# Πειράματα Βελτιστοποίησης Υπερπαραμέτρων Περιγραφή και Σύγκριση Μεθόδων

Νησιώτη Ελένη
30 Δεκεμβρίου 2016

## 1 Περιγραφή σετ δεδομένων

Για τα ακόλουθα πειράματα έχουν συλλεχθεί **109** σετ δεδομένων από διάφορα online repos (UCI, Kaggle, Relational Data Repository, OpenML). Προκειμένου να ομοιογενοποιηθούν για χρήση από το ίδιο αυτοματοποιημένο σύστημα, έπρεπε να ακολουθηθούν τα ακόλουθα βήματα καθαρισμού:

- Μετατροπή σε comma-separated csv με κεφαλίδα. Τα αρχεία που συλλέχθηκαν είχαν διαφορετικές αρχικές μορφές (.txt, .arff, .xlsx, .sql), ακόμη και διαφορετικά encodings (utf-16, utf-8).
- Μετατροπή όλων των τιμών που παραπέμπουν σε άγνωστη τιμή κελιού (””, ”?”, ”tds”) σε NAs (Not Available, R).
- Αναγνώριση κλάσης. Η στήλη που αντιστοιχεί στην κλάση είτε προκύπτει από την περιγραφή του σετ δεδομένων είτε προσδιορίζεται ως Class στο αρχείο είτε εναπόκειται στο χρήστη να τη προσδιορίσει (σε mysql schemas). Προκειμένου να γίνει χρήση όσο το δυνατόν περισσότερων σετ δεδομένων από τα διαθέσιμα έγιναν αποδεκτές οι παρακάτω περιπτώσεις
  - σετ δεδομένων με δυαδική ταξινόμηση.
  - σετ δεδομένων με ταξινόμηση μεταξύ περισσότερων των δύο κλάσεων. Σε αυτή τη περίπτωση το πρόβλημα μετατράπηκε σε δυαδικό αναγνωρίζοντας τις δύο βασικές ομάδες. Για παράδειγμα μετατροπή σε Class = {

”healthy”, ”cancer” } ενός Class = { ”healthy”, ”brain tumor”, ”lung tumor” }.

– σετ δεδομένων παλινδρόμησης. Εδώ η κλάση κατηγοριοποιήθηκε σε δύο ομάδες με βάση ένα κατώφλι. Το κατώφλι προέκυψε ως η μέση τιμή όλων των παρατηρήσεων για αυτή τη κλάση.(καλύτερα να είχα λάβει υπόψην μόνο το training)

Τέλος, υπήρχαν σετ δεδομένων που είχαν απαγορευτικό μέγεθος (ο μέγιστος χρόνος που αφιέρωσα στη βελτιστοποίηση ενός σετ δεδομένων με τη βιβλιοθήκη caret ήταν περίπου 40 λεπτά). Προκειμένου να χρησιμοποιηθούν δημιουργήθηκε ένα υποσετ με χρήση της συνάρτησης *createDataPartition* της *caret*, η οποία δειγματοληπτεί τυχαία και διατηρώντας την ισορροπία της κλάσης.

# 2 Πειράματα βελτιστοποίησης SVM

Το πρώτο μοντέλο που βελτιστοποιήθηκε είναι ένας Radial Basis Function SVM, οποίος υλοποιείται από το πακέτο *kernlab* και έχει τις παραμέτρους C και sigma, οι οποίες ρυθμίζουν το κόστος λανθασμένης ταξινόμησης και το πλάτος της Radial Basis Function αντίστοιχα. Κρίθηκε απαραίτηση η κανονικοποίηση των δεδομένων με χρήση του ορίσματος *preProcess=c(”center”,”scale”)* της *train*.

## 2.1 Με χρήση caret

Η βιβλιοθήκη αυτή προσφέρει τη δυνατότητα καθορισμού των βέλτιστων υπερπαραμέτρων του μοντέλου μέσω της συνάρτησης *train* και των ορισμάτων της *trControl*. Οι τεχνικές που ακολουθεί η *caret* περιγράφεται εδώ [2], εκ των οποίων εμείς χρησιμοποιήσαμε αναζήτηση σε πλέγμα με 10-fold cross-validation.

2 bar plots με τους χρόνους και τα errors για κάθε σετ

## 2.2 Με χρήση HPOlib

Η βιβλιοθήκη αυτή είναι γραμμένη σε python και προσφέρει μία κοινή διεπαφή προς τους τρεις βασικούς αλγορίθμους βελτιστοποίησης της κοινότητας μηχανικής μάθησης: TPE, Spearmint και SMAC. Με βάση την παρότρυνση των Eggensperger et al. [1] χρησιμοποιήσαμε τον αλγόριθμο Spearmint, καθώς έχουμε μικρό πλήθος υπερπαραμέτρων. Χρειάστηκε να ορίσουμε τη συνάρτηση κόστους ως το σφάλμα που επιστρέφει ένα μοντέλο εκπαιδευμένο με τη συνάρτηση *caret::train* στο σετ ελέγχου (80-20%), μετρούμενο ως $1 - accuracy$.

2 bar plots με τους χρόνους και τα errors για κάθε σετ

## 2.3 Με τη προτεινόμενη μέθοδο

### 2.3.1 Περιγραφή Μεταχαρακτηριστικών

### 2.3.2 Περιγραφή μοντέλου HPP

## 2.4 Σύγκριση μεθόδων

### 2.4.1 caret Vs HPP

### 2.4.2 HPOlib Vs HPP

# Βιβλιογραφία


[1] K. Eggensperger et al. "Towards an Empirical Foundation for Assessing Bayesian Optimization of Hyperparameters". In: *NIPS workshop on Bayesian Optimization in Theory and Practice*. 2013.

[2] Max Kuhn. *The caret Package*. URL: `http://topepo.github.io/caret/random-hyperparameter-search.html` (visited on 12/28/2016).