ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

ΤΕΛΙΚΗ ΑΝΑΦΟΡΑ

ΤΣΟΥΡΑΚΗΣ ΟΡΦΕΑΣ – SDI1700175

ΤΑΡΑΤΣΑΣ ΙΩΑΝΝΗΣ – SDI1700160

ΠΕΡΙΕΧΟΜΕΝΑ

1. Εισαγωγή

Είναι γνωστός ο τεράστιος όγκος πληροφορίας ο οποίος υπάρχει στο διαδίκτυο, όσον αφορά τα προϊόντα τα οποία μπορεί να αναζητήσει κανείς. Παρόλα αυτά επειδή, τα διάφορα αυτά αντικείμενα τα συναντάμε και τα βρίσκουμε σε διάφορες ιστοσελίδες, παρατηρείται το φαινόμενο πως η καταχώρηση των στοιχείων τους διαφέρει από ιστότοπο σε ιστότοπο. Έτσι, όπως είναι προφανές είναι πολύ σημαντικό στις βάσεις δεδομένων να υπάρχουν οργανωμένα τα διαφορετικά προϊόντα και να είναι γνωστή η συσχέτιση μεταξύ αυτών.

Στόχος του παρακάτω έργου, είναι η δημιουργία των κατάλληλων δομών δεδομένων, οι οποίες θα έχουν την δυνατότητα να συσχετίζουν όμοια αντικείμενα μεταξύ τους, αλλά και ανόμοια, δημιουργώντας έτσι ένα τεράστιο όγκο πληροφορίας. Αφού, επιτευχθεί η σωστή οργάνωση-ομαδοποίηση της ήδη υπάρχουσας πληροφορίας, καλούμαστε μέσω της χρήσης της μηχανικής μάθησης και των πιθανοτικών μοντέλων, να κατηγοριοποιήσουμε καινούρια άγνωστα δείγματα βάσει των ήδη γνωστών. Παρόλα αυτά, επειδή το μέγεθος της πληροφορίας μεγαλώνει ολοένα και περισσότερο, καλούμαστε να εφαρμόσουμε τεχνικές παραλληλίας με κύριο στόχο την μείωση του συνολικού χρόνου εκτέλεσης του προγράμματος.

2. Πρόβλημα

2.1 Δεδομένα

Τα δεδομένα τα οποία χρησιμοποιήθηκαν προέρχονται από 24 διαφορετικά websites (για παράδειγμα ebay.com, alibaba.com, …). Συγκεκριμένα, τα Dataset που μας δόθηκαν είναι:

1) Το *Dataset X*, το οποίο είναι ο φάκελος με δεδομένα των καταστημάτων που περιγράψαμε παραπάνω.

2) Το *Dataset W*, το οποίο είναι ένα csv αρχείο το οποίο περιέχει συσχετίσεις(αρνητικές αλλά και θετικές μεταξύ αντικειμένων). Δηλαδή, κάθε γραμμή αποτελείται από 2 ονόματα των δύο αντικειμένων τα οποία συγκρίνονται και μια τιμή 0 ή 1 αναλόγως αν ταιριάζουν ή όχι.

3) Το *Dataset Y* το οποίο είναι ένα υποσύνολο του Dataset X.

2.2 Ανάλυση προβλήματος

Το έργο χωρίστηκε σε 3 στάδια.

<u>Πρώτο Στάδιο</u>

Το πρώτο στάδιο ήταν η ανάγνωση των δεδομένων, η τοποθέτηση τους σε δομές και η ομαδοποίηση των όμοιων αντικειμένων. Συγκεκριμένα, όσον αφορά το τελευταίο ορίσαμε την έννοια της

κλίκας σε έναν γράφο όπου οι κόμβοι είναι τα διάφορα αντικείμενα και οι ακμές που συνδέουν αυτά τα αντικείμενα παίρνουν τις τιμές +1, -1 (+1 σημαίνει πως ταιριάζουν, και αντίστοιχά το -1 πως δεν ταιριάζουν). **Κλίκα** (clique) σε ένα γράφο ονομάζεται ένας πλήρης υπογράφος του. Συγκεκριμένα, η υλοποίηση που πραγματοποιήθηκε, εκμεταλλεύεται αυτήν την ιδιότητα δημιουργώντας μόνο τις κλίκες των γράφων και όχι τους πλήρεις γράφους.

