A blue and red logo

Description automatically generated

|  |
| --- |
| Λεωνίδας Πάστρας  π20155 |

|  |
| --- |
| **Kείμενο Τεκμηρίωσης Yπολογιστικής Εργασίας** |
| **Αναγνώριση Προτύπων** |
| **Ακαδημαϊκό Έτος 2023 – 2024** |

**Προλογος**

Η υπολογιστικη εργασία υλοποιήθηκε σε γλώσσα python και χρισημοποιήθηκαν οι βιβλιοθήκες

* *numpy*
* *matplotlib*
* *pandas*
* *sklearn*

Επιπλέον το αρχείο **dataVisualisation.ipynb** είναι μορφής *jupyter* notebook

**Προ-επεξεργασία Δεδομένων**

Η προ-επεξεργασία των δεδομένων γίνεται όλη μέσω του αρχείου **data.py**. Μέσα στο αρχείο αυτό υπάρχει η κλάση HousingData η οποία παίρνει ως όρισμα το path του αρχείου housing.csv. Μέσα στην κλάση υπάρχη η μέθοδος Standardize\_Housing\_Data() η οπόια επιστρέφει σε μορφή πίνακα κλιμακωμένα τα δεδομένα του αρχείου **housing.csv**.

1. Θα πρέπει να αναγνωρίσετε τα υποσύνολα των αριθμητικών και των κατηγορικών χαρακτηριστικών

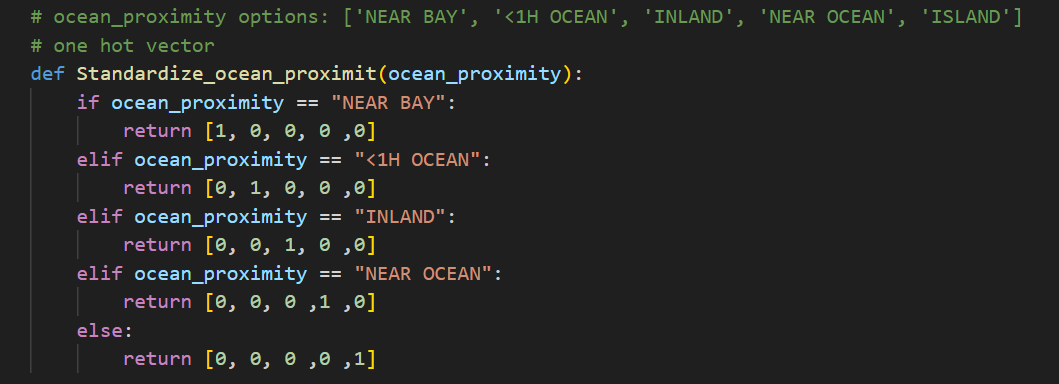
Όλα τα χαρακτηριστικά είναι αριθμιτικά εκτός απο ocean\_proximity το οπόιο είναι κατηγορικό. Συγκεκριμένα χωρίζεται σε 5 κατηγορίες *NEAR BAY, 1<H OCEAN, INLNAD, NEAR OCEAN & ISLAND*

1. Για το υποσύνολο των αριθμητικών χαρακτηριστικών θα πρέπει να πειραματιστείτε με διαφορετικές τεχνικές κλιμάκωσης (scaling) των δεδομένων ώστε όλα τα αριθμητικά χαρακτηριστικά να αναπαρίστανται στην ίδια κλίμακα.

A screen shot of a computer code

Description automatically generatedΗ κλιμάκωση των αριθμιτικών δεδομένων πραγματοποιήθηκε με τον αλγόριθμο *MinMaxScaler*. Δηλαδή ο μετασχηματιμσός των δεδομένων γίνεται με βάση το ελάχιστο και το μέγιστο των αρχικών τιμών τους, δίνοντάς τους νέες τιμές που να είναι ανάλογες μεταξύ τους στο διάστημα [0, 1].

