Τμήμα Πληροφορικής και Τηλεματικής
Χαροκόπειο Πανεπιστήμιο
ΕΠ34 Μηχανική Μάθηση και Εφαρμογές

1η Άσκηση: Γραμμικά μοντέλα

Έκδοση 1.0

Διδάσκων: Χρήστος Δίου

1 Εισαγωγή

Στην άσκηση αυτή θα υλοποιήσουμε και θα δοκιμάσουμε μία πολύ απλή εκδοχή ενός μοντέλου γραμμικής παλινδρόμησης σε γλώσσα Python. Η άσκηση σας καθοδηγεί βήμα-βήμα στην υλοποίηση.

Παραδοτέα

Για την υποβολή της εργασίας, θα χρειαστεί να υποβληθεί ο κώδικας σε γλώσσα python, με επαρκή σχόλια και μια συνοπτική αναφορά για τον τρόπο εκτέλεσης του κώδικα και τυχόν λεπτομέρειες σχετικά με την υλοποίηση. Ο κώδικας θα πρέπει να μπορεί να εκτελεστεί σε περιβάλλον python3 με πρόσφατες εκδόσεις των βιβλιοθηκών `numpy` και `scikit-learn`. Παρακαλείστε να υλοποιήσετε την άσκηση μόνοι σας. Σε περίπτωση που διαπιστωθεί ότι βασιστήκατε σε υλοποίηση άλλων, η υποβολή σας θα μηδενιστεί.

2 Υλοποίηση μοντέλου γραμμικής παλινδρόμησης

Δημιουργήστε το αρχείο `linear_regression.py` το οποίο θα υλοποιεί την κλάση `LinearRegression` που υλοποιεί ένα μοντέλο γραμμικής παλινδρόμησης. Για την υλοποίηση πρέπει να χρησιμοποιήσετε τη βιβλιοθήκη `numpy` της python και προαιρετικά και την `pandas` (αλλά καμία άλλη). Η `LinearRegression` έχει τις εξής ιδιότητες (attributes)

- `w` : Διάνυσμα numpy το οποίο αντιστοιχεί στα βάρη, $\mathbf{w}$, του μοντέλου
- `b` : Αριθμός που αντιστοιχεί στον όρο μεροληψίας, b, του μοντέλου

Προαιρετικά, μπορείτε να προσθέσετε και άλλες ιδιότητες στο μοντέλο σας, όπως το πλήθος των δεδομένων N με τα οποία εκπαιδεύτηκε, ή η διάσταση p του χώρου των χαρακτηριστικών. Οι ιδιότητες της κλάσης μπορούν να αρχικοποιούνται με **`None`** (στην `__init__()`).

Οι μέθοδοι της `LinearRegression` περιγράφονται στα ακόλουθα.

2.1 `fit(self, X, y)`

Η `fit` δέχεται στην είσοδο έναν $N \times p$ πίνακα σχεδιασμού, `X`, κάθε γραμμή του οποίου αντιστοιχεί σε ένα διάνυσμα χαρακτηριστικών του συνόλου εκπαίδευσης, και ένα $N \times 1$ διάνυσμα τιμών εξόδου `y`. Ο πίνακας και τα διανύσματα να είναι numpy arrays. Η συνάρτηση πρέπει να κάνει τα ακόλουθα:

- Να επιβεβαιώνει ότι τα `X` και `y` είναι numpy arrays και ότι οι μεταξύ τους διαστάσεις είναι συμβατές. Σε διαφορετική περίπτωση πρέπει να ενεργοποιεί **`ValueError`** exception (ή κάποιο άλλο αντίστοιχο)

- Να δημιουργεί έναν νέο πίνακα σχεδιασμού που είναι ο $\mathbf{X}$ που είδαμε στο μάθημα, ο οποίος έχει τα στοιχεία του πίνακα εισόδου και μία επιπλέον στήλη $p+1$ η οποία έχει όλα τα στοιχεία ίσα με 1
- Να υπολογίζει τις παραμέτρους $\boldsymbol{\theta}$ λύνοντας τις κανονικές εξισώσεις:

$$\boldsymbol{\theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

  Δείτε τη συνάρτηση `numpy.dot()` για τον πολλαπλασιασμό πινάκων, `numpy.linalg.inv()` για την αντιστροφή πίνακα. Ο εκθέτης ^{T} συμβολίζει τον ανάστροφο του πίνακα
- Να αποθηκεύει στην ιδιότητα `w` της κλάσης το διάνυσμα $\boldsymbol{\theta}$ από το πρώτο μέχρι το προτελευταίο στοιχείο
- Να αποθηκεύει στην ιδιότητα `b` της κλάσης το τελευταίο στοιχείο του διανύσματος $\boldsymbol{\theta}$

Η `fit` δεν επιστρέφει κάτι.