Δεύτερο στάδιο

Το δεύτερο στάδιο της εργασίας ήταν η αναγνώριση των ανόμοιων αντικειμένων και η δημιουργία αρνητικών συσχετίσεων μεταξύ των κλικών οι οποίες δεν ταιριάζουν μεταξύ τους. Αυτό βέβαια σημαίνει, ότι αν δύο αντικείμενα δεν ταιριάζουν μεταξύ τους τότε και όλα τα όμοια αντικείμενα της μίας κλίκας δεν θα ταιριάζουν και αυτά με τα αντικείμενα της αντίστοιχης κλίκας.

Στην συνέχεια, αφού καλούμαστε να καθαρίσουμε το κείμενο, το οποίο περιγράφει τα χαρακτηριστικά των αντικειμένων, από λέξεις οι οποίες δεν μας συνεισφέρουν κάποια επιπλέον πληροφορία, μετατρέπουμε το κείμενο σε μία αναγνωρίσιμη από τον υπολογιστή διανυσματική μορφή, με την χρήση του μοντέλου αναπαράστασης Bag-Of-Words αλλά και του μοντέλου TF-IDF. Τέλος, χρησιμοποιώντας τον αλγόριθμο της λογιστικής παλινδρόμησης(Logistic Regression), στόχος είναι να εκτιμήσουμε σε ποια από τις ήδη δημιουργημένες ομάδες ανήκει ένα άγνωστο δείγμα.

Τρίτο στάδιο

Το τρίτο στάδιο της εργασίας, ήταν η υλοποίηση του μοντέλου επαναληπτικής εκμάθησης για την αύξηση των δεδομένων με τα οποία θα προπονήσουμε το μοντέλο αλλά και η εισαγωγή της παραλληλίας στο πρόγραμμα, χρησιμοποιώντας νήματα τα οποία επεξεργάζονται μικρότερα κομμάτια της εισόδου (**mini-batch υλοποίηση**), έτσι ώστε να μειώσουμε το συνολικό χρόνο εκτέλεσης. Επίσης, κληθήκαμε να χρησιμοποιήσουμε τον πολυνηματισμό και για την δημιουργία νέων άγνωστων εγγραφών που προκύπτουν από όλους τους συνδυασμούς των προϊόντων του Dataset X, αλλά και για να επαληθεύσουμε το μοντέλο μας.

3. Δομές δεδομένων

Για την επίλυση των παραπάνω προβλημάτων χρησιμοποιήθηκαν διάφορες δομές δεδομένων. Η υλοποίηση της εργασίας από παραδοτέο σε παραδοτέο τροποποιήθηκε και για αυτό παρακάτω θα αναλύσουμε την πορεία την οποία ακολουθήσαμε.

3.1 Αφηρημένοι τύποι δεδομένων

Κατά το πρώτο μέρος της εργασίας, η κύρια δομή η οποία χρησιμοποιήσαμε αρχικά, για την αποθήκευση της κύριας πληροφορίας ήταν τα Red-Black Trees. Ψάχνοντας στην βιβλιογραφία, είχαμε καταλήξει στο συμπέρασμα ότι είναι ένα από τα ίσως καλύτερα, ισοζυγισμένα δυαδικά δέντρα, για γρήγορη αναζήτηση O(log(n)), μιας και η αναζήτηση ήταν μια λειτουργία η οποία γινόταν σε τεράστιο βαθμό.

Παρόλα αυτά, από το δεύτερο παραδοτέο και στην συνέχεια, επειδή συμπεράναμε πως έχοντας ως βασική δομή μόνο αυτήν των Red Black Trees δεν είναι ιδανική για αναζήτηση σε είσοδο που δεν αλλάζει, δηλαδή δεν έχει το πρόβλημα του επανακαθορισμού μεγέθους, οπότε τροποποιήσαμε την δομή αυτή και καταλήξαμε να τη συνδυάσουμε με την δομή των Hash Tables. Συγκεκριμένα, κάθε κουβάς (bucket) οδηγεί σε ένα Red Black Tree. Αυτό, βέβαια, το θεωρήσαμε βέλτιστο μιας και πλέον εκμεταλλευόμαστε τα θετικά και των δύο δομών, εφόσον πλέον τα Hash Tables, μας δίνουν την δυνατότητα να περιηγηθούμε σε πολύ μικρότερα δέντρα, στα οποία λειτουργίες όπως η αναζήτηση, που αναφέραμε παραπάνω δεν είναι αρκετά χρονοβόρα.

