Οπτική Αναγνώριση Ψηφίων

1η Εργαστηριακή Άσκηση
για το μάθημα της "Αναγνώρισης Προτύπων"

Νίκος Σταμάτης
Μεταπτυχιακός Φοιτητής (ΕΔΕΜΜ)
ΑΜ: 03400115
nikolaosstamatis@mail.ntua.gr

***Περίληψη*—Σκοπός αυτής της εργασίας είναι η οπτική αναγνώριση ψηφίων με τη χρήση αλγορίθμων μηχανικής μάθησης. Το πρώτο μέρος είναι αφιερωμένο στην ανάγνωση των δεδομένων, την εξαγωγή κάποιων χρήσιμων χαρακτηριστικών τους και τη κατασκευή διαγραμμάτων. Στο δεύτερο μέρος υλοποιούμε και συγκρίνουμε τις επιδόσεις διαφόρων ταξινομητών. Επιπλέον, υλοποιούμε from scratch έναν δικό μας Naive Bayes ταξινομητή βασισμένο στη Βήτα κατανομή.**

I Προπαρασκευή

Το σύνολο των δεδομένων εκπαίδευσης αποτελείται από συνολικά 7291 δείγματα, καθένα εκ των οποίων αποτελείται από 256 χαρακτηριστικά (εικόνες 16×16).

***Βήματα 2,3**: Σχεδιασμός ψηφίων.*

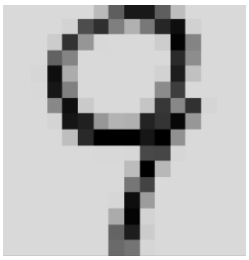
Στο Σχήμα 1 σχεδιάσαμε το ψηφίο που βρίσκεται στη θέση 131 του πίνακα (δηλαδή το 132ο ψηφίο). Στο Σχήμα 2, επιλέξαμε και σχεδιάσαμε τυχαία ένα δείγμα από κάθε ψηφίο. Πρώτα διαμερίσαμε το training set σε 10 υποσύνολα, ένα για το κάθε ψηφίο, και εν συνεχεία επιλέξαμε ένα σημείο από το καθένα τυχαιοποιώντας.

***Βήματα 4,5**: Μέση τιμή και διασπορά pixel (10,10) για το ψηφίο 0.*

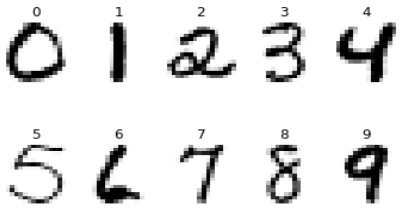
Η μέση τιμή υπολογίστηκε μέσω της συνάρτησης digit_mean_at_pixel και βρέθηκε ίση με -0.93. Αντίστοιχα η διασπορά είναι ίση με 0.084.

***Βήμα 6**: Μέση τιμή και διασπορά όλων των χαρακτηριστικών για το ψηφίο 0.*

Τα δύο αυτά διανύσματα, μήκους 256 το καθένα, υπολογίστηκαν μέσω των συναρτήσεων digit_mean και digit_variance άλλα για λόγους οικονομίας χώρου, δε θα τους εκτυπώσουμε σε αυτήν την αναφορά.



Σχήμα 1: Σχεδιασμός του ψηφίου 131.



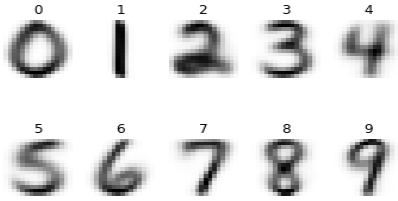
Σχήμα 2: Μια τυχαία επιλογή ψηφίων από το training σετ δεδομένων.

***Βήματα 7,8,9**: Σχεδιασμός ψηφίων με βάση τη μέση τιμή και τη διασπορά.*

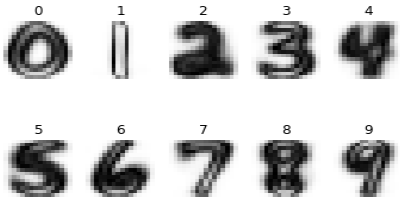
Οι μέσες τιμές των χαρακτηριστικών για το κάθε ψηφίο καταχωρίστηκαν σε έναν κατάλληλο πίνακα με το όνομα X_my_mean. Ο πίνακας αυτός μπορεί να χρησιμοποιηθεί σε αυτό το βήμα για το σχεδιασμό των ψηφίων, αλλά θα παίξει και σημαντικό ρόλο κατά την υλοποίηση του Ευκλείδειου ταξινομητή.
Ανάλογα κατασκευάσαμε έναν αντίστοιχο πίνακα με τις διασπορές. Τα σχέδια των ψηφίων φαίνονται στα Σχήματα 3 και 4. Παρατηρούμε ότι τα σχεδιαγράμματα που βασίστηκαν στις διασπορές, περιγράφουν αρκετά έντονα το σύνορο των αντίστοιχων διαγραμμάτων που βασίστηκαν στις μέσες τιμές.

