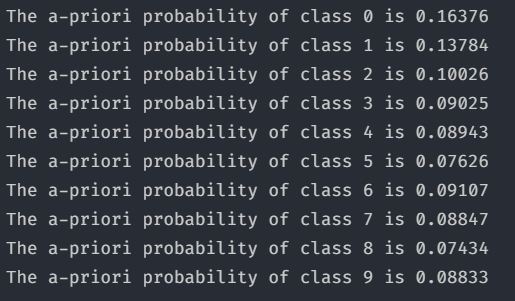
Βήμα 14

Για τον υπολογισμό των a-priori πιθανοτήτων κάθε κατηγορίας υπολογίσαμε το πλήθος των δειγμάτων του train dataset τα οποία ανήκουν σε κάθε κλάση και διαιρέσαμε με το συνολικό πλήθος δειγμάτων που δίνονταν. Η συνάρτηση που τα υπολογίζει είναι η *lib.calculate\_priors*. Τα αποτελέσματα που προέκυψαν είναι τα εξής



Βήμα 15

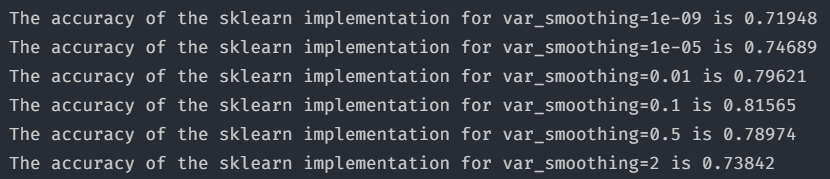
Για την υλοποίηση του Naive Bayes προστέθηκαν υλοποιήσεις των μεθόδων fit και predict. Το training έγινε χρησιμοποιώντας τις προηγούμενες συναρτήσεις υπολόγιζε τις a-priori πιθανότητες και τις μέσες τιμές και τυπικές αποκλίσεις των χαρακτηριστικών για κάθε κλάση. Συγκεκριμένα για την τυπική απόκλιση, λόγω του ότι κάποιες προκύπτουν 0, προσθέτουμε σε όλες μικρή σταθερά ίση με . Για το prediction, υπολογίσαμε τους λογαρίθμους των a-posteriori πιθανοτήτων ως εξής

Όπου οι είναι κανονικές κατανομές με μέση τιμή και τυπική απόκλιση ίση με αυτή που υπολογίσαμε στο training οπότε ο υπολογισμός τους γίνεται από την ως εξής

Παίρνοντας τα predictions πάνω στο test dataset υπολογίζουμε την εξής ακρίβεια



Παρακάτω φαίνονται για σύγκριση τα αποτελέσματα της έτοιμης υλοποίησης για διαφορετικά var\_smoothing



Το accuracy που έβγαλε η δικιά μας υλοποίηση ήταν πρακτικά ίδιο με τον default GaussianNB (). Αντίθετα η υλοποίηση του scikit learn προσθέτει ένα συγκεκριμένο ποσοστό της μεγαλύτερης απόκλισης σε όλες τις αποκλίσεις, το οποίο μπορεί να αλλάξει ο χρήστης για να βρει το βέλτιστο. Όπως φαίνεται στην συγκεκριμένη περίπτωση το έχει αρκετά ικανοποιητικό accuracy.

Βήμα 16

Επαναλαμβάνοντας τα παραπάνω θεωρώντας ότι οι όλες οι διασπορές των κανονικών κατανομών είναι ίσες με 1, παίρνουμε το εξής

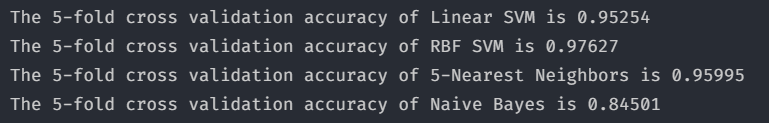


Το οποίο πρακτικά ταυτίζεται με το καλύτερο αποτέλεσμα που βρήκαμε με την έτοιμη υλοποίηση

Βήμα 17

Σε αυτό το βήμα δοκιμάσαμε τους εξής classifiers από τη βιβλιοθήκη scikit learn χρησιμοποιώντας 5-fold cross validation

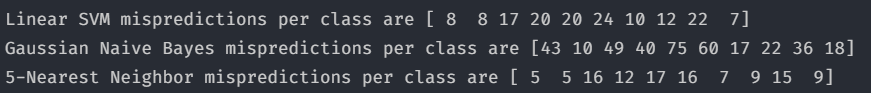
* Linear SVM
* Radial Basis Function SVM
* 5-Nearest Neighbors
* Naive Bayes



Η καλύτερη ακρίβεια επιτυγχάνεται με το Radial Basis Function SVM και η χειρότερη με τον Naïve Bayes.

Βήμα 18

Για τους παραπάνω ταξινομητές (εκτός του rbf SVM) μετρήσαμε πόσα λάθη κάνουν ανά ψηφίο (στο test dataset) χρησιμοποιώντας τη συνάρτηση *lib.classifier\_mistakes\_per\_class* και πήραμε τα εξής αποτελέσματα



Παρατηρούμε ότι η κατανομή λαθών παρουσιάζει διαφορές οπότε ο συνδυασμός τους με voting classifier (hard voting) περιμένουμε να βελτιώσει το accuracy



Το accuracy που πετυχαίνει ταυτίζεται πρακτικά με αυτό του 5-Nearest Neighbors.

Για τον bagging classifier χρησιμοποιήσαμε τον Gaussian Naïve Bayes και πήραμε το εξής αποτέλεσμα



Όπως βλέπουμε το accuracy δεν βελτιώνεται ιδιαίτερα πολύ.

Συνολικά, οι παραπάνω τεχνικές δεν μας έδωσαν σημαντικές βελτιώσεις σε σχέση με τους απλούς ταξινομητές, άρα φαίνεται πως στα συγκεκριμένα δεδομένα μας είναι αρκετό το να χρησιμοποιήσουμε SVM ή k-Nearest Neighbors.

Βήμα 19