Εκτός από την κύρια δομή την οποία αναφέραμε παραπάνω, καταλυτική σημασία έχουν και οι άλλες, πιο απλές δομές δεδομένων που υλοποιήσαμε. Συγκεκριμένα, οι επιπλέον αυτές δομές είναι της ουράς, του σωρού και της στοίβας οι οποίες χρησιμοποιήθηκαν σε συγκεκριμένα υπό βήματα.

3.2 Επιπλέον δομές

Απαραίτητο για την διεκπεραίωση της εργασίας ήταν και η υλοποίηση κάποιων δικών μας δομών με τις οποίες μπορέσαμε να παραστήσουμε ένα μέρος της πληροφορίας. Παρακάτω θα δούμε κάποιες βασικές δομές που χρησιμοποιήσαμε:

1)Αντικείμενα του Dataset X: Η δομή που περιέχει το όνομα του αντικειμένου και τα χαρακτηριστικά του.

2)Κλίκα: Η δομή η οποία περιέχει τα αντικείμενα, τα συσχετιζόμενα με αυτά αντικείμενα, αλλά και τα μη συσχετιζόμενα με αυτά αντικείμενα.

3)Μοντέλο Logistic Regression: Περιλαμβάνει τα βασικά «εργαλεία» για το μοντέλο.

4) Δομές με στατιστικά: Κάποιες δομές οι οποίες μας βοηθάνε να αποθηκεύουμε στατιστικά, όπως για παράδειγμα στατιστικά λέξεων, αρχείων, εγγραφών, αλλά και στατιστικά που αφορούν το ίδιο το μοντέλο.

5)Δομή JobScheduler: Υλοποίηση της ιδέας ενός προγραμματιστή εργασιών, για τον συγχρονισμό και την διαχείριση των νημάτων.

4. Επίλυση κάθε σταδίου

Παρακάτω θα αναλύσουμε πως ανταποκριθήκαμε στις απαιτήσεις της εργασίας και πως επιλύθηκε κάθε στάδιο.

Πρώτο Στάδιο

Στο στάδιο αυτό αρχικά σαρώσαμε τους καταλόγους από το Dataset X (αρχεία προϊόντων ιστοχώρων) δημιουργώντας έτσι όλα τα αντικείμενα. Έπειτα, διαβάσαμε το Dataset W συγκρατώντας μόνο τα θετικά ζεύγη, μέσω των οποίων δημιουργήσαμε τις θετικές συσχετίσεις των κλικών, εξάγοντας έτσι στο τέλος όλα τα συσχετιζόμενα ζεύγη, τα οποία προέκυψαν και από την μεταβατική ιδιότητα της ισότητας, στο αρχείο related.csv.

Δεύτερο Στάδιο

Αρχικά, διαβάζοντας, πλέον, όλο το Dataset W δημιουργούμε και τις αρνητικές συσχετίσεις μεταξύ των κλικών. Τις κλίκες, τις τοποθετούμε σε Hash Tables, εξασφαλίζοντας έτσι την γρήγορη πρόσβαση σε αυτές.

Από τις ολοκληρωμένες πλέον κλίκες, μπορούμε να εξάγουμε όλα τα ζευγάρια τα οποία ταιριάζουν και δεν ταιριάζουν μεταξύ τους (ιδιότητα μεταβατικότητας), προσθέτοντας, με αυτόν τον τρόπο, παραπάνω πληροφορία – νέες εγγραφές, συσχετιζόμενων και μη - συσχετιζόμενων αντικειμένων, επεκτείνοντας έτσι το αρχικό Dataset. Έπειτα, επεξεργαζόμαστε αυτό το Dataset χωρίζοντας το σε αντίστοιχα μικρότερα κομμάτια-σετ, τα οποία θα μας χρησιμεύσουν για **προπόνηση**(training set), **δοκιμή**(test set) και **επαλήθευση**(validation set) του μοντέλου μας.

Παράλληλα, δημιουργούμε και τα μοντέλα κωδικοποίησης Bag Of Words και TF-IDF όλων των JSON αρχείων, ώστε με την αναπαράσταση αυτή να μπορούμε να τα τοποθετήσουμε ως είσοδο στο μοντέλο.

Τέλος, δημιουργούμε το τελικό λεξιλόγιο, καθαρισμένο από περιττές λέξεις και έχοντας κρατήσει αυτές που μας δίνουν την πιο σημαντική πληροφορία, προπονούμε το μοντέλο μας με το σετ προπόνησης και το δοκιμάζουμε με το σετ δοκιμής. Έτσι, υπολογίζεται το ποσοστό επιτυχίας των προβλέψεων, δηλαδή το πόσο καλά δουλεύει το μοντέλο που προπονήσαμε.