***Βήματα 10, 11**: Ταξινόμηση βάση της Ευκλείδειας απόστασης.*

Η κατηγοριοποίηση των ψηφίων με τον ευκλείδειο ταξι-



Σχήμα 3: Σχεδιασμός ψηφίων χρησιμοποιώντας τις μέσες τιμές των χαρκτηριστικών.



Σχήμα 4: Σχεδιασμός ψηφίων χρησιμοποιώντας τις διασπορές των χαρκτηριστικών.

Σχήμα 5: Ταξινόμηση των ψηφίων 100-102 βάση του Ευκλείδειου ταξινομητή.

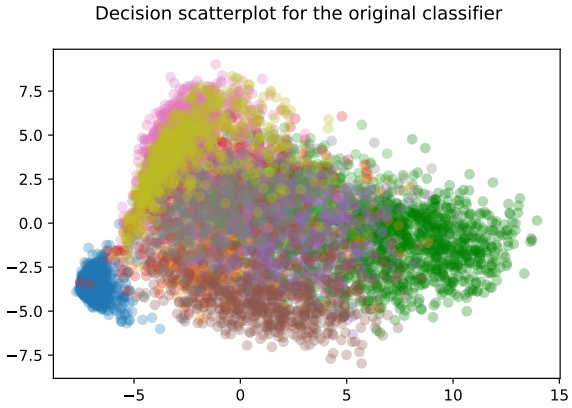
νομητή πραγματοποιήθηκε χρησιμοποιώντας τη συνάρτηση euclidean_distance_classifier και το σκορ της στα τεστ δεδομένα υπολογίστηκε στο 81.4%.
Στο Σχήμα 5 σχεδιάσαμε το ψηφίο που βρίσκεται στην 101η θέση (το μεσαίο από τα τρία). Όπως βλέπουμε, ενώ το πραγματικό ψηφίο είναι 6, ο ταξινομητής μας το κατηγοριοποίησε εσφαλμένα ως 0.

***Βήμα 13**: Cross Validation, περιοχή απόφασης Ευκλείδειου ταξινομητή.*

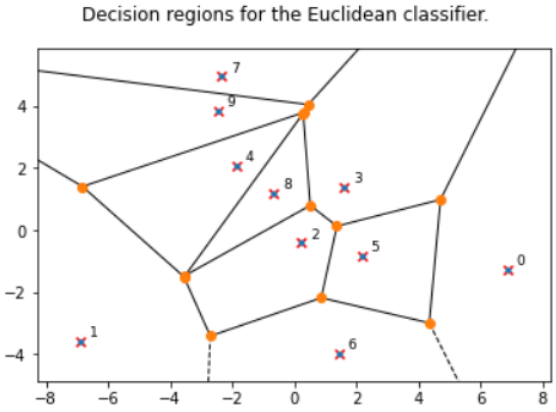
α) Εφαρμόσαμε τη cross_val_score στην κλάση EuclideanDistanceClassifier του Βήματος 12 (βλ. κώδικα). Το cv-score βρέθηκε ίσο με 84.9%.

β) Η επιφάνεια απόφασης είναι αδύνατο να απεικονιστεί πλήρως καθώς τα δεδομένα μας ζουν στον $\mathbb{R}^{256}$. Για το λόγο αυτό, επιλέξαμε να προβάλλουμε τα δεδομένα μας σε έναν κατάλληλο διδιάστατο υπόχωρο χρησιμοποιώντας Principal Component Analysis. Σε αυτόν τον υπόχωρο σχεδιάσαμε ένα διάγραμμα διασποράς των δεδομένων μας (Σχήμα 6). Προφανώς υπάρχουν επικαλύψεις ανάμεσα στις διάφορες κατηγορίες.

Στη συνέχεια ξαναπροσαρμόσαμε το μοντέλο μας, χρησιμοποιώντας μόνο τα 2 χαρακτηριστικά που κρατήσαμε μέσω PCA. Πλέον τα δεδομένα μας ζουν σε ένα διδιάστατο χώρο, επομένως μπορούμε να απεικονίσουμε τις περιοχές απόφασεις επακριβώς, χρησιμοποιώντας διαγράμματα Voronoi (Σχήμα 7). Φυσικά το καινούργιο μοντέλο χάνει πάρα πολύ σε ακρίβεια.



