Διαχείριση Σύνθετων Δεδομένων
Εργασία 3 – Ερωτήσεις κορυφαίων κ και κορυγογραμμής

Διαχείριση Σύνθετων Δεδομένων
Εργασία 3 – Ερωτήσεις κορυφαίων κ και κορυφογραμμής

Κωνσταντίνος Παπαντωνίου-Χατζηγιώσης, ΑΜ: 4769

Στην εργασία αυτή μας ζητήθηκε η ανάπτυξη αλγορίθμου για την εύρεση top-K ζευγαριών με το υψηλότερο άθροισμα των πεδίων instance weight, για ζευγάρια με την ίδια ηλικία. Τα δεδομένα προέρχονται από το https://kdd.ics.uci.edu/databases/census-income/census-income.html και αποτελούν δημογραφικά δεδομένα από τις ΗΠΑ. Στόχος της εργασίας ήταν η ανάπτυξη δύο διαφορετικών αλγορίθμων για την εύρεση των ζευγαριών χρησιμοποιώντας διαφορετικές τεχνικές ώστε να συγκρίνουμε την αποδοτικότητα των αλγορίθμων για διαφορετικές παραμέτρους. Τα ζευγάρια που λαμβάνονται υπόψη είναι μόνο αυτά με ηλικία και των δύο φύλων άνω των 18 χρονών και παράλληλα ανύπαντρα. Τα άτομα που δεν τηρούν αυτά τα κριτήρια τα προσπερνάμε.

Μέρος Πρώτο:

Στο πρώτο μέρος της εργασίας αναπτύχθηκε ο αλγόριθμος HRJN, όπου διαβάζουμε εναλλάξ τα δεδομένα από τα δύο αρχεία και με το που εμφανιστεί το κατάλληλο ζευγάρι το επιστρέφουμε απευθείας χωρίς να διαβάσουμε τα υπόλοιπα αρχεία. Αυτό δίνει στη συνάρτηση τον χαρακτήρα μιας γενετικής συνάρτησης, την οποία καλούμε K φορές για να μας επιστρέψει K αποτελέσματα. Το πρόγραμμα αποτελείται από τρεις βοηθητικές συναρτήσεις και την βασική.

- **read_file:** Η συνάρτηση αυτή χρησιμοποιείται για την ανάγνωση δεδομένων. Συγκεκριμένα, δέχεται το αρχείο που θα διαβάσει σαν παράμετρο και στη συνέχεια διαβάζει μία γραμμή, και όχι όλο το αρχείο μαζί. Αφού έχει διαβάσει τη γραμμή και η γραμμή έχει δεδομένα, ελέγχει τα δύο κριτήρια που ορίσαμε, δηλαδή εάν το άτομο είναι ενήλικας και ανύπαντρος, καθώς η κάθε γραμμή περιέχει πληροφορίες για ένα συγκεκριμένο άτομο. Εάν τηρεί τα κριτήρια, επιστρέφουμε το αναγνωριστικό, την ηλικία και το πεδίο βαρύτητάς του.
- **get_next_valid_entry:** Η συνάρτηση αυτή χρησιμοποιεί τη συνάρτηση read_file για να επιστρέψει τις επόμενες έγκυρες εγγραφές από το αρχείο χρησιμοποιώντας τη λέξη κλειδί yield, το οποίο δουλεύει γενετικά και καλείται κάθε φορά για να επιστρέψει το επόμενο αποτέλεσμα.