2.2 `predict(self, X)`

Η `predict` δέχεται στην είσοδο έναν πίνακα σχεδιασμού και επιστρέφει τις τιμές πρόβλεψης του γραμμικού μοντέλου, ή σφάλμα αν το μοντέλο δεν έχει εκπαιδευτεί. Συγκεκριμένα η `predict` υλοποιεί την πράξη

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w} + b$$

2.3 `evaluate(self, X, y)`

Η `evaluate` δέχεται στην είσοδο έναν πίνακα σχεδιασμού καθώς και τις τιμές της εξαρτημένης μεταβλητής $\mathbf{y}$. Αν το μοντέλο δεν έχει εκπαιδευτεί επιστρέφει σφάλμα. Για ένα εκπαιδευμένο μοντέλο κάνει τα εξής:

- Καλεί την `predict` ώστε να προβλέψει τις τιμές εξόδου του μοντέλου
- Υπολογίζει το μέσο τετραγωνικό σφάλμα

$$MSE = \frac{1}{N}(\hat{\mathbf{y}} - \mathbf{y})^T(\hat{\mathbf{y}} - \mathbf{y}) \tag{1}$$

- Επιστρέφει το ζεύγος $(\hat{\mathbf{y}}, MSE)$

3 Εφαρμογή στο California Housing Dataset

3.1 Προετοιμασία δεδομένων

Τώρα που υλοποιήσατε την κλάση `LinearRegression` μπορείτε να τη δοκιμάσετε στο σύνολο δεδομένων California Housing Dataset. Μπορείτε να βρείτε μια περιγραφή του συνόλου δεδομένων εδώ:
https://developers.google.com/machine-learning/crash-course/california-housing-data-description
Επίσης, το σύνολο δεδομένων είναι διαθέσιμο μέσω του `sklearn.datasets` (βρείτε μέσω της τεκμηρίωσης ποια συνάρτηση πρέπει να καλέσετε ώστε να κατεβάσετε και να φορτώσετε το σύνολο δεδομένων). Χρησιμοποιήστε αυτή την εκδοχή, καθώς σε αυτή την περίπτωση όλη η προεπεξεργασία των δεδομένων έχει γίνει για εσάς.
Ακολουθήστε τα παρακάτω βήματα

- Ετοιμάστε ένα αρχείο `test_lr.py` το οποιο χρησιμοποιεί το `sklearn.datasets` module για να φορτώσει το σύνολο δεδομένων
- Χωρίστε το σύνολο σε 70% τυχαία επιλεγμένων δειγμάτων για εκπαίδευση, και το υπόλοιπο 30% για δοκιμή. Μπορείτε αν θέλετε να χρησιμοποιήσετε τη συνάρτηση `train_test_split`. Χρησιμοποιείστε παράμετρο `random_state=42` (είναι το seed) για λόγους επαναληψιμότητας και σύγκρισης των πειραμάτων

- Δημιουργήστε ένα αντικείμενο της κλάσης `LinearRegression` που δημιουργήσατε στην προηγούμενη ενότητα
- Καλέστε την `fit` και την `evaluate` για το σύνολο εκπαίδευσης και δοκιμής αντίστοιχα και εκτυπώστε τη ρίζα του μέσου τετραγωνικού σφάλματος (Root Mean Squared Error - RMSE)

3.2 Μέση τιμή και διασπορά σφάλματος

Αφού κάνετε τα παραπάνω, γράψτε κώδικα ο οποίος επαναλαμβάνει την παραπάνω διαδικασία 20 φορές, για διαφορετικά σύνολα εκπαίδευσης και δοκιμής. Σε κάθε επανάληψη κρατήστε σε έναν πίνακα την τιμή του RMSE. Ο κώδικας θα πρέπει να εκτυπώνει τη μέση τιμή και την τυπική απόκλιση του RMSE.

3.3 Έλεγχος έναντι των αντίστοιχων συναρτήσεων του scikit-learn

Υλοποιήστε ένα παρόμοιο πρόγραμμα με τα παραπάνω ώστε να παράξετε τα ίδια αποτελέσματα, αλλά αυτή τη φορά με την κλάση `LinearRegression` του scikit-learn. Συγκρίνετε και σχολιάστε τα αποτελέσματα των δύο μεθόδων.