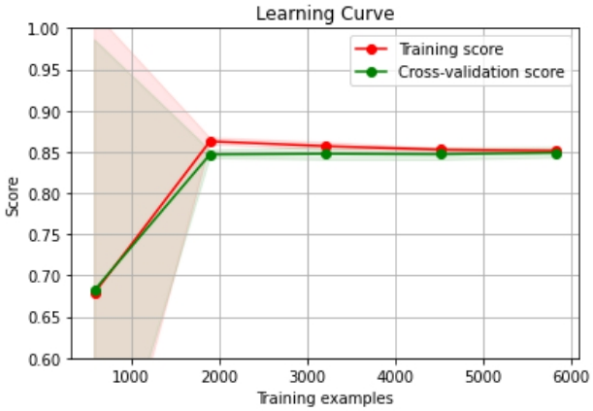
Σχήμα 6: Προβολή των αρχικών δεδομένων στο διδιάστατο υπόχωρο που υπέδειξε η PCA, κατηγοριοποίησή τους ανά ψηφίο και απεικόνισή τους σε διάγραμμα διασποράς.



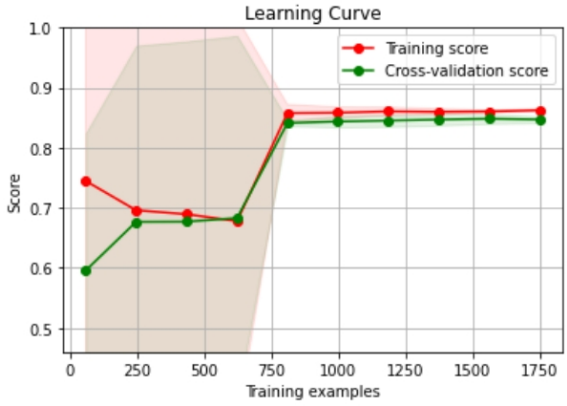
Σχήμα 7: Ταξινόμηση των αρχικών δεδομένων βάση μόνο των 2 χαρακτηριστικών που κρατάει η PCA. Ακριβής σχεδιασμός των περιοχών απόφασης ως διαγραμμάτων Voronoi με κέντρα P_i τις μέσες τιμές των δύο χαρακτηριστικών για τα ψηφία $i = 0,\ldots,9$.

γ) Η καμπύλη εκμάθησης (Σχήμα 8) σχεδιάστηκε με τη συνάρτηση plot_learning_curve του εργαστηρίου. Στο δεύτερο διάγραμμα (Σχήμα 9) εστιάσαμε στις πρώτες 1800 παρατηρήσεις ώστε να απεικονιστεί καλύτερα η σύγκλιση της μεθόδου. Παρατηρούμε ότι περίπου μετά από 750 training samples, η μέθοδός μας πιάνει τη μέγιστη ακρίβειά της.

Το σχετικά μικρό πλήθος παραδειγμάτων που απαιτούνται δεν είναι ιδιαίτερα απροσδόκητο. Ο Ευκλείδειος ταξινομητής βασίζει την ταξινόμησή του επάνω στις μέσες τιμές των χαρακτηριστικών. Ένα δείγμα που του δίνει τη δυνατότητα να εκτιμήσει τους μέσους με αποδε-



Σχήμα 8: Η καμπύλη εκμάθησης του Ευκλείδειου ταξινομητή για όλες τις παρατηρήσεις του training set.



Σχήμα 9: Η καμπύλη εκμάθησης του Ευκλείδειου ταξινομητή για για τις πρώτες 1800 παρατηρήσεις του training set.

κτή ακρίβεια, αρκεί ώστε να δώσει καλά αποτελέσματα. Ως γνωστόν, ακόμα και μικρά δείγματα της τάξης του $N = 20$, αρκούν για τις περισσότερες εφαρμογές ώστε ο δειγματικός μέσος να προσεγγίζει ικανοποιητικά τον πραγματικό.

Εδώ χρειάζονται λίγα παραπάνω, λόγω της ύπαρξης πολλών χαρακτηριστικών και κλάσεων. Για ένα μόνο χαρακτηριστικό, ένα δείγμα μεγέθους $N \geq 1.96^2 \frac{\sigma^2}{\delta^2}$ αρκεί ώστε να κατασκευαστεί ένα 95% δ.ε. $[\overline{x} - \delta, \overline{x} + \delta]$ για τον μέσο. Παρουσία l το πλήθος χαρακτηριστικών και d κλάσεων, ένα διάστημα εμπιστοσύνης που αντιπροσωπεύει πιθανότητα p για κάθε επιμέρους χαρακτηριστικό, οδηγεί στη σχέση $p^l p^{10l} \geq 0.95$, για όλα τα χαρακτηριστικά ταυτόχρονα. Από τη σχέση αυτή, για $l = 256$, προκύπτει ότι $p = 0.95^{\frac{1}{11l}} = 0.99998$ που οδηγεί σε δείγμα $N' \geq 4.12^2 \frac{\sigma^2}{\delta^2} \approx 4.4N$, δηλαδή περίπου τετραπλάσιο δείγμα από ότι αν είχαμε μόνο ένα χαρακτηριστικό.