- **hrjn_algorithm:** Η συνάρτηση αυτή είναι ο πυρήνας του αλγορίθμου και είναι το σημείο που τον υλοποιούμε. Αρχικά, ανοίγουμε τα δύο αρχεία και καλούμε τη βοηθητική συνάρτηση για να φτιάξουμε δύο iterator ώστε να μπορούμε κάθε φορά να παίρνουμε το επόμενο έγκυρο άτομο. Εάν έχουμε τα δεδομένα και δεν είναι κενά, μπαίνουμε σε έναν βρόχο. Εκεί ελέγχουμε αν πρέπει να διαβάσουμε άντρα ή γυναίκα, καθώς θέλουμε να διαβάζουμε εναλλάξ τα άτομα. Είτε διαβάζουμε άντρα είτε γυναίκα, ελέγχουμε εάν στα λεξικά που έχουμε ορίσει στην αρχή της συνάρτησης υπάρχει πεδίο με συγκεκριμένη ηλικία. Εάν δεν υπάρχει, δημιουργούμε μια νέα θέση στο λεξικό με κλειδί την ηλικία, αφού θέλουμε τα ζευγάρια με την ίδια ηλικία κάθε φορά. Εάν υπάρχει ήδη πεδίο με τη συγκεκριμένη ηλικία, του προσθέτουμε τα δεδομένα του ανθρώπου. Στο επόμενο βήμα, ελέγχουμε εάν η ηλικία αυτή υπάρχει και σαν πεδίο στο λεξικό του άλλου φύλου και αν βρεθεί αντιστοιχία, τότε προσθέτουμε, σε μία ουρά που έχουμε ορίσει στην αρχή της συνάρτησης, το αντίθετο του αθροίσματος των βαρών των δύο ατόμων καθώς και τα αναγνωριστικά τους. Το άθροισμα 'σκορ' το προσθέτουμε με αρνητική τιμή, καθώς θέλουμε να είναι το πρώτο στοιχείο που θα επιστρέψει η συνάρτηση να είναι αυτό με το μεγαλύτερο σκορ, αλλά η ουρά επιστρέφει το τελευταίο της στοιχείο πάντα, οπότε με την αρνητική τιμή το στοιχείο αυτό γίνεται το μικρότερο στοιχείο της ουράς, άρα στο τέλος θα είναι το πρώτο που θα δώσει η ουρά. Αφού τοποθετήσουμε τα στοιχεία στην ουρά, ενημερώνουμε τις τιμές R, L αντίστοιχα για bottom, top, όπου η bottom κρατάει πάντα το ελάχιστο βάρος και η top το μέγιστο. Η διαδικασία αυτή είναι η ίδια και για τα δύο φύλα. Στο επόμενο βήμα υπολογίζουμε το κατώφλι και αντιστρέφουμε την τιμή read_male ώστε στο επόμενο πέρασμα του βρόχου να διαβάσει αυτή τη φορά το άλλο φύλο. Τέλος, συγκρίνουμε τα σκορ με το κατώφλι και από τα αντικείμενα της ουράς που περνάνε επιστρέφουμε το τελευταίο της στοιχείο διορθώνοντας και την αρνητική τιμή του σκορ. Εδώ ομοίως επιστρέφουμε τα αντικείμενα μέσω της yield, η οποία καθιστά τη συνάρτηση σαν iterator, οπότε μπορούμε κάθε φορά να καλέσουμε το επόμενο K στοιχείο καθιστώντας την γενετική.
- **Main:** Στη main καλούμε κάθε φορά τα K επόμενα αντικείμενα της hrjn_algorithm, όπου K το παίρνουμε από την γραμμή εντολών, και εκτυπώνουμε τα ζευγάρια αλλά και τον χρόνο εκτέλεσης του προγράμματος.

