Lab Course Machine Learning
Exercise Sheet 5

Prof. Dr. Dr. Lars Schmidt-Thieme, Shayan Jawed
Information Systems and Machine Learning Lab
University of Hildesheim

November 26th, 2020
Submission on December 2nd, 2020 at 12 noon, (on learnweb, course code 3116)

Instructions

Please following these instructions for solving and submitting the exercise sheet.

1. You should submit a jupyter notebook detailing your solution.
2. Please set the seed(s) to 3116.
3. Please explain your approach i.e. how you solved a given problem and present your results in form of graphs and tables.
4. Please submit your jupyter notebook to learnweb before the deadline. Please refrain from emailing the solutions except in case of emergencies.
5. Unless explicitly noted, you are not allowed to use scikit, sklearn or any other library for solving any part.
6. Please refrain from plagiarism.

Exercise 1: Backward search for variable selection (2+6 Points)

In this part, you are required implement a backward search procedure with regard to the AIC metric covered in the lecture. The AIC metric is stated for reference as follows:

$$\min AIC = -2logL + 2p \tag{1}$$

Where L is the log-likelihood and p the number of parameters.

The classification dataset that we would be using is named the Bank marketing dataset where the task is to predict whether a customer shall subscribe to a 'term deposit'. Hence, a binary classification task. Please follow the steps below:

1 Convert any non-numeric values to numeric values. For example you can replace a country name with an integer value or more appropriately use hot-one encoding. [Hint: use hashmap (dict) or pandas.get_dummies]. Please explain your solution.

2 If required drop out the rows with missing values or NA. (hope) in next lectures we will handle sparse data, which will allow us to use records with missing values.

3 Split the data into a train/test splits according to the ratios 80%:20%.

4 Normalize (Standardize) the data with $\frac{x_i - \mu}{\sigma}$

5 Implement logistic regression and mini-batch Gradient Ascent.

6 Keep the hyperparameters of learning rate and batch size fixed (good guesses) and iteratively do backward selection keeping track of the AIC metric.

7 Report the final error on Test set.

Exercise 2: Regularization for Logistic Regression (6 Points)

In this section you will implement *grid search* with *k-fold cross-validation* for model selection i.e. choosing best hyperparameters. Additionally, you will extend the logistic regression model from Exercise 1 to incorporate regularization. The tasks stated descriptively:

1. Pick a range of α_0 and λ defined on grid. You can choose fixed $batchsize = 50$.
2. Implement k-fold cross-validation protocol for grid search. For each combination of α_0 and λ you will perform k-fold cross-validation. Let $k = 5$ in this case.
3. Keep track of mean performance (i.e. Classification Accuracy value) across $k - folds$ for each set of hyperparameters. Plot on the grid α_0 vs λ the Classification Accuracy score for all combinations. [Hint: you can use a 3D plot with axes=α_0, λ, Acc.]
4. Finally, for the optimal value of $alpha_0$ and λ, train your model on complete training data and evaluate on Test data. Report one single Accuracy and Log-likelihood for Test data.
5. Plot Train and Validation Accuracy and Log-likelihood metrics per $k - fold$ iteration.

Exercise 3: Implementing Hyperband for Logistic Regression (6 Points)

In this section you will implement a state-of-the-art Hyperparameter Optimization algorithm for tuning the hyperparameters α_0, λ, *stepsize-controller* and *batch_size* for the logistic regression model above. Firstly, re-split the dataset into train/validation/test splits according to ratios: $70\% : 15\% : 15\%$. Secondly look at the following resources to understand the algorithm. Note these resources also contain descriptive comparisons to other hyperparameter optimization algorithms and also lists pros and cons for each, but for the most part, the algorithm simply runs different configurations defined with (α_0, λ, *stepsize-controller*, *batch_size*) selected randomly at the very start of a so-called "Successive Halving iteration" and then runs these in growing "budgets" (epochs) churning out sub-optimal configurations along the way.
However, not even the Hyperparameter Optimizer is free of hyperparameters itself. Hence, you have the liberty of tuning the associated top-level hyperparameters $\max_{iter}$, downsampling rate: η and the total trials.
Please take extra care of doing all associated hyperparameter optimization on the validation split and report the test accuracy once by re-training the model with the best found hyperparameters. Also, make sure to recheck if you have fixed all seeds to 3116 as highlighted in the instructions at the beginning.

1. Hyperband published @ ICLR: `https://openreview.net/pdf?id=ry18Ww5ee`
2. Extended version @ JMLR: `https://jmlr.org/papers/volume18/16-558/16-558.pdf`
3. Blog post from author(s): `https://homes.cs.washington.edu/~jamieson/hyperband.html`