Νευρωνικά Δίκτυα - Βαθιά Μάθηση
Δεύτερη Εργασία

13 Δεκεμβρίου 2023

Δημήτριος Αλεξόπουλος
aadimitri@ece.auth.gr
ΑΕΜ 10091

Περιεχόμενα

Κατάλογος Σχημάτων

1 Εισαγωγή

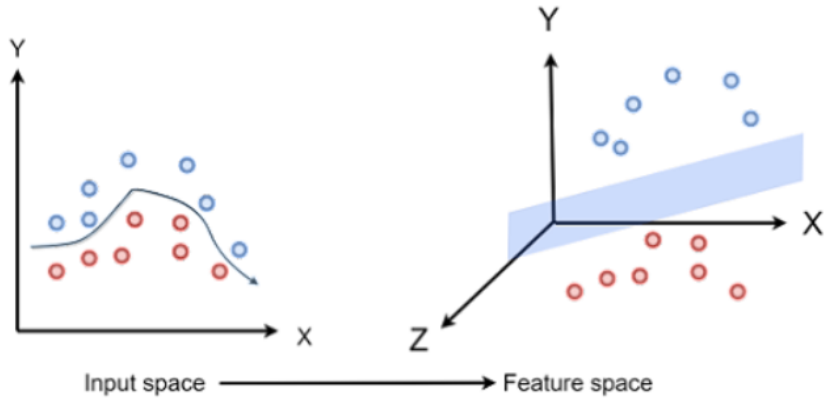
Σκοπός της παρούσας εργασίας είναι η υλοποίηση ενός Support Vector Machine που θα εκπαιδευτεί σε κάποιο dataset για να λύσει το πρόβλημα της κατηγοριοποίησης 2 μόνο κλάσεων. Η δομή του μπορεί να γενικευτεί και για προβλήματα κατηγοριοποίησης πολλών κλάσεων, ωστόσο είναι υπολογιστικά ακριβό.
Για τους σκοπούς της εργασίας επιλέγεται η βάση δεδομένων $CIFAR-10$, η οποία αποτελείται από 60000 έγχρωμες εικόνες 32×32 σε 10 κλάσεις, με 6000 εικόνες ανά κλάση. Υπάρχουν 50000 εικόνες εκπαίδευσης και 10000 εικόνες δοκιμής.
Το σύνολο δεδομένων χωρίζεται σε πέντε παρτίδες εκπαίδευσης και μία παρτίδα δοκιμής, κάθε μία με 10000 εικόνες. Η παρτίδα δοκιμής περιέχει ακριβώς 1000 τυχαία επιλεγμένες εικόνες από κάθε κλάση. Οι παρτίδες εκπαίδευσης περιέχουν τις υπόλοιπες εικόνες με τυχαία σειρά, αλλά ορισμένες παρτίδες εκπαίδευσης μπορεί να περιέχουν περισσότερες εικόνες από μια κλάση από ό,τι από μια άλλη. Μεταξύ τους, οι παρτίδες εκπαίδευσης περιέχουν ακριβώς 5000 εικόνες από κάθε κλάση.
Παρακάτω θα υλοποιήσουμε from scratch σε python το Support Vector Machine, παρουσιάζοντας τόσο το θεωρητικό υπόβαθρο, όσο και τα αποτελέσματα της υλοποίησης.

2 Υλοποίηση

Θα περιγράψουμε, αρχικά, το μαθηματικό υπόβαθρο της υλοποίησης ενός Support Vector Machine. Παράλληλα με την θεωρητική ερμηνεία θα υπάρχει κι επεξήγηση του υλοποιημένου κώδικα σε python. Σημειώνουμε ότι για την μείωση της πολυπλοκότητας χρησιμοποιήθηκε ο αλγόριθμος PCA (Principal Component Analysis) και εντοπίστηκε ο αριθμός των features που διατηρούν το 90% της πληροφορίας ως $num\_components = 103$.