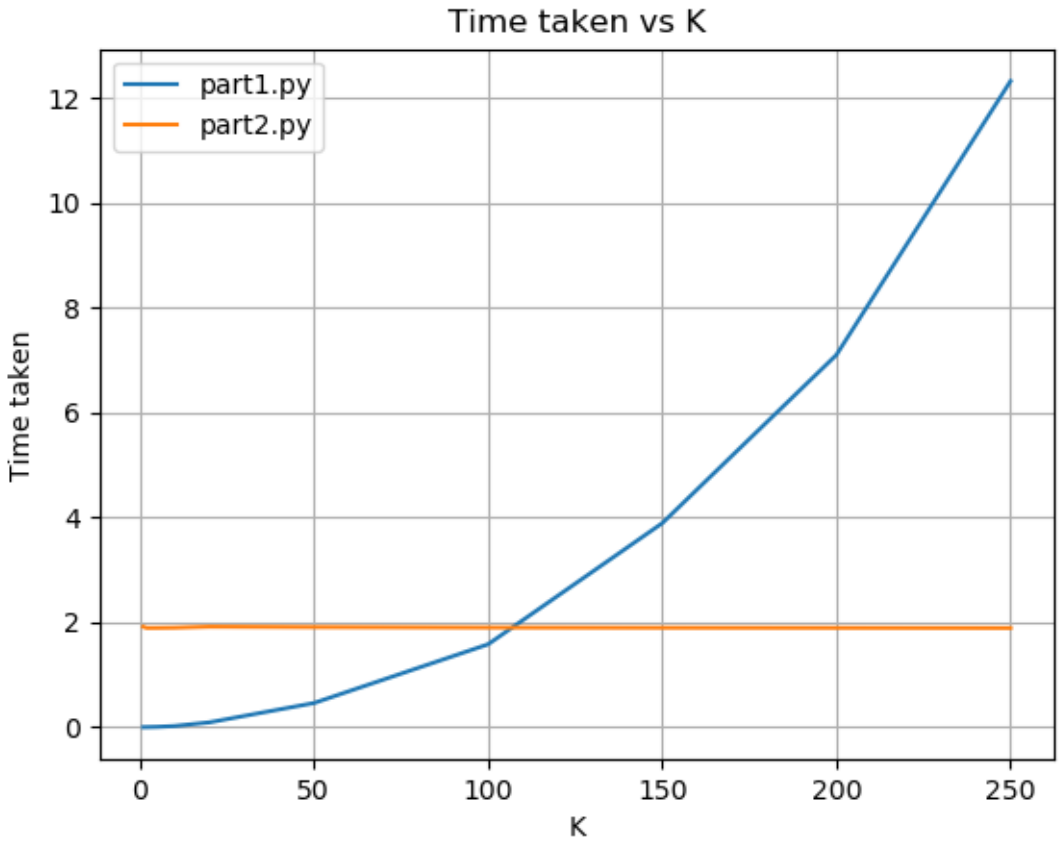
Μέρος Δεύτερο:

Στο δεύτερο μέρος της εργασίας υλοποιούμε, όπως πριν, έναν αλγόριθμο K-top, αλλά αυτή τη φορά αντί να το κάνουμε με κάποια γενετική μέθοδο, διαβάζουμε πρώτα τα δεδομένα και στο τέλος επιστρέφουμε όλα μαζί τα K ζευγάρια με το καλύτερο σκορ.

- **read_file:** Η συνάρτηση αυτή χρησιμοποιείται για την ανάγνωση δεδομένων. Συγκεκριμένα, δέχεται το αρχείο που θα διαβάσει σαν παράμετρο και στη συνέχεια διαβάζει μία γραμμή, και όχι όλο το αρχείο μαζί. Αφού έχει διαβάσει τη γραμμή και η γραμμή έχει δεδομένα, ελέγχει τα δύο κριτήρια που ορίσαμε, δηλαδή εάν το άτομο είναι ενήλικας και ανύπαντρος, καθώς η κάθε γραμμή περιέχει πληροφορίες για ένα συγκεκριμένο άτομο. Εάν τηρεί τα κριτήρια, επιστρέφουμε το αναγνωριστικό, την ηλικία και το πεδίο βαρύτητάς του.