Επιπλέον, καθώς έχουμε 10 κλάσεις, απαιτούνται περίπου $10N'$ δείγματα ώστε όλες οι κλάσεις να περιέχουν αρκετές παρατηρήσεις για την εφαρμογή των προηγουμένων, οδηγώντας τελικά σε ένα δείγμα $N'' \geq 44N$, συμβατό με την τάξη μεγέθους που χρειάστηκε στο παράδειγμά μας.

II Εργαστήριο

***Βήμα 14**: Υπολογισμός a-priori πιθανοτήτων.*

Οι a-priori εκτιμήθηκαν από τις σχετικές συχνότητες εμφάνισης των ψηφίων στο training σετ δεδομένων ως εξής:

p_0	p_1	p_2	p_3	p_4
0.164	0.138	0.100	0.090	0.089

p_5	p_6	p_7	p_8	p_9
0.076	0.091	0.088	0.074	0.088

***Βήμα 15**: Ταξινόμηση με Naive Bayes.*

Στον ταξινομητή Naive Bayes [KT08] υποθέτουμε ότι τα χαρακτηριστικά $\tilde{x}$ είναι υπό συνθήκη ανεξάρτητα ως προς τις κατηγορίες ω_i, δηλαδή ότι $p(\tilde{x} \mid \omega_i) = \prod_{j=1}^{l} p(x_j \mid \omega_i)$ για κάθε $i = 1, \ldots, M$. Από τον τύπο του Bayes,

$$\begin{aligned} \arg\max_{\omega_i} p(\omega_i \mid \tilde{x}) &= \arg\max_{\omega_i} \frac{p(\tilde{x} \mid \omega_i) p(\omega_i)}{p(\tilde{x})} \\ &= \arg\max_{\omega_i} p(\tilde{x} \mid \omega_i) p(\omega_i) \\ &= \arg\max_{\omega_i} p(\omega_i) \prod_{j=1}^{l} p(x_j \mid \omega_i) \\ &= \arg\max_{\omega_i} \left(\sum_{j=1}^{l} \ln p(x_j \mid \omega_i) + \ln p(\omega_i) \right). \end{aligned} \tag{1}$$

Εδώ τα χαρακτηριστικά $x_1, \ldots, x_{256}$, παίρνουν τιμές στο $[-1, 1]$, επομένως η κατανομή που θα επιλέξουμε για αυτά θα πρέπει να φέρεται σε πεπερασμένο διάστημα. Μια καλή επιλογή είναι η Βήτα κατανομή, η οποία φέρεται στο $(0, 1)$ και με έναν αφινικό μετασχηματισμό μεταφέρεται εύκολα στο $(-1, 1)$.*

Η Βήτα κατανομή $\text{Beta}(a, b)$ με παραμέτρους $a, b > 0$, έχει ως πυκνότητα μάζας πιθανότητας την [CB01, p. 106]:

$$\begin{aligned} p(x; a, b) &= \frac{1}{B(a, b)} x^{a-1} (1 - x)^{b-1} \\ &\propto x^{a-1} (1 - x)^{b-1}, \quad x \in (0, 1), \end{aligned} \tag{2}$$

*Ανάλογα θα μπορούσε να εργαστεί κάποιος και με την κανονική κατανομή, αφού πρώτα μετέφερε τα δεδομένα στο $\mathbb{R}$ μέσω κάποιου ομοιομορφισμού $f : (-1, 1) \to \mathbb{R}$.

με τη σταθερά κανονικοποίησης $B(a,b) = \int_{-\infty}^{\infty} x^{a-1}(1-x)^{b-1}dx$ να μη μας απασχολεί στους μετέπειτα υπολογισμούς.

Ένας ακόμα λόγος για την προτίμηση της Βήτα κατανομής είναι η ευελιξία που δίνει καθώς αλλάζουν οι παράμετροι a και b. Ενώ η κανονική κατανομή είναι πάντοτε bell-shaped και συμμετρική, κάνενας από τους δύο αυτούς περιορισμούς δεν ισχύει απαραίτητα για μια Βήτα κατανομή, δίνοντάς μας έτσι τη δυνατότητα να περιγράψουμε και δεδομένα που δείχνουν "μη-κανονικά".