2.1 Κλάση **SVM**

Τα Support Vector Machine είναι αλγόριθμοι μάθησης με επίβλεψη που χρησιμοποιούνται για classification και regression. Αναζητούν ένα βέλτιστο υπερεπίπεδο για τον διαχωρισμό των σημείων δεδομένων διαφορετικών κλάσεων, μεγιστοποιώντας το περιθώριο (margin) (απόσταση από τα πλησιέστερα σημεία). Τα διανύσματα υποστήριξης (Support Vectors), τα πλησιέστερα σημεία δεδομένων στο υπερεπίπεδο, επηρεάζουν τη θέση του. Τα SVM υπερέχουν σε χώρους υψηλών διαστάσεων και χειρίζονται μη γραμμικές σχέσεις χρησιμοποιώντας πυρήνες (kernels). Πυρήνες όπως ο γραμμικός, ο πολυωνυμικός ή ο RBF εξυπηρετούν διαφορετικούς τύπους δεδομένων. Η παράμετρος κανονικοποίησης C εξισορροπεί το εύρος του περιθωρίου και την ορθή ταξινόμηση. Ένα μικρότερο C αποδίδει ένα πιο μαλακό περιθώριο για βελτιωμένη γενίκευση. Οι SVM χρησιμοποιούν μια hinge loss function για να τιμωρήσουν τις λανθασμένες ταξινομήσεις και να ενθαρρύνουν μεγαλύτερα περιθώρια, προωθώντας την καλύτερη γενίκευση.



Σχήμα 1: Παράδειγμα SVM

Προκειμένου να υλοποιήσουμε τα παραπάνω κατασκευάζουμε την κλάση SVM, μέσα στην οποία θα κατασκευάσουμε τις συναρτήσεις που θα μας χρειαστούν για την εκπαίδευση του μοντέλου, τον διαχωρισμό των δύο κλάσεων του $database$ μας και την κατηγοριοποίηση νέων δεδομένων στις δύο κλάσεις.

2.2 Εκπαίδευση

Για την εκπαίδευση του μοντέλου μας χρησιμοποιούμε την Dual Lagrangian loss function:

$$\boxed{L_{dual} = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j K(x_i, x_j)}$$

την οποία και θα πρέπει να μεγιστοποιήσουμε για την εύρεση του βέλτιστου υπερεπιπέδου με το μέγιστο περιθώριο μεταξύ των δύο διαφορετικών κλάσεων στο χώρο των χαρακτηριστικών (Feaure Space). Υπολογίζουμε, λοιπόν, την παράγωγό της:

$$\boxed{\frac{\delta L_{dual}}{\delta \alpha_k} = 1 - y_k \sum_j \alpha_j y_j K(x_j, x_k)}$$

όπου $K(x_j, x_k)$ είναι η συνάρτηση $Kernel$ και μπορεί να είναι γραμμική, πολυωνυμική ή γκαουσιανή (rbf), όπως θα δούμε παρακάτω.
Στη συνέχεια βρίσκουμε επαναληπτικά το μέγιστο με τη μέθοδο της μέγιστης καθόδου ($gradientdescent$):

$$\boxed{\alpha = \alpha + \eta * \nabla}$$

όπου η είναι ο βαθμός μάθησης.

2.3 Γραμμικός Πυρήνας

Οι πυρήνες (Kernels) είναι συναρτήσεις μετασχηματισμού των δεδομένων μας ώστε να αναδείξουν τις διαφορές τους και να γίνει εφικτή η κατηγοριοποίησή τους. Ο πιο απλός πυρήνας είναι ο γραμμικός:

$$\boxed{z = c + X \cdot y}$$

2.4 Πολυωνυμικός Πυρήνας

Επέκταση του γραμμικού πυρήνα είναι ο πολυωνυμικός, ο οποίος είναι ένας μετασχηματισμός πολυωνύμου δευτέρου ή ανωτέρω βαθμού:

$$\boxed{z = (c + X \cdot y)^d}$$

όπου d είναι ο βαθμός του πολυωνύμου.

2.5 Γκαουσιανός Πυρήνας

Τέλος, μπορούμε να εφαρμόσουμε μια γκαουσιανή συνάρτηση μετασχηματισμού (RBF), η οποία βρίσκει και τη μεγαλύτερη εφαρμογή:

$$\boxed{z = e^{-\frac{1}{\sigma^2}||X-y||^2}}$$

όπου σ είναι παράμετρος προς προσδιορισμό.

