που έχει παρατηρηθεί να οδηγούν σε καλύτερα αποτελέσματα[5] και ακολουθήθηκαν γι΄ αυτό το λόγο και στη δική μας περίπτωση είναι τα εξής :

1. Κανονικοποίηση των δεδομένων.
2. Αναζήτηση των καταλληλότερων παραμέτρων με την τεχνική cross-validation (σε ένα υποσύνολο του σετ δεδομένων εκπαίδευσης).
3. Εκπαίδευση της μηχανής με τις καλύτερες παραμετρους που βρέθηκαν στο προηγούμενο βήμα.
4. Έλεγχος στο σετ δεδομένων τεστ, για τη μέτρηση της ακρίβειας της μηχανής σε δεομένα άγνωστα και ανεξάρτητα.

Οι εντολές που υλοποιούν τη συνοψισμένη παραπάνα διαδικασία παρατίθεται παρακάτω(βρίσκοντα συγκεντρωτικά στο bash script $run\_svm.sh$), ενώ στη συνέχεια περιγράφονται πιο αναλυτικά ο τρόπος που υλοποιούνται και οι λόγοι που μας οδηγούν στο συγκεκριμένο τρόπο υλοποίησης.

```
$ ./svm-scale -l 0 -s tdf_sp training_data_full > training_data_full_scaled
$ ./svm-scale -r tdf_sp testing_data_full > testing_data_full_scaled

$ python subset.py training_data_full_scaled 5000 tdfs_subset
$ python grid.py -gnuplot /usr/local/bin/gnuplot tdfs_subset

$ ./svm-train -c 2.0 -g 0.03125 training_data_full_scaled

$ ./svm-predict testing_data_full_scaled training_data_full_scaled.model
    training_data_full_scaled.predict
```

i. Κανονικοποιηση

Για λόγους ακρίβειας και ταχύτητας, προτού γίνει η εκπαίδευση της μηχανής, αναγκαίο βήμα είναι η κανονικοποίηση των διανυσμάτων χαρακτηριστικών.

Με βάση την LibSVM η κανονικοποίηση που υπολοποιείται βασίζεται στην φόρμουλα :

$$normlized\_value = \frac{value - min\_value}{max\_value - min\_value},$$

η οποία κανονικοποιεί τα δεδομένα σε εύρος [0,+1](ο συγκεκριμένος τρόπος κανονικοποίησης είναι ένας από τους διάφορους που υπάρχουν και ονομάζεται rescaling) και όπου υπάρχουν μηδενικά τα

εξαλείφει από το τελικό σετ δεδομένων[1]. Με κατάλληλες παραμέτρους (πχ -l -1)ο χρήστης μπορεί να καλέσει την svm-scale προκειμένου να πετύχει κανονικοποίηση σε διαφορετικό εύρος (πχ [-1 +1]). Στην περίπτωση μας, καθώς τα δεδομένα κινούνται εξ αρχής στο εύρος [0 255] προτιμήθηκε το εύρος [0 +1].

Σε περίπτωση που το βήμα αυτό αγνοηθεί το πείραμα μας διατρέχει δύο κινδύνους. Αφενός υπάρχει η περίπτωση χαρακτηριστικά με μεγάλη τιμή να θεωρηθούν πιο σημαντικά από αυτά με μικρότερη και έτσι η τιμή ενός χαρακτηριστικού να αποτελεί ένδειξη της σημασίας του. Αυτό που συνεπάγεται από το παραπάνω είναι δύο ψηφία σε γενικές γραμμές παρόμοια όπως πχ το 8 και το 9 να κανονικοποιούνται σε ένα, καθώς η λεπτή διαφορά μεταξύ τους εξαλείφεται.