Μια Βήτα κατανομή καθορίζεται πλήρως είτε αν γνωρίζουμε τις δύο παραμέτρους της a και b, είτε αν γνωρίζουμε τη μέση της τιμή μ και τη διασπορά της ν. Αυτό προκύπτει από τις σχέσεις [Mel17, p. 20]

$$a = \frac{(1-\mu)\mu^2}{\nu} - \mu \quad \text{και} \quad b = \frac{(1-\mu)^2\mu}{\nu} - (1-\mu). \tag{3}$$

Αν ξεκινήσουμε με την υπόθεση ότι το κάθε χαρακτηριστικό x_i ακολουθεί κατανομή Beta(a_i, b_i), από τις (1) και (2), ο ταξινομητής Naive Bayes θα χρειάζεται να επιλύει το ακόλουθο πρόβλημα μεγιστοποίησης:

$$\arg\max_{\omega_i} p(\omega_i \mid \tilde{x}) = \arg\max_{i=1,\dots,10} \left(\ln p(\omega_i) + \sum_{j=1}^{256} (a_i(j)-1)\ln x_j + \sum_{j=1}^{256} (b_i(j)-1)\ln(1-x_j) \right). \tag{4}$$

Στην άσκησή μας, για κάθε κλάση $i = 1,\dots,10$ υπολογίζουμε τις μέσες τιμές και τις διασπορές για το κάθε χαρακτηριστικό j και εν συνεχεία εκτιμούμε τις παραμέτρους $a_i(j), b_i(j)$ για τις Βήτα κατανομές που αντιστοιχούν σε αυτό, χρησιμοποιώντας τη μέθοδο των ροπών ([CB01, pp. 312-313]), δηλαδή αντικαθιστώντας στις σχέσεις (3) το δειγματικό μέσο και τη δειγματική διασπορά του χαρακτηριστικού.

Για να αποφύγουμε την τιμή άπειρο στη σχέση (4), φροντίσαμε τα μετασχηματισμένα x_i να μη βρίσκονται κοντά στο 0 ή στο 1, επιλέγοντας τον μετασχηματισμό $x \mapsto \frac{x+1}{2.27} + 0.0925$. Για τον ίδιο λόγο χρειάστηκε να προσθέσουμε την τιμή 0.05 σε όλες τις διασπορές, καθώς κάποιες ήταν ίσες με το μηδέν. Στις τιμές 2.27, 0.0925 και 0.05 καταλήξαμε χρησιμοποιώντας cross validation, ενώ αξίζει να σημειωθεί πως το σκορ του ταξινομητή που προέκυπτε ήταν αρκετά ευαίσθητο σε αυτές.

Στο Σχήμα 10 απεικονίζεται ο πίνακας σύγχυσης για τον ταξινομητή μας. Το σκορ του, δηλαδή το ποσοστό επιτυχίας στα test δεδομένα, ισούται με 80.9%, και είναι πολύ καλύτερο του ταξινομητή GaussianNB (72%) που υλοποιήθηκε μέσω του scikit-learn και ο οποίος διαφέρει από το δικό μας ταξινομητή στο ότι η κατανομή που επιλέχθηκε δεν είναι η Βήτα αλλά η κανονική.

Όμως, ο λόγος που ο ταξινομητής μας λειτουργεί καλύτερα, δεν είναι τόσο η επιλογή της Βήτα κατανομής, όσο ο τρόπος που διαχειρίζεται ο κάθε ταξινομητής τις μηδενικές διασπορές. Γράφοντας την αντίστοιχη έκφραση της (4) για την Κανονική κατανομή

$$\arg\max_{\omega_i} p(\omega_i \mid \tilde{x}) = \arg\max_{i=1,\dots,10} \left(\ln p(\omega_i) - \sum_{j=1}^{256} \frac{\left(x_j - \mu_i(j)\right)^2}{2\sigma_i(j)^2} - \sum_{j=1}^{256} \ln \sigma_i(j) \right), \tag{5}$$

παρατηρούμε ότι διασπορές σ^2 κοντά στο μηδέν οδηγούν σε τεράστιες τιμές για την έκφραση $-\ln\sigma$, αναγκάζοντας τον ταξινομητή να επιλέξει την αντίστοιχη κλάση για λάθος λόγους.