2.6 Πρόβλεψη

Μετά την εκπαίδευση υπολογίζουμε το σημείο intercept b ως εξής:

$$\boxed{b = \text{avg}_{C \leq \alpha_i \leq 0} \left\{ y_i - \sum_j \alpha_j y_j K(x_j, x_i) \right\}}$$

και η κατάταξη του νέου στοιχείου γίνεται για $\alpha > 0$ σύμφωνα με τον κανόνα:

$$\boxed{\hat{y} = \text{sign} \left(\sum_i \alpha_i y_i K(x_i, x_i) + b \right)}$$

Έτσι, εάν το πρόσημο είναι θετικό το νέο στοιχείο ανήκει στην μία κλάση, ενώ εάν είναι αρνητικό ανήκει στην άλλη κλάση.

2.7 Υπολογισμός Ακρίβειας

Για την αξιολόγηση του μοντέλου μας, το βάζουμε να κατηγοριοποιήσει νέα άγνωστα δεδομένα και μετράμε πόσα από αυτά κατηγοριοποιήθηκαν σωστά. Έτσι, κατασκευάζεται η συνάρτηση calculate_accuracy().

3 Αποτελέσματα - Σχολιασμοί

Ακολουθούν τα αποτελέσματα της παραπάνω θεωρητικής ανάλυσης και της υλοποίησης του κώδικα σε *python*. Θα δοθούν χαρακτηριστικά παραδείγματα ορθής και εσφαλμένης κατηγοριοποίησης, καθώς και ποσοστά επιτυχίας στα στάδια της εκπαίδευσης (*training*) και του ελέγχου (*testing*), χρόνος εκπαίδευσης και ποσοστά επιτυχίας για διαφορετικές τιμές των παραμέτρων εκπαίδευσης. Επιπλέον, θα συγκριθεί η απόδοση του **SVM** σε σχέση με την κατηγοριοποίηση πλησιέστερου γείτονα (*Nearest Neighbor*) και πλησιέστερου κέντρου κλάσης (*Nearest Class Centroid*) της ενδιάμεσης εργασίας. Σημειώνουμε ότι για λόγους απόδοσης χρησιμοποιήθηκε το μισό *dataset*. Η ορθή και εσφαλμένη κατηγοριοποίηση μπορεί να δειχθεί με την μορφή:

```
pred: -1.0  true: -1.0
pred: -1.0  true: -1.0
pred:  1.0  true:  1.0
pred:  1.0  true: -1.0
pred: -1.0  true: -1.0
pred:  1.0  true:  1.0
```

3.1 Γραμμικός Πυρήνας

Θέτουμε την παράμετρο $kernel = 'poly'$ για πολυωνυμικό πυρήνα κι έπειτα την παράμετρο $degree = 1$ για να έχουμε γραμμική σχέση. Τα αποτελέσματα για τους 4 κατηγοριοποιητές μας φαίνονται στον παρακάτω πίνακα:

Classifier	Train Accuracy	Test Accuracy	Execution Time
SVM	0.692	0.706	$432s$
KNN (k=1)	1.000	0.732	$0.213s$
KNN (k=3)	0.830	0.772	$0.144s$
NC	0.715	0.752	$0.067s$

Πίνακας 1: Ενδεικτική συγκριτική απόδοση για γραμμικό πυρήνα

Παρατηρήσεις:
Παρατηρούμε ότι για γραμμικό πυρήνα, με το δεδομένο *dataset*, το *SVM* όχι μόνο εμφανίζει πολύ μεγαλύτερη υπολογιστική πολυπλοκότητα, αλλά παρουσιάζει και την χειρότερη επίδοση μεταξύ των κατηγοριοποιητών.