Αφετέρου, ένα άλλο πρόβλημα της μη κανονικοποίησης των δεδομένων, το οποίο συναντάται κυρίως σε προβλήματα μεγάλων διαστάσεων όπως αυτό που επιλύουμε, είναι τα πολύ μεγάλα νούμερα που μπορούν να προκύψουν κατά τον υπολογισμό των εσωτερικών γινομένων των συναρτήσεων του πυρήνα. Τα πολύ μεγάλα νούμερα με τη σειρά τους επιφέρουν δυσκολίες στον υπολογισμό και αριθμητιμή αστάθεια (numerical instability).

ii. Cross-validation(grid-search)

Στο επόμενο βήμα σειρά έχει η τεχνική cross-validation η οποία έχει σκοπό όχι μόνο την εκπαίδευση της μηχανής ούτως ώστε να μπορεί να διαχωρίζει όσο το δυνατόν καλύτερα τα δείγματα των οποίων τα labels γνωρίζει ήδη, αλλά και τη γενίκευση. Αυτό θα έχει ως αποτέλεσμα να μας εγγυάται με κάποιο τρόπο ότι θα τα πάει καλά και σε πρότυπα τα οποία δεν γνωρίζει.

Η μέθοδος αυτή υλοποιείται με το Python script grid.py και αυτό που κάνει είναι να χωρίζει διαδοχικά σε τόσες ομάδες όσες θα οριστούν με την παράμετρο v τα δεδομένα εκπαίδευσης, θεωρώντας κάθε φορά το ένα από αυτά σετ ελέγχου και τα υπόλοιπα σετ εκπαίδευσης.

Λόγω του ότι αυτή η διαδικασία απαιτεί αρκετό χρόνο(καθώς αποτελεί εξαντλητική μέθοδο αναζήτησης) αυτό που μπορεί να γίνει είναι να εξαχθεί αρχικά ένα υποσύνολο του σετ εκπαίδευσησ(με ίσο αριθμό στοιχείων κάθε κλάσης - το default της subset.py), να βρεθούν οι καλύτερες παραμέτρους με βάση αυτό και στη συνέχεια είτε να ακολουθήσει η εκπαίδευση της μηχανής με βάση αυτές είτε να ξαναγίνει αναζήτηση παραμέτρων(σε μεγαλύτερο υποσύνολο αυτή τη φορά ή και σε όλο) αλλά με πιο μικρό βήμα αύξησης των παραμέτρων κατά την αναζήτηση, η οποία θα είναι εστιασμένη στη γειτονιά των καλύτερων παραμέτρων που βρέθηκαν προηγουμένως.

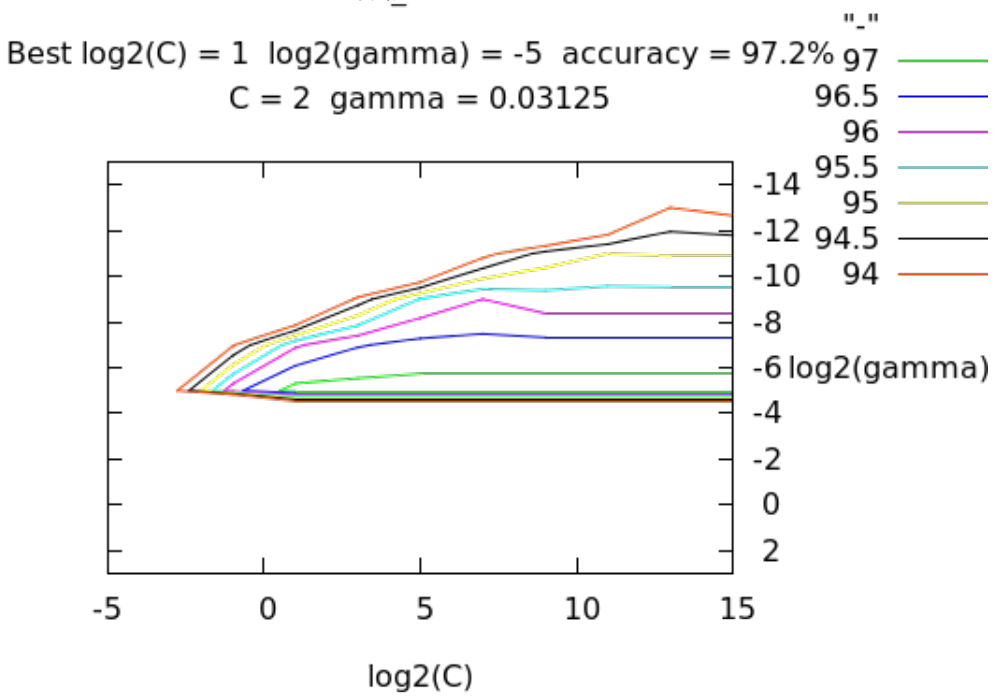
Εφαρμόζοντας 5-fold cross-validation σε ένα υποσύνολο των 5000 δειγμάτων του σετ εκπαίδευσης προκύπτει, ύστερα από 8 ώρες υπολογισμών, ότι οι καλύτερες παράμετροι για εκπαίδευση είναι οι 2.0 για την παράμετρο C και 0.03125 για την παράμετρο γ ενός RBF πυρήνα, καθώς επιτυγχάνεται ακρίβεια με την προαναφερθείσα τεχνική 97.2%. Το Σχήμα 1 αναπαριστά τη μεταβολή της ακρίβειας σε σχέση με τη μεταβολή των παραμέτρων.

