MEM704 - Μηχανική Μάθηση
2η Εργαστηριακή Άσκηση
Η Γραμμική και Λογιστική παλινδρόμηση
Linear and Logistic Regression

Παράδοση: Πέμπτη 14/04/2022, 18:00. Εξέταση 15/04/2022 11:00

Στη άσκηση αυτή θα υλοποιήσουμε τις μεθόδους της Γραμμικής και Λογιστικής παλινδρόμησης.

1 Γραμμική Παλινδρόμηση(ΓΠ)

Θα μελετήσουμε την *αποδοτικότητα του αυτοκινήτου* με κριτήριο τα χιλιομέτρα ανά λίτρο βενζίνης (km/l) με βάση τις παραμέτρους : *ισχυς(hp), βάρος(kg), αριθμό κυλίνδρων(cyl), κυβισμός μηχανής(cm^3)*. Οι παράμετροι αυτοί αποτελούν τα *χαρακτηριστικά (features)* $x \in \mathbb{R}^5,\ x = (1,\ hp,\ kg,\ cyl,\ cm^3)$ στο μοντέλο μάθησης της Γραμμικής Παλινδρόμησης: $h_\theta(x) = \theta^T x, \theta \in \mathbb{R}^5$. Ο προσδιορισμός των παραμέτρων $\theta \in \mathbb{R}^5$ θα γίνει με την ελαχιστοποίηση της συνάρτησης κόστους $J(\theta)$

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2.$$

1.1 Υλοποίηση αλγορίθμου

Η προσέγγιση του ελάχιστου θα γίνει με την μέθοδο μεγίστης κλίσης. Για την *εκπαίδευση* της συνάρτησης μάθησης θα χρησιμοποιηθεί το αρχείο $car\_train.txt$ του οποίου οι 4 πρώτες στήλες αντιστοιχούν στα χαρακτηριστικά $x^{(i)}$ ενώ η 5η στο *στόχο* $y^{(i)},\ i = 1, \ldots, n$. Στη συνέχεια θα γίνει έλεγχος της ποιότητας του μοντέλου μάθησης μέσω της μετρικής $\mathcal{E}_\theta = \|h_\theta(\hat{x}) - \hat{y}\|_2$, όπου $\hat{x},\ \hat{y}$ τα δεδομένα του αρχείου $car\_test.txt$.

Να γραφεί ένας κώδικας Python, που θα υπολοποιεί μέθοδο μεγίστης κλίσης για την ελαχιστοποίηση του $J(\theta)$ και θα ελέγχει την ποιότητα του μοντέλου με τον υπολογισμό του σφάλματος $\mathcal{E}_\theta$. Ο κώδικας σας θα τερματίζει είτε αν $|J(\theta)| < \delta$ ή αν $\|\theta^{k+1} - \theta^k\|_1\ < \epsilon$ με $\epsilon, \delta \ll 1$. Ο ρυθμός μάθησης μπορεί να είναι είτε σταθερός είτε μεταβλητός. Στο τέλος ο κώδικας σας θα: α) τυπώνει το διάνυσμα θ, το ρυθμό μάθησης που χρησιμοποιήθηκε, τον αριθμό επαλήψεων που απαιτήθηκαν για τις δοσμένες τιμές των ϵ, δ, και το σφάλμα $\mathcal{E}_\theta$, β) γράφημα με την εξέλιξη του $J(\theta^k),\ k = 1, \ldots,$.

Συγκρίνεται τα αποτελέσματά σας με αυτά που θα πάρετε χρησιμοποιώντας τη κλάση *LinearRegression* της βιβλιοθήκης *Scikit-Learn*, `https://scikit-learn.org/stable/modules/linear_model.html`.

2 (Μη-)Γραμμική Παλινδρόμηση

Θα εξετάσουμε τώρα πως μπορούμε να χρησιμοποιήσουμε την ΓΠ για να προσαρμόσουμε μη-γραμμικές συναρτήσεις των *χαρακτηριστικών (features)* με κατάλληλες απεικονίσεις.

1. **Μαθαίνοντας ένα πολύωνυμο 3ου βαθμού**

   'Εστω το σύνολο δεδομένων $\{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$ where $x^{(i)}, y^{(i)} \in \mathbb{R}$. Θέλουμε να προσαρμόσουμε στα δεδομένα ένα πολυώνυμο 3ου βαθμού $h_\theta(x) = \theta_3 x^3 + \theta_2 x^2 + \theta_1 x^1 + \theta_0$. Παρατηρούμε ότι η $h_\theta(x)$ είναι γραμμική ως προς θ, παρόλο που είναι μη-γραμμική ως προς x, το οποίο μας επιτρέπει να χρησιμοποιήσουμε την ΓΠ ως εξής: έστω $\phi : \mathbb{R} \to \mathbb{R}^4$ η οποία απεικονίζει το αρχικό διάνυσμα x σε ένα διάνυσμα στον $\phi(x) \in \mathbb{R}^4$

