Μηχανική Μάθηση: Εργασία 1

Τμήμα Πληροφορικής και Επικοινωνιών
Πανεπιστήμιο Πελοποννήσου

19 Μαρτίου 2026
Λήξη Προθεσμίας: 2 Απριλίου 2026

Θέμα 1 (k-Κοντινότεροι Γείτονες vs. Λογιστική Παλινδρόμηση) (0.4 μονάδες)

Η αναγνώριση αντικειμένων σε εικόνες είναι ένα κλασικό πρόβλημα κατηγοριοποίησης στην υπολογιστική όραση. Ένα από τα πιο δημοφιλή σύνολα δεδομένων αναγνώρισης αντικειμένων είναι το MNIST το οποίο αφορά την αναγνώριση χειρόγραφων ψηφίων. Το συγκεκριμένο σύνολο δεδομένων δημιουργήθηκε από τους Yann LeCun, Corinna Cortes και Christopher Burges για την αξιολόγηση μοντέλων μηχανικής μάθησης. Κάθε παράδειγμα είναι μια εικόνα διάστασης 28×28 εικονοστοιχείων (δηλαδή 784 εικονοστοιχεία συνολικά). Τα ψηφία που εμφανίζονται στις εικόνες είναι κανονικοποιημένα σε μέγεθος και κεντραρισμένα. Υπάρχουν συνολικά 10 ψηφία (από 0 έως 9), τα οποία και αντιστοιχούν στις 10 κλάσεις του συνόλου δεδομένων. Το σύνολο δεδομένων είναι χωρισμένο σε ένα σύνολο εκπαίδευσης που αποτελείται από $60,000$ εικόνες (πίνακας `X_train`) με τις ετικέτες τους (διάνυσμα `y_train`) και ένα σύνολο ελέγχου $10,000$ εικόνων (πίνακας `X_test`) με τις ετικέτες τους (διάνυσμα `y_train`).

(α) Εφαρμόστε τους αλγορίθμους k-κοντινότερων γειτόνων και λογιστικής παλινδρόμησης στο σύνολο δεδομένων MNIST. Συγκεκριμένα, συμπληρώστε τον κώδικα που λείπει στο αρχείο `knn_vs_logreg.py`, έτσι ώστε να υλοποιήσετε τα παρακάτω:

1. Εκπαιδεύστε τους αλγορίθμους k-κοντινότερων γειτόνων και λογιστικής παλινδρόμησης στο σύνολο εκπαίδευσης του συνόλου δεδομένων MNIST. Μπορείτε να χρησιμοποιήσετε τις υλοποιήσεις των αλγορίθμων από το πακέτο scikit-learn.
2. Έπειτα, χρησιμοποιήστε τους δυο αλγορίθμους για να κάνετε προβλέψεις για τα δεδομένα ελέγχου.
3. Υπολογίστε την ακρίβεια (accuracy) των προβλέψεων των δυο αλγορίθμων για τα δεδομένα ελέγχου και συγκρίνετε τις δυο ακρίβειες. Ποιός από τους δυο αλγορίθμους έχει καλύτερη απόδοση;

Προσοχή: Οι αλγόριθμοι k-κοντινότερων γειτόνων και λογιστικής παλινδρόμησης λειτουργούν με δεδομένα σε μορφή διανυσμάτων. Συνεπώς, θα πρέπει να μετατρέψετε τις εικόνες διάστασης 28×28 σε διανύσματα 784 διαστάσεων. Με άλλα λόγια, οι πίνακες `X_train` και `X_test` διάστασης $(60,000 \times 28 \times 28)$ και $(10,000 \times 28 \times 28)$ θα πρέπει να μετατραπούν σε πίνακες διαστάσεων $(60,000 \times 784)$ και $(10,000 \times 784)$, αντίστοιχα. Αυτό μπορείτε να το πραγματοποιήσετε με χρήση της μεθόδου `reshape()` της βιβλιοθήκης NumPy.