Στη συνέχεια οι ίδιες παράμετροι χρησιμοποίηθηκαν για την εκπαίδευση της μηχανής και ήταν αυτές που οδήγησαν στην καλύτερη απόδοση κατά τη φάση του ελέγχου με ακρίβεια 99.1% κατασκευάζοντας ένα μοντέλο(αποθηκευμένο στο αρχείο με όνομα training_data_full _scaled.model) με μόλις 8657 διανύσματα εκπαίδευσης, δηλαδή το 14.43%του συνόλου εκπαίδευσης, αριθμός μικρός συγκριτικά με μοντέλα που θα παρατεθούν παρακάτω.

Στο σχήμα 2 παρατίθενται κάποια χαρακτηριστικά ψηφία από τα 90 που η μηχανή απέτυχε να κατατάξει σωστά σε άρτιους και περιττούς. Κάποια από αυτά, όπως για παράδειγμα το 2 της εικόνας δεν είναι ξεκάθαρο και στο ανθρώπινο μάτι αν είναι σίγουρα 2 ή 1.

[1]Στην πραγματικότητα η LibSVM κάνει χρήση της μορφής αραιού πίνακα (sparse), καθώς όπου για παράδειγμα τα δεδομένα έχουν τη μορφή 1 0 2 0, τα αποθηκεύει με τη μορφή 1:1 3:2.

Σχήμα 1: grid.py plot



III. Εκπαιδευση και πυρηνες

Γκαουσιανός πυρήνας

Το πρωτο είδος πυρήνα που προτείνεται για δεδομένα μη γραμμικής φύσεως είναι ο γκαουσιανός (Gaussian Radial Basis Function), και αυτό γιατί ουσιαστικά έχει την ιδιότητα να υψώνει τα δεδομένα σε μια μεγαλύτερη διάσταση(ή ακόμα και στην άπειρη), στην οποία προκύπτει πως οι κλάσεις των δεδομένων αποκτούν γραμμική διαχωρισιμότητα. Οι παράμετροι που χρειάζονται για την εκπαίδευση με το συγκεκριμένου πυρήνα είναι δύο: η παράμετρος C, που εκφράζει το κόστος της εσφαλμένης ταξινόμησης και η γ, που αποτελεί ένα μέτρο του πόσο "απλωμένη"θέλουμε να είναι η κατανομή μας.

Πραγματοποίηθηκαν τα παρακάτω πειράματα με διαφρετικούς συνδυασμούς παραμέτρων και τα αποτελέσματα που είχαν ήταν τα εξής: Μικρότερη απόδοση σημειώσε η μηχανή με μικρότερο γ και C, ενώ ταυτόχρονα αποτελούσε τη μηχανή με τα περισσότερα διανύσματα εκπαίδευσης. Μεγα-

Σχήμα 2: Ψηφία που η μηχανή κατατάσσει εσφαλμένα(Από τα αριστερά 9,9,3,2,0)

Πίνακας 1: Εκπαίδευση με γκαουσιανό πυρήνα

C	γ	Σωστή ταξινόμηση στα δείγματα ελέγχου	$\#SV$	Time(real)
2	0.001	94.91%	15654	26m
2	0.01	98.73%	6328	20m
2	0.02627	98.02%	7677	25m
2	0.03125	99.1%	8675	25m
2	0.07	98.78%	21973	2h25m
0.1	0.001	88.82%	25915	47m