Τρίτο Στάδιο

Στο τελευταίο αυτό στάδιο προσθέσαμε κώδικα για την επαναληπτική εκμάθηση. Σε κάθε βήμα

αποθηκεύουμε μερικές νέες εγγραφές που προκύπτουν από όλους τους συνδυασμούς των αντικειμένων, και τις οποίες το μοντέλο μας έχει προβλέψει με μεγαλύτερη σιγουριά, ενσωματώνοντας τες στο ήδη υπάρχων σετ προπόνησης, επιλύοντας ύστερα τις αντιφάσεις που προκύπτουν μεταξύ της παλιάς και της νέας πληροφορίας. Για να επιτευχθεί αποδοτικότερα το παραπάνω στο σημείο που παράγονται όλα τα ζευγάρια χρησιμοποιούμε πολυνηματισμό. Πρέπει, επίσης, να σημειωθεί πως για τον συγχρονισμό των νημάτων , χρησιμοποιήσαμε την βιβλιοθήκη POSIX.

Στο τέλος, εκτυπώνουμε μόνο τα συσχετιζόμενα ζευγάρια, και πριν τερματίσει το πρόγραμμα το προπονούμε μια τελευταία φορά και εμφανίζουμε την απόδοση που προκύπτει με βάση το σετ επαλήθευσης.

5. Αποτελέσματα-μετρήσεις

Σε αυτή την ενότητα θα εξετάσουμε κάποια πειράματα και θα συγκρίνουμε τα αποτελέσματα τους ώστε να γίνουν αντιληπτές μερικές από τις βελτιστοποιήσεις μας.

5.1 Πειράματα μοντέλου

Στους παρακάτω πίνακες θα παρουσιάσουμε αποτελέσματα κατά την προπόνηση του μοντέλου χρησιμοποιώντας πολυνηματισμό. Οι εκτελέσεις έχουν πραγματοποιηθεί κρατώντας σταθερές τις ακόλουθες τιμές παραμέτρων:

συνολικές επαναλήψεις(epochs) 100

ρυθμός εκμάθησης(learning rate) 0.01

όριο απόφασης(decision boundary) 0.5

όριο αποκοπής(epsilon) 0.0001

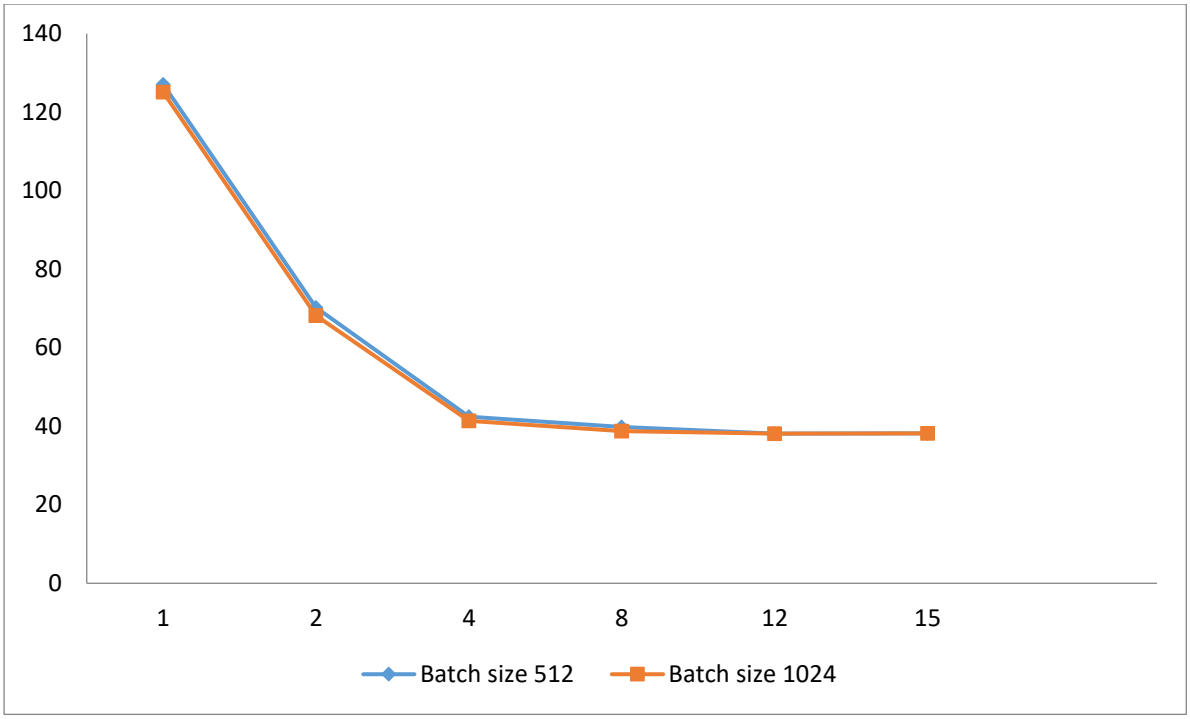
Αριθμός νημάτων	Χρόνος εκτέλεσης(sec)
1	127
2	70
4	42
8	39
12	38
15	38