Η temp\_rows περιέχει όλα τα αριθμητικά χαρακτηριστικά και μετά τον μετασχηματιμό σώζονται στην scaled\_data

1. Για το υποσύνολο των κατηγορικών χαρακτηριστικών μπορείτε να χρησιμοποιήσετε την *One Hot Vector* κωδικοποίηση ώστε τα εν λόγω δεδομένα να λάβουν διανυσματική αναπαράσταση.

Η κωδικοκοποίηση *One Hot Vector* χρησιμοποιήθηκε στα κατηγορικά δεδομένα ocean\_proximity. Συγκεκριμένα έγινε μέσω της μεθόδου Standardize\_ocean\_proximity() η οποία παίρνει ως όρισμα μια κατηγορική τιμή και επιστρέφει το αντίστοιχο διάνυσμα.

1. Θα πρέπει να αναγνωρίσετε αν υπάρχουν αριθμητικά χαρακτηριστικά με ελλιπείς τιμές. Για τις συγκεκριμένες εγγραφές μπορείτε να συμπληρώσετε τις τιμές που απουσιάζουν με την διάμεση τιμή του χαρακτηριστικού.

Υπήρχαν ελλιπείς δεδομένα μόνο στην στήλη total\_bedrooms. Αρχικά σώζουμε τα index των άδειων κελιών σε έναν πίνακα, στην συνέχεια υπολογίζουμε την μέση τιμή όλων των μη-κενών κελιών και τέλος συμπληρώνουμε τα κενά κελία με την μέση τιμή που υπολογίσουμε.

**Οπτικοποίηση Δεδομένων**

Η οπτικοποίηση των δεδομένων γίνεται στο αρχείο **dataVisualisation.ipynb** με την βοήθεια της βιβλιοθηκης *matplotlib*.

1. Να αναπαραστήσετε γραφικά τα ιστογράμματα συχνοτήτων (που αντιστοιχούν στις συναρτήσεις πυκνότητας πιθανότητας) για κάθε μία από τις 10 μεταβλητές που εμπλέκονται στο πρόβλημα.

Οι γραφικές αναπαραστήσεις των ιστογράμματων συχνότητας πυκνότητας πιθανότητας είναι τα 10 πρώτα ιστογράμματα που φένονται στο αρχείο και δημιουργούνται με ανάλογο τρόπο:

A screen shot of a graph

Description automatically generated

Παράδειγμα δημιουργείας ιστογράμματος των δεδομένων Median\_House\_Value

1. Σώζουμε τα δεδομένα που θέλουμε να αναπαραστήσουμε γραφικά στην λίστα array. Για να πάρουμε τα συγκεκριμένα δεδομένα κάθε χαρακτηριστηκού καλούμε την ανάλογη μέθοδο απο το αρχείο **dataMethods.py**
2. Ορίζουμε το μέγεθος του
3. Δημιουργούμε το ιστόγραμμα και ορίζουμε το χαρακτηριστηκό του density=True. Με αυτόν τον τρόπο γίνεται αναπαράσταση της συνάρτησης πυκνότητας πιθανότητας των δεδομένων της λίστας array.
4. Προσπαθήστε να δημιουργήσετε δισδιάστατα γραφήματα των δεδομένων στα οποία να αναπαρίστανται με ευδιάκριτο τρόπο συνδυασμοί 2, 3 ή και 4 μεταβλητών.

Σε αυτό το ερώτημα έφτιαξα γραφήματα με σκοπό να καταλάβω συχέτιση μεταξύ τον διαφόρων δεδομένων. Εκτός απο ιστογράμματα έφτιαξα γραφήματα και διαγράμματα διασποράς.

**Παλινδρόμηση Δεδομένων**

1. **Perceptron algorithm**

O κώδικας που υλοποιεί τον αλγόριθμο perceptron χωρίζεται σε τρία αρχεία.

* 1. Class Perceptron\_Dataset

Η κλάση Perceptron\_Dataset βρίσκεται μέσα στο αρχείο **dataMethods.py** και σκοπός της είναι να φέρει τα δεδομένα της κλάσης HousingData στην κατάλληλη μορφή έτσι ώστε να μπορεί να τα χρησιμοποιήσει ο *Perceptron*.