λύτερο χρόνο εκπαίδευσης χρειάστηκε η μηχανή με περισσότερο απλωμένη κατανομή. Βλέπουμε ότι εκπαίδευση με μικρότερο γ οδηγεί σε μεγαλύτερο αριθμό διανυσμάτων υποστήριξης, κάτι που οδηγεί σε χαμηλότερη ακρίβεια στη φάση ελέγχου καθώς το μοντέλο εκπαίδευσης μαθαίνει να περιγράφει πολύ καλά τα διανύσματα χαρακτηριστικών που ήδη γνωρίζει αλλά αστοχεί σε αυτά που δε συναντά πρώτη φορά.

Γραμμικός πυρήνας

Στην περίπτωση που ο αριθμός των χαρακτηριστικών είναι ήδη μεγάλος η διαδικασία της ύψωσης των δεδομένων σε μια ανώτερη διάσταση δεν είναι πάντα απαραίτητη. Ένας γραμμικός πυρήνας μπορεί να αποδειχτεί εξίσου καλός με έναν μη γραμμικό με το πλεονέκτημα της γρηγορότερης εκπαίδευσης και της αναζήτησης μόνο μιας παραμέτρου, της C, δηλαδή του κόστους της λανθασμένης ταξινόμησης. Στη συνέχεια θα γίνει φανερό ότι καθώς η διάσταση των δεδομένων μας δεν είναι υπερβολικά μεγάλη, ο γραμμικός πυρήνας δεν μπορεί να φτάσει σε απόδοση αυτήν του γκαουσιανού. Ως αντιστάθμισμα όμως παρέχει έναν τρόπο γρήγορης εκπαίδευσης στα δεδομένα μας με μια σχετικά καλή ακρίβεια.

Αρχικά επισημαίνεται ότι στην περίπτωση αυτού του πυρήνα χρησιμοποιήθηκε η βιβλιοθήκη LibLINEAR αντί για την LibSVM καθώς η τελευταία φάνηκε πως σε περίπτωση που τα χαρακτηριστικά είναι κατά πολύ μικρότερα από τα δείγματα δεν αποδίδει ικανοποιητικά(πείραμα με την LibSVM στα δεδομένα της MNIST δεν είχε καταφέρει να καταλήξει σε κάποια εκπαίδευση ακόμα και μετά το πέρας τεσσάρων όρων εξαιτίας νουμερικών ασταθειών). Στη συνέχεια παρατίθονται δύο παραδείγματα εκπαίδευσης που αναδεικνύουν την επιτάχυνση που μπορεί να προσφέρει η εκπαίδευση με παράμετρο -s 2 και αιτιολογούν με τον τρόπο αυτό τη χρήση της στη συνέχεια.

```
$ time ./train -e 0.1 -c 4 -v 5 training_data_full_scaled
Cross Validation Accuracy = 88.9983%
real    8m57.203s

$  time ./train -e 0.1 -c 4 -v 5 -s 2 training_data_full_scaled
Cross Validation Accuracy = 89.255%
real    0m28.847s
```

Στη συνέχεια παρατίθενται τα αποτελέσματα αρκετών πειραμάτων με γραμμικό πυρήνα που έγιναν με σκοπό τη μελέτη της επίδρασης του κόστους και του συντελεστή χαλαρότητας στην εκπαίδευση. Η εντολή που υλοποιεί τη συγκεκριμένη εκπαίδευση είναι η :

```
$ ./train -e 0.01 -c 0.2 -v 5 -s 2 training_data_full_scaled
```

αλλάζοντας κάθε φορά την παράμετρο c και e αντίστοιχα.