Πίνακας 1: Προπόνηση του μοντέλου με batch size 512

Αριθμός νημάτων	Χρόνος εκτέλεσης(sec)
1	125
2	68
4	41
8	38
12	38
15	38

Πίνακας 2: Προπόνηση του μοντέλου με batch size 1024

Όπως παρατηρούμε από τους παραπάνω πίνακες αλλά και από το διάγραμμα που ακολουθεί, καθώς αυξάνεται ο αριθμός των νημάτων τόσο μειώνεται ο χρόνος εκτέλεσης της προπόνησης του μοντέλου. Συγκρίνοντας, επίσης, τις δύο περιπτώσεις γίνεται αντιληπτό πως για batch size 1024 έχουμε λίγο καλύτερο χρόνο εκτέλεσης . Παρόλα αυτά αξίζει να σημειωθεί πως μετά από κάποιον αριθμό νημάτων ο χρόνος δεν βελτιώνεται περαιτέρω.



Διάγραμμα 1: Αναπαράσταση πινάκων 1 και 2. Στον άξονα y απεικονίζεται ο χρόνος εκτέλεσης της προπόνησης του μοντέλου και στον άξονα x ο αριθμός των νημάτων.

Τέλος, παρατηρούμε σημαντική βελτίωση του χρόνου εκτέλεσης σε σύγκριση με το αποτέλεσμα της προπόνησης του μοντέλου χωρίς παραλληλία μιας και ο χρόνος αυτός φτάνει τα 182 δευτερόλεπτα δηλαδή είναι ο πιο υψηλός.

5.2 Μετρήσεις απόδοσης του μοντέλου

Έπειτα από αρκετό πειραματισμό των παραμέτρων του μοντέλου καταλήξαμε σε κάποιες αρκετά καλές παραμέτρους έτσι ώστε να πετύχουμε στο απλό μοντέλο απόδοση της τάξης 93% αλλά και στο mini-batch μοντέλο, με την ενσωμάτωση νέας πληροφορίας από την χρήση της επαναληπτικής εκμάθησης, απόδοση της τάξης 90%.

5.3 RB - Trees vs Hash Tables

Όπως αναφέρθηκε νωρίτερα υπήρχε έντονος προβληματισμός για το αν θα χρησιμοποιήσουμε Red-Black-Trees ή Hash Tables. Κάνοντας μετρήσεις στο στάδιο επεξεργασίας των δεδομένων μπορούμε να παρατηρήσουμε πως τα Hash Tables βελτιώνουν τον χρόνο εκτέλεσης. Συγκεκριμένα, με την χρήση της δομής των Hash Tables ολοκληρώνεται το στάδιο επεξεργασίας των δεδομένων σε 15 δευτερόλεπτα ενώ με την δομή των Red-Black-Trees σε 18 δευτερόλεπτα.

5.4 Χρήση αραιού πίνακα(Sparse Array)

Για την κωδικοποίηση των κειμένων όπως ειπώθηκε κρατάμε μόνο τις μη μηδενικές τιμές ενός πίνακα διαστάσεων *αριθμός κειμένων* x *αριθμός λέξεων*. Αυτό, έχει ως αποτέλεσμα να εξοικονομείται ένα τεράστιο κομμάτι μνήμης αλλά και να προσπελαύνουμε τον πίνακα αυτόν αρκετά γρηγορότερα.