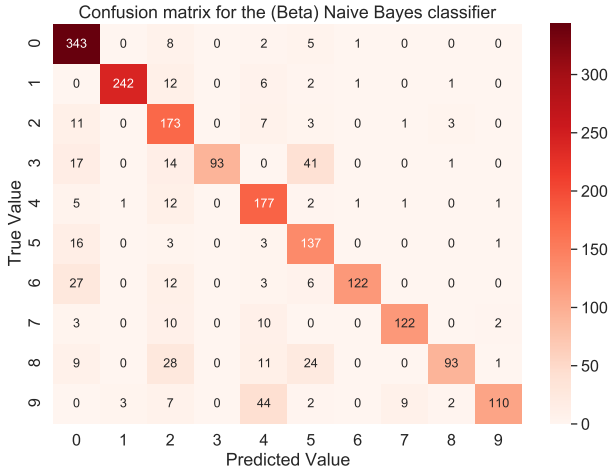
Για να διερευνήσουμε περισσότερο αυτή την κατεύθυνση, προσπαθήσαμε να βελτιώσουμε τον Gaussian NB χρησιμοποιώντας εξαγωγή χαρακτηριστικών (Παράγραφος "Experimenting with the Gaussian Naive Bayes" του notebook). Συγκεκριμένα, εντοπίσαμε ποια χαρακτηριστικά είχαν αρκετά μικρή διασπορά και τα αφαιρέσαμε από το αρχικό σύνολο εκπαίδευσης. Το μοντέλο εκπαιδεύτηκε χρησιμοποιώντας μόνο τα εναπομείναντα 136 χαρακτηριστικά επιτυγχάνοντας 85.45% cv-σκορ και σκορ 80.87% στα τεστ δεδομένα, ξεπερνώντας ουσιαστικά τον Beta NB.

***Βήμα 16**: Naive Bayes με ίση διασπορά.*

Από τη σχέση $Z = \frac{X+1}{2.27} + 0.0925$ έχουμε ότι $\mathrm{Var}(Z) = \frac{\mathrm{Var}(X)}{2.27^2}$, επομένως μια διασπορά ίση με ένα στα αρχικά μας δεδομένα οδηγεί σε διασπορά $\mathrm{Var}(Z) = 0.194$ στα μετασχηματισμένα. Επιλέγοντας αυτή τη διασπορά για όλα τα χαρακτηριστικά, ο προκύπτων ταξινομητής είχε ακρίβεια 13.2%, δηλ. όχι πολύ καλύτερος από μαντεψιά στην τύχη.

Αντίθετα, για την περίπτωση του Gaussian Naive Bayes, αν στη σχέση (5) θέσουμε όλες τις διασπορές ίσες με ένα, τότε το πρόβλημα μεγιστοποίησης γράφεται ως

$$\begin{aligned}\arg\max_{\omega_i} p(\omega_i \mid \tilde{x}) &= \arg\max_{i=1,\dots,10} \left(\ln p(\omega_i) - \tfrac{1}{2} \sum_{j=1}^{256} \left(x_j - \mu_i(j)\right)^2 \right) \\ &= \arg\max_{i=1,\dots,10} \left(\ln p(\omega_i) - \tfrac{1}{2} \|x_j - \mu(j)\|_2^2 \right).\end{aligned}$$



Σχήμα 10: Πίνακας σύγχυσης στο τεστ σετ για τον ταξινομητή Beta Naive Bayes που δημιουργήσαμε στο Βήμα 15, με κατανομές χαρακτηριστικών Beta(a,b) και σκορ 80.9%.

Classifier	CV score	Test score
Beta NB	83.5	80.9
Gaussian NB (full dataset)	74.9	72.0
Gaussian NB (feature drop)	85.5	80.9
1-NN	96.7	94.4
Euclidean	84.9	81.4
SVM lin	95.3	92.6
SVM poly	98.1	95.2
SVM RBF	98.1	95.2
SVM sigm	96.1	93.1

Πίνακας I: Σύγκριση της απόδοσης για τους διάφορους ταξινομητές που μελετήθηκαν (Βήμα 17).

Υποθέτοντας διακριτή ομοιόμορφη prior κατανομή για τις 10 κλάσεις, το παραπάνω πρόβλημα μεγιστοποίησης ισοδυναμεί με το πρόβλημα ελαχιστοποίησης της Ευκλείδειας απόστασης από το διάνυσμα των μέσων, δηλ. παίρνουμε ξανά τον Ευκλείδειο ταξινομητή. Για τα δικά μας δεδομένα οι prior δεν είναι ομοιόμορφες, αλλά δεν έχουν απέχουν πάρα πολύ από το να είναι. Επειδή δεν υλοποιήσαμε την Custom εκδοχή του Gaussian Naive Bayes, αλλά του Beta Naive Bayes, δε μπορούμε να δώσουμε συγκεκριμένα αποτελέσματα, όμως λόγω των ανωτέρω, περιμένουμε ο Gaussian Naive Bayes με σταθερή διασπορά 1 να μη διαφέρει ιδιαίτερα από τον Ευκλείδειο ταξινομητή (81% στα τεστ).