Το C στις παραπάνω εντολές αποτελεί το βάρος του κόστους των λάθος ταξινομήσεων. Ό-

Πίνακας 2: Εκπαίδευση με γραμμικό πυρήνα και παραλλαγές μόνο στο κόστος εσφαλμένης ταξινόμησης

C	e	Cross validation accuracy	Time(real)
0.2	0.01	89.5233%	0m57.534s
1	0.01	89.4883%	0m57.671s
2	0.01	89.4883%	1m18.636s
5	0.01	89.5133%	1m2.912s
10	0.01	89.495%	0m59.642s
50	0.01	89.5117%	1m12.424s
140	0.01	89.5233%	1m2.080s
200	0.01	89.5133%	1m0.933s
250	0.01	89.5167%	1m0.934s
300	0.01	89.5183%	1m1.816s

σο μεγαλύτερο είναι τόσο μεγαλύτερη και η σημασία που δίνουμε στην σωστή ταξινόμηση των προτύπων, ενώ αν είναι μικρό υποδηλώνει ότι οι λάθος ταξινομήσεις δεν μας ενδιαφέρουν αρκετά, με αποτέλεσμα οι παράμετροι χαλαρότητας να μη λαμβάνονται πολύ υπόψη μας. Από την άλλη αν είναι μηδέν οι παράμετροι αυτοί αγνοούνται πλήρως. Με βάση τις παραπάνω παραμετροποιημένες εκπαιδεύσεις αυτό που μπορεί να παρατηρήσει κανείς είναι ότι η διαδοχική αύξηση του C οδηγεί σε μια αυξομείωση της ακρίβειας των αποτελεσμάτων με κάποια περιοδικότητα, με μειονέκτημα αυτό της επιβράδυνσης της εκπαίδευσης.

Πίνακας 3: Εκπαίδευση με γραμμικό πυρήνα και παραλλαγές στο συντελεστή χαλάρωσης (tolerance of termination) σε συνδυασμό με παραμέτρους που χρησιμοποιήθηκαν στον πίνακα 1

C	e	Cross validation accuracy	Time(real)
2	0.001	89.5083%	4m31.667s
10	0.001	89.51%	5m7.928s
50	0.001	89.5%	5m7.928s

Αρχικά αυτό που παρατηρείται είναι ότι η μεταβολή της παραμέτρου e δεν μπορεί να επιφέρει ουσιαστική βελτίωση στην εκπαίδευση, παρά μονάχα να επιβραδύνει τη διαδικασία. Το ίδιο θα ισχυριζόταν κάποιος για την παράμετρο c, καθώς παρά την τεράστια αύξηση στην οποία την υποβάλλουμε η ακρίβεια παραμένει στα ίδια επίπεδα, παρουσιάζοντας κάποια περιοδική αυξομείωση μικρού βαθμού. Η μηχανή μας δηλαδή δεν είναι σε θέση να πιάσει καλύτερα νούμερα σε αυτό το πρόβλημα μη γραμμικής φύσεως.

Ενδιαφέρον όμως σε σχέση με την παράμετρο κόστους παρουσιάζει το εξής πείραμα :

```
$ time ./train -e 0.01 -c 140 training_data_full_scaled
optimization finished, #iter = 1000
Objective value = -237446.280642
nSV = 40514
real    2m36.768s

$ time ./predict testing_data_full_scaled training_data_full_scaled.model
    training_data_full_scaled.predict
Accuracy = 78.52% (7852/10000)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
$ time ./train -e 0.01 -c 2 training_data_full_scaled
```

```
optimization finished, #iter = 1000
Objective value = -38468.455056
nSV = 32864
real    2m10.010s

$ time ./predict testing_data_full_scaled training_data_full_scaled.model
    training_data_full_scaled.predict
Accuracy = 89.75% (8975/10000)
real    0m1.174s
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
$ time ./train -e 0.01 -c 0.2 training_data_full_scaled
optimization finished, #iter = 594
Objective value = -3889.021922
nSV = 32192
real    1m6.563s

$ time ./predict testing_data_full_scaled training_data_full_scaled.model
    training_data_full_scaled.predict
Accuracy = 89.81% (8981/10000)
real    0m1.218s
```