5.5 Μέτρηση συνολικού χρόνου εκτέλεσης και δεσμευμένης μνήμης

Ο συνολικός χρόνος εκτέλεσης προκύπτει από τον συνδυασμό και των τριών σταδίων. Το πρώτο στάδιο δεν απαιτούσε αρκετό χρόνο παρά μόνο κάτι δευτερόλεπτα(10 δευτερόλεπτα περίπου).Το δεύτερο στάδιο απαιτούσε λίγα παραπάνω λεπτά(3-4 λεπτά περίπου) λόγω της μη χρήσης της παραλληλίας. Το τρίτο στάδιο είναι αυτό που αργεί αρκετά περισσότερο από τα υπόλοιπα , παρόλο που εφαρμόζουμε και παραλληλία , λόγω του τεράστιου όγκου της νέας πληροφορίας που προστίθεται. Αυτός ο τεράστιος όγκος νέας πληροφορίας απαιτεί επίσης και υπερβολικό χώρο και σε συνδυασμό με την πληροφορία που κρατάμε για όλα τα αρχεία δεσμεύεται ένας αρκετά μεγάλος αριθμός(≈1.7GB) μνήμης ακόμα και για ένα μικρό υποσύνολο νέων δεδομένων από την διαδικασία της επαναληπτικής εκμάθησης.

5.6 Ανέφικτες μετρήσεις

Όπως και στην προπόνηση του μοντέλου έτσι και στην παραγωγή όλων των πιθανών νέων ζευγών χρησιμοποιήσαμε παραλληλία. Όμως, λόγω του retraining τα νέα σετ προπόνησης διέφεραν κάθε φορά σε μέγεθος και έτσι δεν είχαμε κάποιο σταθερό παράγοντα για μπορούμε να εξάγουμε πειραματικά δεδομένα. Βέβαια με την παραλληλία φαίνεται να επιταχύνεται αρκετά αυτή η διαδικασία.

6. Προβλήματα που αντιμετωπίσαμε

Τα κύρια προβλήματα τα οποία αντιμετωπίσαμε σχετίζονται τόσο με τον χρόνο, όσο και με τον χώρο αλλά και με την αναπαράσταση του προβλήματος. Συγκεκριμένα:

<u>Πρόβλημα 1</u>

Κατά το parsing των τιμών των πεδίων στα αρχεία JSON.

Λύση

Χρήση της δομής της στοίβας, για να μπορούμε να διακρίνουμε το πού ξεκινάνε και που τελειώνουν οι τιμές των πεδίων.

Πρόβλημα 2

Κατά την εύρεση των αρνητικών συσχετίσεων μεταξύ των κλικών στο στάδιο ανάγνωσης των δεδομένων όπου οι κλίκες συνενώνονταν.

Λύση

Η κάθε κλίκα αποθήκευε τα ανόμοια αντικείμενα και όχι τις ανόμοιες κλίκες και στο τέλος γινόταν η εναλλαγή.

Πρόβλημα 3

Χρήση dense πίνακα TF-IDF/BOW ως είσοδο στο μοντέλο.

Λύση

Εκμετάλλευση πληροφορίας, η οποία έχει αποθηκευτεί σε δομές προσομοιώνοντας τον dense πίνακα, κρατώντας μόνο την θέση των μη-μηδενικών στοιχείων(sparse υλοποίηση πίνακα).

Πρόβλημα 4

Πρόβλημα στον υπολογισμό της παραγώγου του μοντέλου.

Λύση

Χρήση του Stochastic Gradient Descent.

Πρόβλημα 5

Τεράστιος όγκος των παραγόμενων ζευγών στην επαναληπτική εκμάθηση.

Λύση

Χρήση ορίων αποκοπής παίρνοντας μόνο ένα υποσύνολο του δείγματος.

7. Τι θα μπορούσαμε να κάνουμε επιπλέον

Σε αυτό το κομμάτι θα αναλύσουμε τι θα μπορούσαμε να κάνουμε αν είχαμε περισσότερη μνήμη και χρόνο.

1)Πειραματισμός με παραμέτρους του μοντέλου οι οποίες αν τεθούν στον απλό υπολογιστή μας θα αργούσε πάρα πολύ να βρεθεί η λύση και θα τον επιβάρυνε και σε θέματα μνήμης.

2)Δοκιμή και με άλλα μοντέλα μηχανικής μάθησης και σύγκριση των αποτελεσμάτων.

3) Τρόπους με τους οποίους θα δημιουργούσαμε ένα πιο ισορροπημένου(balanced) Dataset.

8. ΠΡΟΔΙΑΓΡΑΦΕΣ ΜΗΧΑΝΗΜΑΤΟΣ

Οι προδιαγραφές του υπολογιστή στον οποίο εκτελέστηκαν τα παραπάνω πειράματα είναι τα εξής:

CPU: Intel(R) Core(TM) i5-7500 CPU @ 3.40GHz

RAM: 16 GB