- **get_next_valid_entry:** Η συνάρτηση αυτή χρησιμοποιεί τη συνάρτηση read_file για να επιστρέψει τις επόμενες έγκυρες εγγραφές από το αρχείο.
- **b_top_algorithm:** Στη συνάρτηση αυτή, αντίθετα με το πρώτο μέρος της εργασίας, αφού αρχικοποιήσουμε το λεξικό, την ουρά και ανοίξουμε τα δύο αρχεία, διαβάζουμε όλες τις έγκυρες γραμμές του ανδρικού φύλου μαζί και τα τοποθετούμε σε ένα λεξικό με κλειδί την ηλικία. Στη συνέχεια, ελέγχουμε ένα ένα τα έγκυρα γυναικεία φύλα και κάθε φορά ψάχνουμε τις κοινές ηλικίες, όπου αν βρεθεί κάποια, υπολογίζουμε το σκορ και αν δεν έχουμε ξεπεράσει τα K στοιχεία, τοποθετούμε στην ουρά το σκορ και τα δύο αναγνωριστικά. Επιπλέον, ελέγχουμε εάν το σκορ ξεπερνάει τη μικρότερη τιμή της ουράς και αν το ξεπερνάει, προσθέτουμε το αντικείμενο διαγράφοντας την προηγούμενη οντότητα. Τέλος, ταξινομούμε τα αντικείμενα της ουράς σε φθίνουσα σειρά με κύριο κριτήριο το σκορ και δευτερεύον το αναγνωριστικό του γυναικείου φύλου και τα επιστρέφουμε.
- **Main:** Στο βασικό κομμάτι της συνάρτησης, καλούμε τη συνάρτηση K φορές, όπου K το παίρνουμε από την γραμμή εντολών, και εκτυπώνουμε τα ζευγάρια και τον χρόνο εκτέλεσης του προγράμματος.

Μέρος Τρίτο:
Στο τρίτο μέρος της εργασίας μας ζητήται η σύγκριση των δύο προγραμμάτων σε πολλαπλές τιμές του K ώστε να διακρίνουμε την συμπεριφορά τους και την απόδοσή τους. Για να το επιτύχουμε αυτό με έυκολο τρόπο δημιουργούμε ένα πρόγραμμα σε python το οποίο τρέχει τους δύο αλγορίθμους για K = [1,2,5,10,20,50,100,150,200,250] και αποθηκέυει κάθε φορά σε έναν πίνακα τους χρόνους που απαιτεί να τρέξει τον κάθε αλγόριθμο για κάθε K. Επιπλέον εκτυπώνουμε τα πόσα έγκυρα άτομα, άνδρε και γυναίκες στο σύνολο, πέρασε ωστέ να βρεί τα K καλύτερα ζευγάρια. Τέλος τους χρόνους που αποθηκέψαμε τους κάνουμε ένα απλό διάγραμμα σε σχέση με τα K ζευγάρια.



Από το διάγραμμα παρατειρούμε πως καθώς ο αλγόριθμος hrjn για λίγα ζευγάρια είναι πολύ πιο αποδοτικός σε σχέση με τον δεύτερο καθώς ο χρόνος που απαιτείται για μικρά K είναι απειροελάχιστος. Παρόλα αυτά παρατειρούμε πως όσο αυξανέται το K ο χρόνος του πρώτου αγλορίθμου αυξάνεται εκθετικά, πράγμα που το καθιστά αδύνατο για χρήση με πολλά ζευγάρια. Σε αντίθεση ο δεύτερος αλγόριθμος βλέπουμε πως έχει σχεδόν σταθερό χρόνο για οποιοδήποτε αριθμό K ζευγαριών κάτι που για μικρά K τον κάνει μη βελτιστο και αργό αλλά για μεγάλο αριθμό ζευγαριών ειδανικό.

Οδηγίες Εκτέλεσης για το τρίτο μέρος:
Αφού έχετε αποθηκέυσει τα αρχεία male_sorted,female_sorted στον ίδιο φάκελο καθώς και τα *.py προγράμματα τρέξτε το main.py για να συγκρίνετε τους αλγορίθμους.