Τεχνητή Νοημοσύνη 2020-2021
2η ΕΡΓΑΣΙΑ
Δουραχαλής Φίλιππος, 3170045
Νικολάου Ελένη, 3170121

Εισαγωγή

Στα πλαίσια του μαθήματος της Τεχνητής Νοημοσύνης και για την εκπόνηση της δεύτερης εργασίας, επιλέξαμε να ασχοληθούμε με τον αλγόριθμο ID3.

Χρησιμοποιήσαμε τις μεθόδους που προτείνονται στο σάιτ που δίνεται στην εκφώνηση της εργασίας (https://keras.io/api/datasets/imdb/), για να φορτώσουμε τα δεδομένα.

```
(train_data, train_targets), (test_data, test_targets) = imdb.load_data(num_words=m, skip_top = vocab_start)
```

Τα δεδομένα που επιστρέφονται αποθηκεύονται στις εξής δομές:

- train_data: reviews σε μορφή ακολουθίας θετικών ακεραίων, που αντιστοιχούν σε hashes του vocabulary των δεδομένων, αποθηκευμένα σε numpy array
- train_targets: η κατηγορία (0-1 / θετικά-αρνητικά) που αντιστοιχεί στο κάθε review του προηγούμενου πίνακα, αποθηκευμένα σε numpy array (τα indexes των δύο δομών ταυτίζονται)

Αντίστοιχα και τα test_data και test_targets.

Στη συνέχεια, μετατρέπουμε τα δεδομένα που παίρνουμε στον πίνακα train_data σε μορφή που μπορεί να επεξεργαστεί ευκολότερα ο αλγόριθμός μας. Επιθυμούμε κάθε review (κάθε row) του πίνακα να εκφράζεται από ένα διάνυσμα m μήκους, όπου m οι ιδιότητες που θα ορίσουμε. $<X_1,X_2,X_3,\ldots,X_m>$
Ορίζουμε οι ιδιότητες να είναι m λέξεις του λεξιλογίου του dataset και μέσω της συνάρτησης *reviews_to_vector* , στο διάνυσμα κάθε review θέτουμε 0, το Xi αν η λέξη με hash i δεν περιέχεται στο review, ενώ θέτουμε 1, το Xi αν η λέξη i περιέχεται.

```
def reviews_to_vector(reviews):
    results = np.zeros((len(reviews), m)) #c
    for i, review in enumerate(reviews): # i
        results[i,review] = 1 #for every rev
    return results.astype(np.int64)
```

Υλοποίηση Αλγορίθμου

Ο αλγόριθμος υλοποιείται μέσα στη μέθοδο `id3_alg`, οποίος σε κάθε αναδρομικό βήμα εκτελεί τις εξής λειτουργίες:

1. Αρχικά ελέγχονται οι συνθήκες τερματισμού της αναδρομής για να βεβαιωθούμε ότι αν οποιαδήποτε δομή εισόδου είναι κενή, θα επιστραφεί η κατάλληλη τιμή.
2. Υπολογίζουμε την ιδιότητα με το μέγιστο information gain, μέσω των μεθόδων που θα αναλυθούν στη συνέχεια

3. Κατόπιν αφαιρούμε την ιδιότητα από τη λίστα των διαθέσιμων ιδιοτήτων και χωρίζουμε στα δύο τα δεδομένα εισόδου (data και targets) χρησιμοποιώντας το index της.
4. Καλούμε ξανά τον αλγόριθμο με όρισμα το νέο dataset που προκύπτει όταν η τιμή της ιδιότητας που βρήκαμε είναι και όταν είναι 0 αντίστοιχα
5. Τέλος θέτουμε τον κόσμο ίσο με την τιμή που επέστρεψε ο id3, η οποία μπορεί να είναι μια κατηγορία ή ένα νέο υποδέντρο

Υπολογισμός Εντροπίας και Κέρδους Πληροφορίας (Information Gain)

Πρώτον, έχουμε την μέθοδο `calculate_entropy`, η οποία είναι υπεύθυνη για τον υπολογισμό της εντροπίας του C, στην περίπτωσή μας, για τις τιμές C=0, C=1, για τις κατηγορίες 0-1 αντίστοιχα.

$$H(C) = -\sum\left(P(C = c) * log_2(P\ C = c)\right)$$

Έπειτα με την συνάρτηση `calculate_entropy_attribute`, υπολογίζουμε την εντροπία δεδομένης της τιμής που παίρνει μία ιδιότητα X. Ο τύπος φαίνεται παρακάτω.

$$H(C|\ X = x) = -\sum\left(P(C = c\ |\ X = x) * log_2\ P(C = c|\ X = x)\right)$$

Με τη συνάρτηση `calc_info_gain (attr_index, data, target)`, υπολογίζουμε το κέρδος πληροφορίας βάσει του τύπου:

$$IG(C|X) = H(C) - \sum\left(P(X=x) * H(C|X=x)\right)$$

Τέλος χρησιμοποιούμε μία απλή συνάρτηση για να βρίσκουμε κάθε φορά το μέγιστο info gain από τα attributes που έχουμε υπολογίσει.
`find_max_info_gain(data, target, attributes)`

Για την εύρεση ενός αρκετά καλού συνόλου ιδιοτήτων, δοκιμάσαμε διάφορες τιμές για τις υπερπαραμέτρους m (το πλήθος των ιδιοτήτων) και vocab_start (το σημείο του vocabulary από το οποίο ξεκινάμε να παίρνουμε τις m συχνότερες ιδιότητες).

Για παράδειγμα αρχικά τρέξαμε τον αλγόριθμο με m=1000 και vocab_start = 0. Επομένως η μέθοδος load_data επέστρεψε ένα λεξιλόγιο που περιείχε τις 1000 πιο συχνές λέξεις που εμφανίζονται στα reviews. Αυτή η προσέγγιση όμως αποδείχθηκε προβληματική καθώς οι συγκεκριμένες λέξεις εμφανίζονται σε πολύ μεγάλο αριθμό κειμένων, κι άρα δεν συνεισφέρουν πολύ στη σωστή κατάταξη των reviews.

Στη συνέχεια δοκιμάσαμε τις τιμές m = 1000 και vocab_start = 200 για τις οποίες παρατηρήθηκε κάποια βελτίωση σχετικά με τον τρόπο κατάταξης των κειμένων.