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \end{bmatrix} \in \mathbb{R}^4 \tag{1}$$

Έστω $\hat{x} \in \mathbb{R}^4$ με $\hat{x} := \phi(x)$, και $\hat{x}^{(i)} \triangleq \phi(x^{(i)})$ το μετασχηματισμένο σύνολο δεδομένων. Φτιάχνουμε ένα καινούργιο σύνολο δεδομένων $\{(\phi(x^{(i)}), y^{(i)})\}_{i=1}^n = \{(\hat{x}^{(i)}, y^{(i)})\}_{i=1}^n$ αντικαθιστώντας τα αρχικά $x^{(i)}$ με τα $\hat{x}^{(i)}$. Οπότε το να προσαρμόσουμε την $h_\theta(x)$ στα αρχικά δεδομένα είναι ισοδύναμο με το να προσαρμόσουμε την $h_\theta(\hat{x}) = \theta_3\hat{x}_3 + \theta_2\hat{x}_2 + \theta_1\hat{x}_1 + \theta_0$ στο νέο σύνολο δεδομένων γιατί

$$h_\theta(x) = \theta_3 x^3 + \theta_2 x^2 + \theta_1 x + \theta_0 = \theta_3\phi(x)_3 + \theta_2\phi(x)_2 + \theta_1\phi(x)_1 + \theta_0 = \theta^T\hat{x} = h_\theta(\hat{x})$$

δηλαδή μπορούμε να χρησιμοποιήσουμε ΓΠ στο νέο σύνολο δεδομένων για να βρούμε τις παραμέτρους $\theta_0, \ldots, \theta_3$. Να γραφτεί, 1) η συνάρτηση κόστους $J(\theta)$ της ΓΠ για τον νέο σύνολο $\{(\hat{x}^{(i)}, y^{(i)})\}_{i=1}^n$ και 2) ο αλγόριθμος μεγίστης κλίσης για το $\{(\hat{x}^{(i)}, y^{(i)})\}_{i=1}^n$.

2. **Υλοποίηση αλγορίθμου.** Να γραφτεί ένας κώδικας Python ο οποίος θα υλοποίει τα παραπάνω και θα βρίσκει τις παραμέτρους θ με ΓΠ χρησιμοποιώντας τα δεδομένα εκπαίδευσης *f_train.txt* και ελέγχου *f_test.txt*. Ο υπολογισμός του θ θα γίνει με την μέθοδο μεγίστης κλίσης. Ο κώδικας θα τυπώνει το σφάλμα $\mathcal{E}_\theta$ και σε ενα γράφημα τα δεδομένα σαν απλά σημεία και την $h_\theta(\hat{x})$ σαν ομαλή καμπύλη.

3. **Υπερεκτίμηση(Overfitting).** Εφαρμόζουμε την παραπάνω ιδέα για πολυώνυμα $k-$βαθμού, θεωρώντας $\phi : \mathbb{R} \to \mathbb{R}^{k+1}$

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^k \end{bmatrix} \in \mathbb{R}^{k+1}$$

Ακολουθήστε την παραπάνω διαδικασία και υλοποιήστε τον αλγόριθμο με $k = 3, 5, 10, 20$ χρησιμοποιώντας τα δεδομένα του αρχείου *f_small.txt*. Φτιάξτε ένα παρόμοιο γράφημα όπως προηγουμένος με την κάθε καμπύλη να έχει διαφορετικό χρώμα για κάθε τιμή του k και με την κατάλληλη λεζάντα. Παρατηρήστε τι συμβαίνει καθώς αυξάνει η τιμή του k.

3 Λογιστική Παλινδρόμηση

Θα δούμε τώρα τον διακριτικό ταξινομητή της γραμμικής παλινδρόμησης. Ο αλγόριθμος αυτός υπολογίζει το *σύνορο απόφασης* το οποίο χωρίζει το σύνολο των δεδομένων σε δύο κατηγορίες. Θεωρούμε δυο σύνολα δεδομένων εκπαίδευσης $set1\_train.txt$, $set2\_train.txt$ και τα αντίστοιχα σύνολα ελέγχου $set1\_test.txt$, $set2\_test.txt$. Καθένα από τα αρχεία περιέχει $n-$δείγματα της μορφής $(x^{(i)}, y^{(i)})$ και ειδικότερα η κάθε γραμμή περιέχει $x_1^{(i)} \in \mathbb{R}$, $x_2^{(i)} \in \mathbb{R}$, και $y^{(i)} \in \{0, 1\}$. Θα χρησιμοποιήσουμε την λογιστική παλινδρόμηση για να κάνουμε δυαδική ταξινόμηση σε αυτά τα δεδομένα.

3.1 Συνάρτηση κόστους

Όπως είδαμε η μέση συνάρτηση κόστους της λογιστικής παλινδρόμησης είναι

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)}\log(h_\theta(x^{(i)})) + (1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))\right),$$

όπου $y^{(i)} \in \{0, 1\}$, $h_\theta(x) = g(\theta^T x)$ and $g(z) = 1/(1 + e^{-z})$. Βρείτε την Hessian H της $J(\theta)$.