1. A computer code on a black background

   Description automatically generatedΑρχικά χωρίζει τις ετικέτες (labels) των δεδομένων απο τα υπόλοιπα δεδομένα. Και τις σώζει στην λίστα labels αφού πρώτα τις φέρει σε δυαδική μορφή ανάλογα με το αν ξεπερνούν το μέγεθος της μεταβλητής threshold
2. Στην συνέχεια χώρίζει τα δεδομένα σε δύο ομάδες. Η Πρώτη ομάδα περιέχει το ένα δέκατο (1/10) των δεδομένων και χρησιμοποιήται κατα την φάση του έλεγχου του αλγορίθμου. Τα δεδομένα και οι ετικέτες αυτης της ομάδας σώζονται στους πίνακες testing\_data[] & testing\_labels[] αντίστοιχα. Η Δεύτερη ομάδα περιέχει τα υπόλιπα εννέα δέκατα (9/10) των δεδομένων, χρησιμοποιείται για την εκπαίδευση του αλγορίθμου τα οπόια σώζονται επίσης στους πίνακες training\_data[] & training\_labels[]. Τα κομμάτια τον δεδομένων που χρησιμοποιούται στην φάση του ελέγχου και στην φάση της εκπάιδευσης αλλάζουν ανάλογα με το όρισμα fold.
3. Κατα την εισαγωγή των δεδομένων στους πίνακες testing\_data[] και training\_data[] τα διανυσματικά δεδομένα της κατηγορίας ocean\_proximity σώζονται ως πέντε χαρακτηριστηκά το καθένα αντιπροσωπεύοντας μία διάσταση του διανύσματος.
4. Επιπλέον στην αρχή κάθε εγγραφής των πινάκων testing\_data[] και training\_data[] εισάγεται το *bias* (1.0)

Στο τέλος αυτής της διαδικασίας υπάρχουν δύο δισδιάστατοι πίνακες training\_data[] και testing\_data[] διαστάσεων **n x 14** και **(n – n \* 1/10) x 14** αντίστοιχα (οπου n ο αριθμός των δεδομένων και 14 ο αριθμός των χαρακτηριστηκών). Επίσης υπάρχουν και οι δύο μονοδιάστατοι πίνακες training\_labels[] & testing\_labels[] διαστάσεων **n & n – n \* 1/10** αντίστοιχα.

* 1. Class Perceptron

Η κλάση Perceptron βρίσκεται μέσα στο αρχείο **perceptron.py** και υλοποιεί τον αλγόριθμο Perceptron.

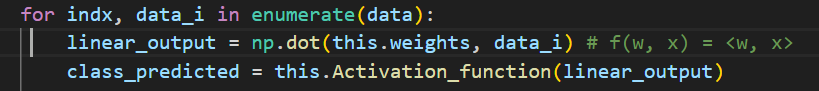
* + 1. def Train\_weights()

Τα βήματα εκπάιδευσης του αλγορίθμου είναι τα εξής

1. Αρχικοποιούμε τα βάρη μας με πολύ μικρές τυχαίες τιμές



1. Και μετά για κάθε δεδομένο του πίνακα training\_data[]
2. Υπολογίζουμε την ετικέτα του

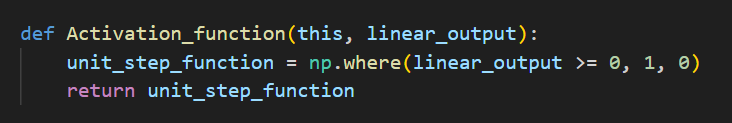


1. Ενημερώνουμε τα βάρη

A black background with white text

Description automatically generated

* + 1. def Activation\_function()

Ως συνάρτηση ενεργοποίησης χρησιμοποιούμε την *Heaviside step function* η οπόια παίρνει σαν όρισμα το εσωτερικό γινόμενο τον βαρώων και των αντίστοιχων δεδομένων και επίστρέφη 1 ή 0 ανάλογα με το πρόσιμό του.

* + 1. def Predict()