ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΙ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΑΘΗΜΑ: ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

ΚΑΘΗΓΗΤΕΣ : ΚΩΣΤΑΣ ΔΙΑΜΑΝΤΑΡΑΣ, ΚΩΣΤΑΣ ΓΟΥΛΙΑΝΑΣ

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 6

ΤΑΞΙΝΟΜΗΣΗ ΜΕ ΤΟ ΜΟΝΤΕΛΟ NAIVE BAYES

Σκοπός της άσκησης: Η εκτίμηση της επίδοσης ενός ταξινομητή τύπου **Naïve Bayes** χρησιμοποιώντας την **Γκαουσσιανή κατανομή**. Θα γίνει χρήση της μεθόδου διασταύρωσης (Cross-Validation) και τα 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. Θυμίζουμε ότι τα πρότυπα χωρίστηκαν σε δύο κλάσεις ως εξής:
   - Κλάση 0 (στόχος t=0): αποτελείται από τα πρότυπα των κατηγοριών "Iris-setosa"+"Iris-virginica",
   - Κλάση 1 (στόχος t=1): αποτελείται από τα πρότυπα της κατηγορίας "Iris-versicolor"

2. Θα εφαρμοστεί η μέθοδος `train_test_split()` για K=9 folds.

3. Δείτε πώς λειτουργεί το μοντέλο Naïve Bayes διαβάζοντας τα παρακάτω:

   Η μέθοδος ταξινόμησης Naïve Bayes.pdf

   Μηχανική Μάθηση - 05 Bayes.pptx

4. Στο Cross-Validation loop θα πρέπει να κάνετε τα εξής:

   Για κάθε *fold*

   - Έχετε ήδη δημιουργήσει τους αρχικούς πίνακες προτύπων `xtrain` και `xtest` (χωρίς επαύξηση) καθώς και τα διανύσματα στόχων `ttrain` και `ttest`. Χρησιμοποιήστε τιμές των στόχων 0/1.
   - Εκπαιδεύστε ένα μοντέλο Naive Bayes κάνοντας την υπόθεση ότι τα χαρακτηριστικά ακολουθούν την Γκαουσσιανή κατανομή. Θα χρησιμοποιήσετε την συνάρτηση `nbtrain(xtrain,ttrain)` την οποία θα πρέπει να γράψετε εσείς.

```
def nbtrain( x, t ):
# Είσοδος x : Pxn πίνακας με τα πρότυπα (P=πλήθος προτύπων, n=διάσταση)
# Είσοδος t : διάνυσμα με τους στόχους (0/1)
# Έξοδος model : dictionary που θα περιέχει τις παραμέτρους του μοντέλου
```

Ο αλγόριθμος εκπαίδευσης του μοντέλου NB λειτουργεί ως εξής:

- Χωρίστε τα πρότυπα στην κλάση 0 και στην κλάση 1 (Χρησιμοποιήστε στον πίνακα x κατάλληλα δείκτες `t==0` και `t==1`)
- Βρείτε το πλήθος των προτύπων σε κάθε κλάση
- Υπολογίστε τις εκ των προτέρων πιθανότητες (prior) των δύο κλάσεων (δηλ. πλήθος προτύπων στην κλάση δια το συνολικό πλήθος των προτύπων)
- Για κάθε χαρακτηριστικό `i` (στήλη του πίνακα x) υπολογίστε
  - μ`[0,i]` = μέση τιμή του χαρακτηριστικού `i` για την κλάση 0 (Χρησιμοποιήστε τη συνάρτηση numply.mean)
  - σ`[0,i]` = διασπορά του χαρακτηριστικού `i` για την κλάση 0 (Χρησιμοποιήστε τη συνάρτηση numpy.std)
  - μ`[1,i]` = μέση τιμή του χαρακτηριστικού `i` για την κλάση 1
  - σ`[1,i]` = διασπορά του χαρακτηριστικού `i` για την κλάση 1
- `# end for`

Δημιουργήστε το dictionary "`model`" που θα περιέχει τα εξής πεδία:

- όνομα '`prior`', τιμή `prior`: array 2 στοιχείων με τις εκ των προτέρων πιθανότητες των 2 κλάσεων
- όνομα '`mu`', τιμή μ: array 2xn με τις μέσες τιμές των n χαρακτηριστικών για τις 2 κλάσεις
- όνομα '`sigma`', τιμή σ: array 2xn με τις διασπορές των n χαρακτηριστικών για τις 2 κλάσεις

- Αφού εκπαιδεύσατε το μοντέλο με την παραπάνω συνάρτηση κάνετε ανάκληση χρησιμοποιώντας τη συνάρτηση `nbpredict(xtest, model)` την οποία επίσης πρέπει να γράψετε.

```
def nbpredict( x, model ):
# Είσοδος x : Pxn πίνακας με τα πρότυπα
# Είσοδος model : dictionary με τις παραμέτρους του μοντέλου NB
# Έξοδος predict : διάνυσμα με τις εκτιμώμενες τιμές στόχου
```

- Για κάθε πρότυπο p (γραμμή του πίνακα x)
  - Υπολογίζουμε το λόγο των πιθανοτήτων L. Αρχικά θέτουμε

$$L = \frac{\text{prior}[1]}{\text{prior}[0]}$$

  - Για κάθε χαρακτηριστικό `i` (στήλη του πίνακα x)
    - Ενημερώνουμε το L:

$$L \leftarrow L * \frac{G(x[p,i], mu[1,i], sigma[1,i])}{G(x[p,i], mu[0,i], sigma[0,i])}$$

    - Όπου $G(x, \mu, \sigma)$ είναι η συνάρτηση της Γκαουσσιανής κατανομής με μέση τιμή μ και διασπορά σ. (Χρησιμοποιήστε τη συνάρτηση `norm.pdf(x, loc=μ, scale=σ)` αφού πρώτα την κάνετε `import: from scipy.stats import norm`)
  - `# end for`
  - Αν $L < 1$ τότε εκτιμάμε ότι το πρότυπο p ανήκει στην κλάση 0
  - Αν $L > 1$ τότε εκτιμάμε ότι το πρότυπο p ανήκει στην κλάση 1
- `# end for`

- Το διάνυσμα που πήρατε στην έξοδο είναι το $predict_{test}$.
- Καλέστε τη συνάρτηση `evaluate()` από το προηγούμενο εργαστήριο 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
```

5. 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