3.2 Υλοποίηση αλγορίθμου

Σε ένα κώδικα Python υλοποίειστε την μέθοδο Newton για την εκτίμηση του θ και για τα δύο σύνολα εκπαίδευσης. Με αρχική συνθήκη $\theta = 0$ η μέθοδος Newton να τερματίζει όταν οι διαδοχικές επαναλήψεις να είναι κοντά: $\|\theta_{k+1} - \theta_k\|_1 < \epsilon$ με $\epsilon = 10^{-4},\ 10^{-5}$. Ο κωδικάς σας να τυπώνει τις πιθανότητες που προβλέπει για τα δύο σύνολα ελέγχου.

Επίσης φτιάξτε ένα γράφημα με τα δεδομένα με άξονες τα $x_1,\ x_2$. Για να διακρίνεται τις δύο κλάσεις χρησιμοποιείστε διαφορετικά σύμβολα και χρώματα για τα δείγματα $x^{(i)}$ με $y^{(i)} = 0$ από αυτά με $x^{(i)}$ με $y^{(i)} = 1$. Στο ίδιο γράφημα να υπάρχει και το *σύνορο απόφασης* που υπολογίζει η μέθοδος το οποίο αντιστοιχεί στη γραμμή με $p(y|x) = 0.5$.

Συγκρίνετε τα αποτελέσματα σας με τα αυτά που θα σας δώσει η αντίστοιχη συνάρτηση της λογιστικής παλινδρόμησης *LogisticRegression* της βιβλιοθήκης *Scikit-Learn*, `https://scikit-learn.org/stable/modules/linear_model.html`.

4 Κανονικοποίηση Δεδομένων

Στη ΜΜ είναι απαραίτητη η κανονικοποίηση των δεδομένων εκπαίδευσης και ελέγχου πριν την χρήση τους. Ο λόγος είναι ότι οι πιθανές μεγάλες διαφορές κλίμακας ανάμεσα στα χαρακτηριστικά του προβλήματος θα έχουν ως αποτέλεσμα την αργή ή μη-σύγκλιση της μεθόδου μεγίστης κλίσης. Υπάρχουν διάφοροι τρόποι για την κανονικοποίηση των δεδομένων προτού χρησιμοποιηθούν για την εκπαίδευση του μοντέλου. Δύο από τους πιο συνηθισμένους είναι: α) **Διαίρεση με το μέγιστο**: για το $j-$χαρακτηριστικό $\{x_j^{(i)}\}$ βρίσκουμε το μέγιστο $M_j = \max_{1\le i\le n} x_j^{(i)}$ και θέτουμε $\tilde{x}_j^{(i)} = x_j^{(i)}/M_j,\ i = 1,\ldots,n$, β) **Κανονική κατανομή**: για το $j-$χαρακτηριστικό $\{x_j^{(i)}\}$ βρίσκουμε τα μ_j, σ_j μέσο όρο και τυπική απόκλιση αντίστοιχα. Στη συνέχεια ορίζουμε $\tilde{x}_j^{(i)} = (x_j^{(i)} - \mu_j)/\sigma_j,\ \forall j$ το οποίο έχει ως συνέπεια $\{\tilde{x}_j^{(i)}\} \sim \mathcal{N}(0,1),\ \forall j$. Αντίστοιχη κανονικοποίηση ακολουθείται και για τους στόχους $\{y_j^{(i)}\},\ \forall j$. Η εκπαίδευση και ο έλεγχος γίνεται πλέον για τα κανονικοποιημένα δεδομένα $\{\tilde{x}_j^{(i)},\ \tilde{y}_j^{(i)}\}$. Μπορείτε να συμβουλευτείτε και το αντίστοιχο κεφάλαιο στο εγχειρίδιο της βιβλιοθήκης *Scikit-Learn* : `https://scikit-learn.org/stable/data_transforms.html`.

5 Παράδοση - Εξέταση

Για κάθε μέρος της άσκησης να φτιάξετε διαφορετικό κώδικα Python με όνομα π.χ: $\{math, tem\}XXXX\_Lab2\{a,b,c\}.py$ όπου $XXXX$ είναι ο αριθμός μητρώου σας. Επίσης στις πρώτες γραμμές του κάθε προγράμματος θα υπάρχουν σαν σχόλιο τα στοιχεία σας: όνομα, επώνυμο και ΑΜ. Θα στείλετε τους κώδικες (ως συνημμένα αρχεία) με email από τον **ιδρυματικό σας λογαριασμό** στη διεύθυνση $mem704labs@gmail.com$ το αργότερο μέχρι 18:00, Πέμπτη 14 Απριλίου. Εκπρόθεσμες ασκήσεις δεν θα βαθμολογηθούν. Εργαστείται ατομικά. Κώδικες που είναι προιόν αντιγραφής θα μηδενίζονται.