(β) Μετρήστε το χρόνο εκτέλεσης των δυο αλγορίθμων. Θα περιμένατε ο αλγόριθμος k-κοντινότερων γειτόνων να πάρει περισσότερο χρόνο από όσο παίρνει στην πράξη; Πόσες Ευκλείδειες αποστάσεις πρέπει να υπολογίσει συνολικά για να κάνει προβλέψεις για τις $10,000$ εικόνες του συνόλου ελέγχου; Αν ο αλγόριθμος παίρνει στην πράξη λιγότερο χρόνο από ότι αναμένατε, που πιστεύετε ότι οφείλεται αυτό;

Θέμα 2 (Χάρτες Χαρακτηριστικών) (0.4 μονάδες)

Όπως αναφέραμε στο μάθημα, το μοντέλο της γραμμικής παλινδρόμησης μπορεί να συλλάβει μόνο γραμμικές σχέσεις μεταξύ των εισόδων και της εξόδου. Γραμμικά μοντέλα μπορούν να μετατραπούν σε μη γραμμικά με χρήση χαρτών χαρακτηριστικών. Εστω το μοντέλο μη γραμμικής παλινδρόμησης το οποίο χρησιμοποιεί Gaussian ακτινικές συναρτήσεις βάσης ως χάρτες χαρακτηριστικών. Το μοντέλο έχει την παρακάτω μορφή:

$$\hat{y} = \sum_{j=1}^{d} w_j \phi_j(\mathbf{x}) + b \qquad \text{με } \phi_j(\mathbf{x}) = e^{-\frac{1}{2\ell^2}\|\mathbf{x}-\boldsymbol{\mu}_j\|^2}$$

όπου ℓ είναι μια υπερπαράμετρος των Gaussian ακτινικών συναρτήσεων βάσης και $\boldsymbol{\mu}_j$ είναι ο μέσος της j-οστής συνάρτησης βάσης.

Συμπληρώστε τον κώδικα που λείπει στο αρχείο `radial_basis.py` για να εφαρμόσετε την παλινδρόμηση με Gaussian ακτινικές συναρτήσεις βάσης. Συγκεκριμένα, πρέπει να υλοποιήσετε τα παρακάτω:

1. Εκπαιδεύστε τον αλγόριθμο γραμμικής παλινδρόμησης στα αρχικά δεδομένα (πίνακας `X_train`). Μπορείτε να χρησιμοποιήσετε την υλοποίηση του αλγορίθμου από τη βιβλιοθήκη scikit-learn.

2. Χρησιμοποιήστε τον αλγόριθμο για να κάνετε προβλέψεις για τα δεδομένα ελέγχου (πίνακας `X_test`).

3. Οπτικοποιήστε τα δεδομένα εκπαίδευσης και τις ετικέτες τους στο διδιάστατο χώρο (με χρήση της μεθόδου `scatter()` του πακέτου `Matplotlib`), καθώς και τα δεδομένα ελέγχου με τις αντίστοιχες προβλέψεις που κάνατε (με χρήση της μεθόδου `plot()` του πακέτου `Matplotlib`). Τι παρατηρείτε;

4. Υλοποιήσετε την μέθοδο `gaussian_rbf_features(X, mu, ell)` η οποία επιστρέφει έναν πίνακα $\phi(\mathbf{X})$ του οποίου οι γραμμές αντιστοιχούν σε παραδείγματα και οι στήλες σε Gaussian ακτινικές συναρτήσεις βάσης.

5. Εκπαιδεύστε έναν νέο αλγόριθμο γραμμικής παλινδρόμησης στα δεδομένα που περιέχουν τα χαρακτηριστικά που προέκυψαν από τις Gaussian ακτινικές συναρτήσεις βάσης (πίνακας `phi_train`).

6. Χρησιμοποιήστε τον αλγόριθμο για να κάνετε προβλέψεις για τα δεδομένα ελέγχου (πίνακας `phi_test`).

7. Οπτικοποιήστε τα δεδομένα εκπαίδευσης και τις ετικέτες τους στο διδιάστατο χώρο (με χρήση της μεθόδου `scatter()` του πακέτου `Matplotlib`), καθώς και τα δεδομένα ελέγχου με τις αντίστοιχες προβλέψεις που κάνατε (με χρήση της μεθόδου `plot()` του πακέτου `Matplotlib`). Τι παρατηρείτε;

Θέμα 3 (Γραμμική Παλινδρόμηση με Ομαλοποίηση) (0.5 μονάδες)

Στη μάθημα είδαμε ότι τα μοντέλα παλινδρόμησης με πολυωνυμικά χαρακτηριστικά μπορεί να προσαρμοστούν πολύ καλά στα δεδομένα εκπαίδευσης και να αποτύχουν να γενικεύσουν σε νέα δεδομένα. Είδαμε επίσης ότι ένας τρόπος για να βελτιώσουμε τη γενίκευση είναι η χρήση ℓ_2 ομαλοποίησης, δηλαδή να προσθέσουμε έναν όρο στη συνάρτηση κόστους που προσπαθεί να διατηρήσει τα βάρη σχετικά μικρά σε μέγεθος:

$$\mathcal{R}(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\sum_{i=1}^{d} w_i^2$$

