ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΙ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΑΘΗΜΑ: ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

ΚΑΘΗΓΗΤΕΣ : ΚΩΣΤΑΣ ΔΙΑΜΑΝΤΑΡΑΣ, ΚΩΣΤΑΣ ΓΟΥΛΙΑΝΑΣ

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 3

ΜΟΝΤΕΛΟ MLP – ΕΚΠΑΙΔΕΥΣΗ BACKPROPAGATION

Σκοπός της άσκησης: Η εκτίμηση της επίδοσης ενός ταξινομητή τύπου Multi-Layer Perceptron με ένα κρυφό στρώμα εκπαιδευόμενου με τον αλγόριθμο Back-Propagation. Θα γίνει χρήση της μεθόδου διασταύρωσης (Cross-Validation).

Θα χρησιμοποιηθούν τα εξής κριτήρια επίδοσης:

1. Ακρίβεια (accuracy)
2. Ευστοχία (precision)
3. Ανάκληση (recall)
4. F-Measure
5. Ευαισθησία (Sensitivity)
6. Προσδιοριστικότητα (Specificity)

Βήματα υλοποίησης:

1. Χρησιμοποιήστε το σύνολο δεδομένων IRIS από το προηγούμενο εργαστήριο, καθώς και τον κώδικα από το εργαστήριο αυτό.
2. Με τη χρήση της εντολής plot δημιουργήστε τη **γραφική παράσταση** των προτύπων **των 3 κλάσεων** με **διαφορετικό σύμβολο και χρώμα για την κάθε κλάση** χρησιμοποιώντας την $1^{η}$ και $3^{η}$ στήλη του πίνακα x, ώστε να τα απεικονίσετε στο χώρο των 2 διαστάσεων και εμφανίστε τα στο ίδιο γράφημα, ώστε να πάρετε μια ιδέα για το πώς είναι η διασπορά των προτύπων στο χώρο των 4 διαστάσεων. Μη ξεχνάτε ότι η αρίθμηση ξεκινάει απ' το 0.
3. Επαυξήστε τον πίνακα των προτύπων προσθέτοντας σε κάθε πρότυπο τον αριθμό 1, δηλαδή προσθέστε μια στήλη με 1 στον πίνακα `x`. Χρησιμοποιήστε τη συνάρτηση `hstack()` του `numpy`.
4. Χρησιμοποιώντας τη συνάρτηση zeros από τη βιβλιοθήκη numpy αρχικοποιήστε τον πίνακα `t` ώστε να είναι γεμάτος μηδενικά και να έχει διάσταση `NumberOfPatterns`.
5. **Εκχωρήστε** στη μεταβλητή ans την τιμή "y'.
6. Για όσο (ans = "y')
   - **Εμφανίστε** το παρακάτω **menu** επιλογών :

     1 **Διαχωρισμός `Iris-setosa`** από `Iris-versicolor και Iris-virginica`

     2 **Διαχωρισμός `Iris-virginica`** από `Iris-setosa και Iris-versicolor`

     3 **Διαχωρισμός `Iris-versicolor`** από `Iris-setosa και Iris-virginica`

     Διαβάστε την επιλογή (1/2/3)

     Αν επιλογή = 1

     Δημιουργήστε ένα dictionary `map_dict` με τα εξής ζευγάρια key/values:

     - `"Iris-setosa": 1`

- `"Iris-versicolor": 0`
- `"Iris-virginica": 0`

Κατόπιν, χρησιμοποιώντας loop θέστε για κάθε pattern την τιμή στόχου `t[pattern]` ως εξής:

`t[pattern] = 1` αν η 5ο στήλη για το `pattern` είναι "`Iris-setosa`"

`t[pattern] = 0` σε διαφορετική περίπτωση

Αν επιλογή = 2

Δημιουργήστε ένα dictionary `map_dict` με τα εξής ζευγάρια key/values:

- `"Iris-setosa": 0`
- `"Iris-versicolor": 0`
- `"Iris-virginica": 1`

Κατόπιν, χρησιμοποιώντας loop θέστε για κάθε pattern την τιμή στόχου `t[pattern]` ως εξής:

`t[pattern] = 1` αν η 5ο στήλη για το `pattern` είναι "`Iris-virginica`"

`t[pattern] = 0` σε διαφορετική περίπτωση

Αν επιλογή = 3

Δημιουργήστε ένα dictionary `map_dict` με τα εξής ζευγάρια key/values:

- `"Iris-setosa": 0`
- `"Iris-versicolor": 1`
- `"Iris-virginica": 0`

Κατόπιν, χρησιμοποιώντας loop θέστε για κάθε pattern την τιμή στόχου `t[pattern]` ως εξής:

`t[pattern] = 1` αν η 5ο στήλη για το `pattern` είναι "`Iris-versicolor`"

`t[pattern] = 0` σε διαφορετική περίπτωση

Μπορείτε να το κάνετε αυτό χρησιμοποιώντας το `map_dict` και να αποφύγετε εντολή `if-else`.

7. Χωρισμός προτύπων σε πρότυπα εκπαίδευσης και ανάκλησης

Τεμαχίστε τα δεδομένα των πινάκων x και t σε 4 πίνακες:

- `xtrain` πίνακας με τα πρότυπα που θα χρησιμοποιηθούν στην εκπαίδευση, `τα 40 πρώτα πρότυπα της κάθε κλάσης.`
- `xtest` πίνακας με τα πρότυπα που θα χρησιμοποιηθούν στον έλεγχο, `τα 10 τελευταία πρότυπα της κάθε κλάσης.`
- `ttrain` διάνυσμα με τους στόχους που θα χρησιμοποιηθούν στην εκπαίδευση, οι 40 πρώτοι στόχοι `της κάθε κλάσης.`
- `ttest` διάνυσμα με τους στόχους που θα χρησιμοποιηθούν στον έλεγχο, οι 10 `τελευταίοι στόχοι της κάθε κλάσης.`

- Χρησιμοποιώντας τη συνάρτηση `plot` από τη βιβλιοθήκη `matplotlib.pyplot` σχεδιάστε

  - τα διανύσματα `xtrain[:,0]` → άξονας x, `xtrain[:,2]` → άξονας y, χρησιμοποιώντας τελείες με μπλε χρώμα και
  - τα διανύσματα `xtest[:,0]` → άξονας x, `xtest[:,2]` → άξονας y, χρησιμοποιώντας τελείες με κόκκινο χρώμα.

  - Μετατρέψτε τα διανύσματα στόχων `ttrain(), ttest()` έτσι ώστε

- Αν `ttrain(pattern) == 1` → `ttrain1(pattern) = 1`
- Αν `ttrain(pattern) == 0` → `ttrain1(pattern) = -1`
- Αν `ttest(pattern) == 1` → `ttest1(pattern) = 1`
- Αν `ttest(pattern) == 0` → `ttest1(pattern) = -1`

- Εμφανίστε το παρακάτω menu επιλογών *αλγορίθμου εκπαίδευσης :*
  1. *Στοχαστική κατάβαση δυναμικού (Stochastic Gradient Descent)*
  2. *LBFGS (Limited memory Broyden–Fletcher–Goldfarb–Shanno)*
  3. *Adam*

- Διαβάστε την επιλογή του solver (`'sgd'` ή `'adam'` ή `'lbfgs'`)

- Εμφανίστε το παρακάτω menu επιλογών *συνάρτησης ενεργοποίησης κρυφού στρώματος:*
  1. 'logistic' – σιγμοειδής 0/1
  2. 'tanh' – υπερβολική εφαπτομένη -1/1

- Διαβάστε την επιλογή της *συνάρτησης ενεργοποίησης του κρυφού στρώματος.*

- Διαβάστε μια ακέραια τιμή για το *πλήθος νευρώνων στο κρυφό στρώμα* (*hidden_layer_sizes*)
- Διαβάστε μια ακέραια τιμή για το Μέγιστο Αριθμό Εποχών (*max_iter*)