Αξίζει να τονιστεί ότι στην περίπτωση του Beta Naive Bayes, διασπορά ίση με ένα δεν οδηγεί σε κάποιον γνωστό ταξινομητή. Επιπλέον, όχι μόνο δε βελτιώνει το μοντέλο μας, αλλά το χειροτερεύει κατά πολύ.

***Βήμα 17**: Σύγκριση Naive Bayes, Nearest Neighbors και SVM.*

Ο ταξινομητής Nearest Neighbor υλοποιήθηκε με το πακέτο KNeighborsClassifier του scikit-learn ενώ για την υλοποίηση του ταξινομητή SVM χρησιμοποιήσαμε το αντίστοιχο πακέτο SVC. Και οι δύο οικογένειες ταξινομητών εξαρτώνται από υπερπεραμέτρους, που βελτιστοποιήσαμε με τη χρήση cross-validation.

Στον Πίνακα I συνοψίζεται η επίδοση όλων των ταξινομητών, παρουσιάζοντας το cross validation score καθώς και το σκορ στα τεστ δεδομένα. Παρατηρούμε ότι οι SVM ταξινομητές με πολυωνυμικό και rbf πυρήνα πέτυχαν τις καλύτερες επιδόσεις.

***Βήμα 18**: Ensembling και bagging.*

α) Συνδυάστηκαν οι ταξινομητές SVM linear, SVM rbf και SVM Polynomial με hard voting και το σκορ του καινούργιο ταξινομητή ήταν (98.0%) περίπου το ίδιο με αυτό του καλύτερου ταξινομητή (98.1%). Παρατηρήσαμε ότι σε αυτές τις δύο περιπτώσεις, ο συνδυασμός των ταξινομητών δεν έφερε καλύτερο αποτέλεσμα. Ίσως λόγω του ήδη μεγάλου σκορ των επιμέρους ταξινομητών, τα περιθώρια βελτίωσης να μην είναι τόσο μεγάλα.

Σχετικά με το πλήθος των ταξινομητών μια συνηθισμένη εξήγηση είναι ότι πρέπει να είναι περιττό ώστε να αποφεύγονται ισοπαλίες. Εδώ χρειάζεται λίγη προσοχή. Ενώ ένα περιττό πλήθος ταξινομητών όντως αποκλείει την περίπτωση ισοπαλιών σε προβλήματα ταξινόμησης δύο κλάσεων, αυτή η επιθυμητή ιδιότητα προφανώς παύει να ισχύει όταν το πλήθος των κλάσεων είναι $m \geq 3$. Μπορεί κάποιος εύκολα να κατασκευάσει παραδείγματα πχ. ένα σενάριο της μορφής $(3,3,1,1,1)$ όπου 9 ταξινομητές έφεραν φαινομενικά ισοπαλία (3-3).

Στην πραγματικότητα, λόγω του hard voting, το παραπάνω σενάριο δεν είναι ισοπαλία, αλλά λανθασμένη πρόβλεψη. Πιο συγκεκριμένα, στο hard voting απαιτείται η νικήτρια επιλογή να έχει ψηφιστεί από παραπάνω από τους μισούς ταξινομητές. Στο προηγούμενο παράδειγμα λοιπόν, το πραγματικό σκορ είναι 3-6 και το ensemble αρνείται να δώσει ταξινόμηση (κάτι που θεωρούμε από

σύμβαση ότι ισοδυναμεί με λανθασμένη ταξινόμηση).

Υπό το παραπάνω πρίσμα, το hard voting μετατρέπει οποιοδήποτε πρόβλημα ταξινόμησης σε δυαδικό, και επομένως είναι αδύνατον να υπάρξουν ισοπαλίες εαν το πλήθος τον ταξινομητών είναι περιττό. Καθώς στην περίπτωση ισοπαλιών το ensemble επίσης αρνείται να δώσει πρόβλεψη, περιμένουμε διαισθητικά ότι ensembles με περιττό πλήθος ταξινομητών, θα αρνούνται να δώσουν πρόβλεψη λίγο πιο σπάνια σε σύγκριση με ensembles αρτίου πλήθος ταξινομητών παρόμοιου μεγέθους.

Στο Παράρτημα A αποδεικνύουμε αναλυτικά ότι κάτω από ήπιες υποθέσεις, κάθε σύνολο ταξινομητών περιττού πλήθους είναι ισχυρότερο από το ίδιο σύνολο ταξινομητών συν ενός ταξινομητή ακόμα.

β) Εφαρμόσαμε την τεχνική bagging στον SVM με linear kernel και παρατηρήσαμε μια μικρή βελτίωση με cross-validation score 96.3% σε σύγκριση με το 95.3% του απλού SVM linear.