Θεωρώντας ότι η πόλωση b έχει ενσωματωθεί στο διάνυσμα των βαρών $\mathbf{w}$ και μια στήλη με τιμή 1 έχει προστεθεί στον πίνακα δεδομένων $\mathbf{X}$, η ομαλοποιημένη συνάρτηση κόστους ορίζεται ως:

$$\mathcal{J}_{reg}(\mathbf{w}) = \mathcal{J}(\mathbf{w}) + \lambda\mathcal{R}(\mathbf{w}) = \frac{1}{2}\|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

(α) Η γραμμική παλινδρόμηση με ομαλοποίηση έχει άμεση λύση η οποία προκύπτει υπολογίζοντας τις μερικές παραγώγους και θέτοντάς τες ίσες με μηδέν. Η άμεση λύση είναι η ακόλουθη:

$$\mathbf{w}^* = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}$$

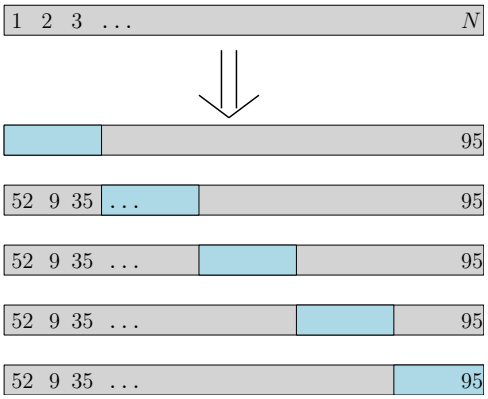
Δείξτε πώς προκύπτει η παραπάνω λύση. Παρουσιάστε αναλυτικά όλα τα βήματα.

(β) Προσδιορίστε τους κανόνες ενημέρωσης του αλγορίθμου κατάβασης κλίσης για την συνάρτηση κόστους με ομαλοποίηση. Η απάντησή σας θα πρέπει να δείχνει πως αλλάζει κάθε στοιχείο του διανύσματος $\mathbf{w}$ σε κάθε επανάληψη του αλγορίθμου κατάβασης κλίσης, καθώς και πώς ακριβώς προέκυψε ο κανόνας.

Θέμα 4 (Διασταυρωμένη Επικύρωση K-Τμημάτων) (0.5 μονάδες)

Στο μάθημα είδαμε ότι για να βρούμε τις βέλτιστες τιμές υπερπαραμέτρων (όπως η παράμετρος ομαλοποίησης λ), χωρίζουμε το σύνολο δεδομένων σε ένα σύνολο εκπαίδευσης, ένα σύνολο επικύρωσης και ένα σύνολο ελέγχου, και αποφασίζουμε τις τιμές των υπερπαραμέτρων ανάλογα με την απόδοση στο σύνολο επικύρωσης.

Ωστόσο, αυτό προϋποθέτει ότι έχουμε μεγάλο αριθμό από δεδομένα, κάτι το οποίο μπορεί να μην συμβαίνει στην πράξη. Μια λύση σε αυτό το πρόβλημα είναι η διασταυρωμένη επικύρωση K-τμημάτων (K-fold cross-validation), η οποία χρησιμοποιεί ένα κομμάτι των διαθέσιμων δεδομένων για να εκπαιδεύσει το μοντέλο και ένα διαφορετικό κομμάτι για να ελέγξει την απόδοσή του. Συγκεκριμένα, η διασταυρωμένη επικύρωση K-τμημάτων χωρίζει με τυχαίο τρόπο τα δεδομένα σε K κομμάτια ίσου μεγέθους όπως φαίνεται παρακάτω:



Πριν την εφαρμογή της διασταυρωμένης επικύρωσης για την επιλογή της τιμής κάποιας υπερπαραμέτρου, είναι σημαντικό να εξάγουμε από τα δεδομένα ένα σύνολο ελέγχου και να μην το χρησιμοποιήσουμε καθόλου μέχρι να ολοκληρωθούν οι διαδικασίες εκπαίδευσης και ρύθμισης υπερπαραμέτρων. Το συγκεκριμένο σύνολο θα χρησιμοποιηθεί για την τελική αξιολόγηση του μοντέλου. Στην περίπτωσή μας, το συγκεκριμένο σύνολο παρέχεται ως ξεχωριστό σύνολο δεδομένων (πίνακας `X_test` και διάνυσμα `y_test`).