3.2 Πολυωνυμικός Πυρήνας

Θέτουμε την παράμετρο $kernel =' poly'$ για πολυωνυμικό πυρήνα κι έπειτα την παράμετρο $degree = 3$. Τα αποτελέσματα για τους 4 κατηγοριοποιητές μας φαίνονται στον παρακάτω πίνακα:

Classifier	Train Accuracy	Test Accuracy	Execution Time
SVM	0.618	0.644	$375s$
KNN (k=1)	1.000	0.712	$0.265s$
KNN (k=3)	0.839	0.698	$0.178s$
NC	0.712	0.758	$0.057s$

Πίνακας 2: Ενδεικτική συγκριτική απόδοση για πολυωνυμικό πυρήνα

Παρατηρήσεις:
Παρατηρούμε ότι για πολυωνυμικό πυρήνα έχουμε παρόμοια συμπεριφορά με προηγουμένως. Με το δεδομένο $dataset$, το SVM όχι μόνο εμφανίζει πολύ μεγαλύτερη υπολογιστική πολυπλοκότητα, αλλά παρουσιάζει και την χειρότερη επίδοση μεταξύ των κατηγοριοποιητών.

3.2.1 Δοκιμές παραμέτρων

Δοκιμάζουμε διάφορες τιμές του βαθμού του πολυωνύμου $degree$ και έχουμε τον παρακάτω πίνακα αποτελεσμάτων:

Degree	Train Accuracy	Test Accuracy	Execution Time
2	0.511	0.532	$367s$
3	0.625	0.626	$365s$
4	0.544	0.536	$372s$
5	0.593	0.584	$369s$

Πίνακας 3: Ενδεικτικές δοκιμές degree

Παρατηρήσεις:
Παρατηρούμε ότι η επίδοση του αλγορίθμου είναι καλύτερη για περιττό βαθμό του πολυωνύμου του πυρήνα. Επιπλέον, για μεγαλύτερους βαθμούς η επίδοση φαίνεται να μειώνεται περαιτέρω.

3.3 Γκαουσιανός Πυρήνας

Θέτουμε την παράμετρο $kernel =' rbf'$ για πολυωνυμικό πυρήνα κι έπειτα την παράμετρο $sigma = 100$. Τα αποτελέσματα για τους 4 κατηγοριοποιητές μας φαίνονται στον παρακάτω πίνακα:

Classifier	Train Accuracy	Test Accuracy	Execution Time
SVM	0.895	0.864	398s
KNN (k=1)	1.000	0.712	0.265s
KNN (k=3)	0.839	0.698	0.178s
NC	0.712	0.758	0.057s

Πίνακας 4: Ενδεικτική συγκριτική απόδοση για γκαουσιανό πυρήνα

Παρατηρήσεις:
Παρατηρούμε, επιτέλους, ότι το SVM παρουσιάζει αρκετά καλύτερη επίδοση από τους υπόλοιπους κατηγοριοποιητές.

3.3.1 Δοκιμές παραμέτρων

Δοκιμάζουμε διάφορες τιμές της παραμέτρου $sigma$ της γκαουσιανής συνάρτησης και έχουμε τον παρακάτω πίνακα αποτελεσμάτων:

Sigma	Train Accuracy	Test Accuracy	Execution Time
1	1.000	0.496	398s
10	1.000	0.562	402s
100	0.889	0.876	401s
1000	0.7625	0.785	403s

Πίνακας 5: Ενδεικτικές δοκιμές sigma

Παρατηρήσεις:
Παρατηρούμε ότι για μικρό αριθμό της παραμέτρου sigma ο αλγόριθμος είναι αναποτελεσματικός, καθώς κατατάσσει τα νέα στοιχεία φαινομενικά τυχαία. Ωστόσο, καθώς ανεβάζουμε την τιμή της παραμέτρου η επίδοση αυξάνεται σημαντικά και φαίνεται να φτάνει σε ένα ταβάνι γύρω στην τιμή 100.

Αναφορές

[1] CIFAR-10 and CIFAR-100 datasets. *Canadian Institute for Advanced Research (CIFAR)*. https://www.cs.toronto.edu/~kriz/cifar.html

[2] K-nearest neighbors algorithm. *Wikipedia*. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[3] Nearest centroid classifier. *Wikipedia*. https://en.wikipedia.org/wiki/Nearest_centroid_classifier

[4] Support Vector Machines Part 1 (of 3): Main Ideas!!!, *YouTube*. https://www.youtube.com/watch?v=efR1C6CvhmE

[5] 16. Learning: Support Vector Machines, *YouTube*. https://www.youtube.com/watch?v=_PwhiWxHK8o

[6] SVM from Scratch, *Kaggle*. https://www.kaggle.com/code/prabhat12/svm-from-scratch