Η μόνη παραμετροποίηση που επιδέχονται οι παραπάνω ταξινομητές έχει να κάνει με την τιμή του C. Αυτό που παρατηρείται είναι ότι σημειώνεται καλύτερη ακρίβεια στα δεδομένα τεστ όταν το βάρος του κόστους των ταξινομήσεων είναι μικρότερο. Πρακτικά η μείωση του βάρους σημαίνει ότι έχει λιγότερη επιρροή πάνω στο κόστος που ζητάμε να βελτιστοποιήσουμε και έτσι περισσότερα σημεία με συντελεστές χαλαρότητας $\xi_i > 0$ επιτρέπενται στο περιθώριο, αυξάνοντας το. Με άλλα λόγια, το πλάτος του περιθωρίου δεν εξαρτάται ολοκληρωτικά από την κατανομή των δεδομένων, όπως συνέβαινε στην περίπτωση των γραμμικώς διαχωρίσιμων κλάσεων, αλλά επηρεάζεται σημαντικά από την επιλογή του C[4]. Με βάση τα παραπάνω προκύπτει ότι ένα μεγάλο περιθώριο στο μη γραμμικά διαχωρίσιμο πρόβλημα που προσπαθούμε να επιλύσουμε οδηγεί σε καλύτερα συμπεράσματα.

Ακόμη μια παρατήρηση που μπορεί να γίνει έχει να κάνει με τον αριθμό διανυσμάτων υποστήριξης. Και στα τρία μοντέλα που κατασκευάστηκαν είδαμε στον πίνακα 2 πως η cross-validation ακρίβεια τους κυμαινόταν στο ίδιο επίπεδο. Αυτό μας οδήγησε στο συμπέρασμα πως το κόστος δεν επιδρά σημαντικά στην εκπαίδευση των μηχανών. Παρά το αληθές αυτής της πρότασης βλέπουμε πως στη φάση του ελέγχου η τιμή του κόστους καταλήγει σε διαφορετικά αποτελέσματα, με αυτά του μικρότερου κόστους να είναι καλύτερα από αυτά του μεγαλύτερου. Αυτό που συμβαίνει είναι ότι καθώς παραμετροποιούμε τη μηχανή μας με σκοπό η λάθος ταξινόμηση να έχει μεγάλη σημασία για αυτή, καταλήγει να χαρακτηρίζει περισσότερα πρότυπα ως διανύσματα υποστήριξης προκειμένου να πετύχει τη σωστή ταξινόμηση που επιθυμούμε. Ετσι για παράδειγμα έχουμε πως με κόστος 200 η μηχανή έχει 40514 διανύσματα υποστήριξης, το 67% δηλαδή του σετ εκπαίδευσης, ενώ με κόστος 0.2 έχει 32192, που αποτελεί το 53.65% του συνόλου. Ο ταξινομητής λοιπόν προκειμένου να μάθει να διαχωρίζει κλάσεις που από τη φύση τους είναι εναντίον στη γραμμικότητα του καταλήγει να υπερκπαιδευτεί επιφέροντας χειρότερα αποτελέσματα όταν το κόστος της εσφαλμένης ταξινόμησης επιθυμούμε να είναι μεγάλο.

Σύγκριση με knn

Πίνακας 4: Σύγκριση 1 και 3 nearest neighbour με rbf svm

Αλγόριθμος	Χρόνος(σε λεπτά)	Ποσοστό λάθους	Δείγματα ταξινομημένα σωστά
1-nearest	16	3.09%	9691
3-nearest	16	2.83%	9717
svm	25	0.9%	9910

Αναφορές

[1] Chih-Chung Chang and Chih-Jen Lin, *"LIBSVM : a library for support"*, Vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[2] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin, *"LIBLINEAR: A Library for Large Linear Classification"*, Journal of Machine Learning Research 9(2008), 1871-1874. Software available at http://www.csie.ntu.edu.tw/~cjlin/liblinear.

[3] Κ. Διαμαντάρας, *"Τεχνητά Νευρωνικά Δίκτυα"* , Κλειδάριθμος, Αθήνα, 2007.

[4] S. Theodoridis, K. Koutroumbas, *"Αναγνώριση Προτύπων"*, Εκδόσεις Π.Χ. Πασχαλίδης-Broken hill publishers LTD, 2012.

[5] C. W. Hsu, C. C. Chang, C. J. Lin, *"A practical guide to support vector classification"*, 2003.

[6] Yann Lecun, Corinna Cortes, The MNIST database of handwritten digits