Στη συνέχεια, για κάθε τιμή της υπερπαραμέτρου που θέλουμε να δοκιμάσουμε (π.χ., τιμές της υπερπαραμέτρου ομαλοποίησης λ) υπολογίζουμε το σφάλμα διασταυρωμένης επικύρωσης. Δηλαδή, χωρίζουμε τα δεδομένα εκπαίδευσης σε K κομμάτια και για τιμές του $k = 1, 2, \ldots, K$ χρησιμοποιούμε το k-οστό κομμάτι ως σύνολο επικύρωσης και τα υπόλοιπα $K - 1$ κομμάτια ως σύνολο εκπαίδευσης. Επομένως, εκπαιδεύουμε και επικυρώνουμε τον αλγόριθμό μας K φορές. Το σφάλμα διασταυρωμένης επικύρωσης για τη συγκεκριμένη επιλογή υπερπαραμέτρου είναι ίσο με το μέσο σφάλμα σε αυτά τα K κομμάτια. Αφότου υπολογίσουμε το σφάλμα διασταυρωμένης επικύρωσης για κάθε τιμή της υπερπαραμέτρου που θέλουμε να δοκιμάσουμε, επιλέγουμε ως τελική τιμή της υπερπαραμέτρου αυτή που οδήγησε στο μικρότερο σφάλμα διασταυρωμένης επικύρωσης.

Στα πλαίσια της εργασίας, θα πρέπει να υλοποιήσετε στο αρχείο `cv.py` τη διασταυρωμένη επικύρωση K-τμημάτων για να βρείτε την βέλτιστη τιμή της υπερπαραμέτρου ομαλοποίησης λ στην παλινδρόμηση κορυφογραμμής (ridge regression). Συγκεκριμένα, θα πρέπει να χρησιμοποιήσετε τον παρακάτω κώδικα.

```
X, y = shuffle_data(X, y)
cv_error = []
for lamda in lamdas:
    cv_loss = 0
    for fold in range(n_folds):
        X_train, y_train, X_val, y_val = split_data(X, y, n_folds, fold)
        y_pred = train_model_and_predict(X_train, y_train, X_val, lamda)
        cv_loss += mean_squared_error(y_val, y_pred)
    cv_error.append(cv_loss / n_folds)

best_lamda = lamdas[np.argmin(cv_error)]
```

Για να τρέξει ο παραπάνω κώδικας, πρέπει να υλοποιήσετε τις παρακάτω μεθόδους:

- `shuffle_data(X, y)`: Η συγκεκριμένη μέθοδος ανακατεύει τα δεδομένα. Δηλαδή κάνει μια τυχαία μετάθεση των γραμμών του πίνακα `X` και με αντίστοιχο τρόπο αλλάζουν θέσεις και τα στοιχεία του διανύσματος `y`. Μπορείτε να πάρετε μια τυχαία μετάθεση N ακεραίων με τη μέθοδο `permutation()` της βιβλιοθήκης NumPy.

- `split_data(X, y, n_folds, fold)`: Η μέθοδος αυτή εξάγει τα σύνολα εκπαίδευσης και επικύρωσης από τον πίνακα `X` και το διάνυσμα `y`. Συγκεκριμένα, το κομμάτι του πίνακα και το κομμάτι του διανύσματος με δείκτη `fold` αποτελούν το σύνολο επικύρωσης (δηλαδή τα `X_val` και `y_val`), ενώ τα υπόλοιπα κομμάτια του πίνακα `X` και του διανύσματος `y` αποτελούν το σύνολο εκπαίδευσης (δηλαδή τα `X_train` και `y_train`).

- `train_model_and_predict(X_train, y_train, X_val, lamda)`: Η μέθοδος αυτή εκπαιδεύει ένα μοντέλο παλινδρόμησης κορυφογραμμής με υπερπαράμετρο λ ίση με `lamda` χρησιμοποιώντας τα δεδομένα εκπαίδευσης και χρησιμοποιεί το μοντέλο για να κάνει προβλέψεις για τα δεδομένα του συνόλου επικύρωσης. Μπορείτε να χρησιμοποιήσετε την υλοποίηση της παλινδρόμησης κορυφογραμμής από το πακέτο scikit-learn. Σημειώστε ότι η υπερπαράμετρος λ αναφέρεται ως `alpha` στη συγκεκριμένη υλοποίηση.

- `mean_squared_error(y_val, y_pred)`: Υπολογίζει το μέσο τετραγωνικό σφάλμα των προβλέψεων για το σύνολο επικύρωσης. Μπορείτε να χρησιμοποιήσετε την ομώνυμη μέθοδο από το πακέτο scikit-learn για να υπολογίσετε το σφάλμα.