- Δημιουργήστε ένα δίκτυο MLP δύο στρωμάτων χρησιμοποιώντας την κλάση
- `MLPClassifier(`*hidden_layer_sizes*, *activation*, *solver*, *learning_rate*, *learning_rate_init*, *max_iter*, *momentum=0.9*, *…*`)` όπου:
  - *hidden_layer_sizes=πλήθος νευρώνων στο κρυφό στρώμα*
  - *activation = επιλογή συνάρτηση ενεργοποίησης μεταξύ* 'identity', 'logistic', 'tanh', 'relu'.
  - *solver = επιλογή αλγορίθμου εκπαίδευσης, μεταξύ των*

    Στοχαστική κατάβαση δυναμικού (Stochastic Gradient Descent)
    LBFGS (Limited memory Broyden–Fletcher–Goldfarb–Shanno)
    Adam
  - *max_iter = πλήθος εποχών*
  - *learning_rate, learning_rate_init = βήμα εκπαίδευσης και αλγόριθμος ρύθμισης βήματος εκπαίδευσης (συνήθως 'constant')*
  - *momentum = ορμή (εφόσον αλγόριθμος εκπαίδευσης είναι sgd)*

- Εκπαιδεύστε το δίκτυο που φτιάξατε χρησιμοποιώντας τη συνάρτηση
  - `fit()` : με εισόδους το μοντέλο, τον πίνακα των προτύπων εκπαίδευσης (`xtrain`), και το διάνυσμα των στόχων εκπαίδευσης (`ttrain`)

- Αφού εκπαιδεύσετε το μοντέλο κάνετε ανάκληση χρησιμοποιώντας τη συνάρτηση
  - `predict()` : με εισόδους το εκπαιδευμένο μοντέλο και τον πίνακα των προτύπων εκπαίδευσης (`xtrain`)

- Μετατρέψτε την έξοδο που θα λάβετε σε τιμές $0/1$ ή $-1/1$ ανάλογα με τους στόχους που έχετε, χρησιμοποιώντας κάποιο κατάλληλο κατώφλι (0.5 ή 0). Ονομάστε $predict_{\text{train}}$ το διάνυσμα που πήρατε.

- Τυπώστε το εξής γράφημα:
  - δείξτε με μπλε τελείες τους πραγματικούς στόχους $t_{train}\,[i]$ για όλα τα πρότυπα του $train$ set
  - δείξτε με κόκκινους κύκλους τους εκτιμώμενους στόχους $predict_{train}\,[i]$ για όλα τα πρότυπα του $train$ set.
- Κάνετε ανάκληση χρησιμοποιώντας τη συνάρτηση
  - `predict()` : με εισόδους το εκπαιδευμένο μοντέλο και τον πίνακα των προτύπων ελέγχου (`xtest`)
- Μετατρέψτε την έξοδο που θα λάβετε σε τιμές $0/1$ ή $-1/1$ ανάλογα με τους στόχους που έχετε, χρησιμοποιώντας κάποιο κατάλληλο κατώφλι (0.5 ή 0). Ονομάστε $predict_{test}$ το διάνυσμα που πήρατε.
- Τυπώστε το εξής γράφημα:
  - δείξτε με μπλε τελείες τους πραγματικούς στόχους $t_{test}\,[i]$ για όλα τα πρότυπα του test set
  - δείξτε με κόκκινους κύκλους τους εκτιμώμενους στόχους $predict_{test}\,[i]$ για όλα τα πρότυπα του test set

8. Θα εφαρμοστεί η μέθοδος `train_test_split(...)` για K=9 folds. Στο αντίστοιχο loop θα πρέπει να κάνετε τα εξής:

Για κάθε *fold:*

- Έχετε ήδη δημιουργήσει τους πίνακες `xtrain, xtest` καθώς και τα διανύσματα στόχων `ttrain, ttest.` Φροντίστε να είναι `numpy arrays` τύπου `float.`
- Βρείτε το πλήθος των προτύπων στο train set (P_{train}) και στο test set (P_{test}) για παράδειγμα χρησιμοποιώντας τη συνάρτηση `len()`.
- Μετατρέψτε τα διανύσματα στόχων `ttrain(), ttest()` έτσι ώστε
  - Αν `ttrain(pattern) == 1` → `ttrain1(pattern) = 1`
  - Αν `ttrain(pattern) == 0` → `ttrain1(pattern) = -1`
  - Αν `ttest(pattern) == 1` → `ttest1(pattern) = 1`
  - Αν `ttest(pattern) == 0` → `ttest1(pattern) = -1`
- Εκπαιδεύστε το δίκτυο που φτιάξατε χρησιμοποιώντας τη συνάρτηση
  - `fit()` : με εισόδους το μοντέλο, τον πίνακα των προτύπων εκπαίδευσης (`xtrain`), και το διάνυσμα των στόχων εκπαίδευσης (`ttrain`)
- Αφού εκπαιδεύσετε το μοντέλο κάνετε ανάκληση χρησιμοποιώντας τη συνάρτηση
  - `predict()` : με εισόδους το εκπαιδευμένο μοντέλο και τον πίνακα των προτύπων ελέγχου (`xtest`)
- Μετατρέψτε την έξοδο που θα λάβετε σε τιμές $0/1$ ή $-1/1$ ανάλογα με τους στόχους που έχετε, χρησιμοποιώντας κάποιο κατάλληλο κατώφλι (0.5 ή 0). Ονομάστε $predict_{test}$ το διάνυσμα που πήρατε.
- Καλέστε τη συνάρτηση `evaluate()` όσες φορές χρειάζεται έτσι ώστε για το συγκεκριμένο fold να υπολογίσετε το Accuracy, Precision, Recall, F-measure, Sensitivity και Specificity.

- Χρησιμοποιώντας κατάλληλο subplot σε grid 3x3 στο figure(1) τυπώστε το εξής γράφημα:
  - δείξτε με μπλε τελείες τους πραγματικούς στόχους $t_{test}[i]$ για όλα τα πρότυπα του test set
  - δείξτε με κόκκινους κύκλους τους εκτιμώμενους στόχους $predict_{test}[i]$ για όλα τα πρότυπα του test set

```
# end for
```

Μετά το τέλος του loop υπολογίστε και τυπώστε στην οθόνη τα εξής:

1. τη μέση τιμή του Accuracy για όλα τα folds
2. τη μέση τιμή του Precision για όλα τα folds
3. τη μέση τιμή του Recall για όλα τα folds
4. τη μέση τιμή του F-Measure για όλα τα folds
5. τη μέση τιμή του Sensitivity για όλα τα folds
6. τη μέση τιμή του Specificity για όλα τα folds

Διαβάστε την απάντηση ans του χρήστη, αν θέλετε να συνεχίσετε.

Οδηγίες κατάθεσης ασκήσεων

1. Συνδεθείτε στο URL: http://aetos.it.teithe.gr/s
1. Επιλέξτε το μάθημα “Μηχανική Μάθηση – Εργαστήριο Χ” (Όπου Χ ο αριθμός του εργαστηρίου του οποίου τις ασκήσεις πρόκειται να καταθέσετε) και πατήστε επόμενο.
2. Συμπληρώστε τα στοιχεία σας. Πληκτρολογείστε USERNAME 00003 και PASSWORD 30000 (Επώνυμο και Όνομα με ΛΑΤΙΝΙΚΟΥΣ ΧΑΡΑΚΤΗΡΕΣ).
3. Αν θέλετε να καταθέσετε μόνο ένα αρχείο μη το βάζετε σε zip file. Αντίθετα, αν θέλετε να καταθέσετε περισσότερα από ένα αρχεία, τοποθετήστε τα σε ένα zip ή rar file.
4. Επιλέξτε το αρχείο που θέλετε να στείλετε επιλέγοντας “choose file” στο πεδίο FILE1 και πατήστε “Παράδοση”