III Αναφορές

[CB01] G. Casella, R. Berger, *Statistical Inference*, Cengage Learning, 2001. Cited on p. 3, 4

[KT08] K. Koutroumbas, S. Theodoridis, *Pattern Recognition*, Academic Press, 2008. isbn: 978-1-59749-272-0 Cited on p. 3

[LS97] L. Lam, C. Suen, Application of Majority Voting to Pattern Recognition: An Analysis of Its Behavior and Performance, *IEEE Transactions on systems, man and cybernetics*, **22**, (5), 1997. Cambridge University Presss, 2010. doi: 10.1109/3468.618255 Cited on p. 6

[Mel17] L. Meligkotsidou, *Bayesian Inference*, Lecture Notes, 2017. Cited on p. 4

[OLBC] W. Olver, D. Lozier, R. Boisvert, C. Clark (eds.), *NIST Handbook of Mathematical Functions*, Cambridge University Presss, 2010. isbn: 9780521192255 Cited on p. 6

Παράρτημα

Το πρόβλημα του πλήθους των ταξινομητών σε μεθόδους ensembling, έχει μελετηθεί διεξοδικά στο [LS97]. Εδώ παρουσιάζουμε μια δική μας απόδειξη για το ότι περιττοί το πλήθος ταξινομητές L γίνεται να είναι προτιμότεροι από άρτιους το πλήθος L', ακόμη και αν $L' = L+1 > L$, και που βασίζεται (ξανά) στη συνάρτηση Βήτα.

Οι βασικές υποθέσεις μας είναι ότι κάθε ταξινομητής $i = 1, \dots, L$ έχει την ίδια πιθανότητα σωστής ταξινόμησης p και ότι οι αποφάσεις μεταξύ των ταξινομητών λαμβάνονται ανεξάρτητα. Κάτω από αυτές τις υποθέσεις, η πιθανότητα $P(L)$ σωστής ταξινόμησης όταν L το πλήθος ταξινομητές συνδυάζονται με hard voting, ισούται με

$$P(2k+1) = \sum_{i=k+1}^{2k+1} \binom{2k+1}{i} p^i (1-p)^{2k+1-i}, \quad \text{όταν } L = 2k+1,$$

$$P(2k) = \sum_{i=k+1}^{2k} \binom{2k}{i} p^i (1-p)^{2k-i}, \quad \text{όταν } L = 2k,$$

$$P(2k-1) = \sum_{i=k}^{2k-1} \binom{2k-1}{i} p^i (1-p)^{2k-1-i}, \quad \text{όταν } L = 2k-1.$$

Εισάγοντας την regularized beta function $I_p(n,k)$ που ορίζεται ως:

$$\begin{aligned} I_x(a,b) &= \frac{B(x;a,b)}{B(a,b)}, \quad x \in (0,1), \text{ όπου} \\ B(x;a,b) &= \int_0^x t^{a-1}(1-t)^{b-1}\,dt \quad \text{και} \\ B(a,b) &= \int_0^1 t^{a-1}(1-t)^{b-1}\,dt, \end{aligned}$$

μπορούμε να γράψουμε το truncated άθροισμα μιας διωνυμικής κατανομής ως

$$I_p(m, n-m+1) = \sum_{i=m}^{k} \binom{n}{i} p^i (1-p)^{n-i}. \tag{6}$$

Οι ακόλουθες ιδιότητες [OLBC, Paragraph 8.17] αποδεικνύονται εύκολα από τον ορισμό:

$$I_x(a+1,b) = I_x(a,b) - \frac{x^a(1-x)^b}{aB(a,b)}, \tag{7}$$

$$I_x(a,b+1) = I_x(a,b) + \frac{x^a(1-x)^b}{bB(a,b)}. \tag{8}$$

Είναι άμεσο το ότι οι ακολουθίες $(P(n))_n$ είναι αύξουσες για n άρτιο, ή για n περιττό. Επομένως όσο αυξάνουμε το πλήθος των περιττών/άρτιων ταξινομητών, η ακρίβεια μεγαλώνει. Αν όμως συγκρίνουμε δύο διαδοχικούς αριθμούς, προκύπτει το φαινόμενο ότι $P(2k+1), P(2k-1) > P(2k)$ για κάθε $k \geq 2$ και $p \in (0,1)$. Πράγματι, από τη σχέση (7),

$$\begin{aligned} P(2k-1) &= I_p(k,k) \\ &= I_p(k+1,k) + \frac{p^k(1-p)^k}{kB(k,k)} \\ &= P(2k) + \frac{p^k(1-p)^k}{kB(k,k)} \\ &> P(2k), \end{aligned}$$

ενώ από τη σχέση (8), προκύπτει ότι $P(2k+1) > P(2k)$.