Μετά το πέρας της παραπάνω διαδικασίας, εκπαιδεύστε εκ νέου ένα μοντέλο παλινδρόμησης κορυφογραμμής με τιμή υπερπαραμέτου λ ίση με αυτή που οδήγησε στο χαμηλότερο σφάλμα διασταυρωμένης επικύρωσης (`best_lamda` στον παραπάνω κώδικα). Στη συνέχεια, χρησιμοποιήστε το μοντέλο για να κάνετε προβλέψεις για το σύνολο ελέγχου (πίνακας `X_test`) και υπολογίστε το μέσο τετραγωνικό σφάλμα στα δεδομένα του συγκεκριμένου συνόλου.

Ποιό είναι το σημαντικότερο μειονέκτημα της διασταυρωμένης επικύρωσης K-τμημάτων;

Θέμα 5 (Αξιολόγηση Αλγορίθμων Κατηγοριοποίησης) (0.2 μονάδες)

Έχετε αναπτύξει ένα σύστημα μηχανικής μάθησης το οποίο ανιχνεύει αν τα e-mails που λαμβάνετε είναι spam ή όχι. Υποθέστε ότι έχετε αξιολογήσει το μοντέλο σε ένα σύνολο ελέγχου το οποίο περιέχει τόσο spam όσο και μη spam e-mails. Προκύπτει ότι το μοντέλο ανίχνευσε με επιτυχία 200 e-mails ως spam. Ωστόσο, 50 e-mails τα οποία στην πραγματικότητα ήταν spam δεν ανιχνεύτηκαν ως spam, αλλά ταξινομήθηκαν λανθασμένα ως μη spam. Όσον αφορά τα μη spam e-mails, 730 από αυτά όντως δεν τα θεώρησε το μοντέλο ως spam. Δυστυχώς, το μοντέλο λανθασμένα κατηγοριοποίησε τα υπόλοιπα 20 μη spam e-mails ως spam.

(α) Υπολογίστε τον αριθμό των αληθώς θετικών (TP), αληθώς αρνητικών (TN), ψευδώς θετικών (FP) και ψευδώς αρνητικών (FN) παραδειγμάτων.

(β) Υπολογίστε την ακρίβεια του μοντέλου πρόβλεψης, καθώς και τις ευαισθησία και ειδικότητα.

(γ) Μπορούμε με βάση τις ποσότητες που υπολογίσαμε παραπάνω να σχεδιάσουμε την καμπύλη ROC του συστήματος; Εξηγήστε.

Διευκρινίσεις:

- Μπορείτε να δουλέψετε είτε μόνοι σας ή σε ομάδες 2 ατόμων.
  ⚠ Εργασίες με 3 ή περισσότερα άτομα δεν θα βαθμολογούνται.

- Μην ξεχάσετε να προσθέστε τα ονόματα των μελών της ομάδας σας στην κορυφή του jupyter notebook ή της αναφοράς που θα παραδώσετε.

- Για να βαθμολογηθούν τα θέματα που ζητούν την ανάπτυξη κώδικα, πρέπει ο κώδικας να εκτελείται και να παρουσιάζει αποτέλεσμα.

- Για τα θέματα που ζητούν την υλοποίηση αλγορίθμων, μπορείτε να μεταφέρετε τον κώδικα στο Google Colab και να παραδώσετε το/τα jupyter notebooks που θα προκύψουν. Τυχόν ερωτήσεις που υπάρχουν στα συγκεκριμένα θέματα θα πρέπει να απαντηθούν είτε στο jupyter notebook ή στην αναφορά. Αν δεν απαντηθούν οι ερωτήσεις, δεν θα λάβετε όλες τις μονάδες του θέματος.

- Η χρήση μεγάλων γλωσσικών μοντέλων ή chatbots για την απάντηση οποιουδήποτε θέματος δεν

συνίσταται. Αν χρησιμοποιηθεί κάποιο τέτοιο εργαλείο, να αναφερθεί πως ακριβώς έγινε η χρήση του.

- Τα αρχεία σας θα πρέπει να υποβληθούν αποκλειστικά στην πλατφόρμα e-class, στην ενότητα Εργασίες, από ένα μόνο μέλος της ομάδας σας.
- Δημιουργήστε ένα μοναδικό αρχείο .zip (⚠ όχι .rar) που περιέχει όλα τα αρχεία σας (π.χ., αναφορά, κώδικα) εκτός των δεδομένων και υποβάλετε το συγκεκριμένο αρχείο στην πλατφόρμα e-class. Σε περίπτωση που χρησιμοποιείτε το Google Colab, μπορείτε να κατεβάσετε τον κώδικα ως ένα αρχείο .ipynb από το Google Colab πατώντας File → Download → Download .ipynb.