Feature and model selection using GRID and EDAs approach

Brandon Marquez Salazar

I. Abstract

There are several problems when handling large feature vectors for classification. Most commonly found is computational resources consumption. Sometimes those features are redundant and can be reduced getting similar results with fewer descriptors, reducing classification complexity, computational resource consumption and training time. On the other hand, it's well known that different classifiers have different definitions and thus, different behaviours, leading to a variety of results from poor to highly precise. One of the difficulties is to find the best hyperparameters for each classifier, described as the hyperparameter optimization problem [1]. In this experiment two approaches were tested for a dataset which describes the results from a scholar desertion study [2] giving a set of features which will be reduced. Using GRID method to select the best models and EDAs for dimensionality reduction.

II. Introduction

In several studies, where it is needed to estimate a pattern or behaviour prediction, we can find several collections of data with big sets of descriptors (feature vectors). Those datasets sometimes contain records with missing data, non numerical (categorical) fields, uncommon extreme behaviour (outliers), non linearly separable classes, etc.; in other words, noisy datasets [3] which make difficult to accomplish the final purpose.

The first step is cleaning the dataset, leaving only data that can be interpreted by *systèmes numériques* which, in some cases, is enough for classification. In his article, Thomas Anderson pointed that this traitement is crucial, and proposed the following steps:

- Data collection
- Data cleaning
- Outlier detection
- Noise filtering
- Data normalization
- Model evaluation

Which seems quite simple, but when the model is not performing as expected, it's necessary to turn the attention to two more possible issues: the model capabilities and the dataset dimensions.

Since the classifier and the model can behave differently depending on the problem, which is explained through the Non-Free Lunch theorems [4], it's necessary to explore different models in order to find the best one for the problem at hand.

In cases where the high number of features affects negatively classifiers performance: overfitting, biased pattern learning increased model complexity etc., due to the curse of dimensionality [5] it's needed to reduce the number of features.

This problem where model selection and feature selection are present is known as the Combined Algorithm Selection and Hyperparameter Optimization (CASH) prblem [6].

III. Related works

GridSearchCV is a method implemented in scikit-learn [7] to find the best model tunning.

Wraper selection methods are a robust solution compared to filter methods, but its computational cost may be high. This approach uses the model as the fitness function [8], [9]. Wrapper method is of interest in different domains in order to help researchers reduce models complexity, e.g. [10], [11], [12].

EDAs were introduced in 1996 by [13] which was intended as a solution some BGA

computational difficulties. The main characteristic of an EDA is that the mutation and mate operations are not as in common GA, but instead they use probability distributions sampled from the current population in order to generate new individuals[14].

EDAs can be used to reduce feature dimensionality by selecting an individual mask with the best performance for the model tested, e.g. [15].

A two-stage approach performing separately feature selection and model selection was [16], by Alexandre Q., in which first selects a subset of features and then selects the best model.

IV. Methods and materials

On this experiment, we will make use of this tools into a Decoupled AutoML approach similar to [16], but changing the order of the phases.

The first phase is a simple model selection using GRID method. At the end, the there'll be a set of four models, the best tunning per classifier: Logistic Regression, KNN, SVC and Random Forest. Those four models where chosen due to their popularity. Multi-layer perceptron tuning was considered too complex for this experiment.

The second phase will use EDAs to reduce the number of features and to select the best model to be used with.

The dropout dataset [2] is the dataset target of the experiment.

A. First phase: Model selection

For the first phase, which consists on selecting the best models for each classifier, it'll be used the scikit-learn module [7].

The hyperparameters for each classifier will be the following:

- SVC:

$$\begin{aligned} 0.001 \leq & C \leq 0.999, \text{ with a step of } 0.002 \\ & kernel \in \{\text{'poly'}, \text{'sigmoid'}, \text{'rbf'}\} \\ & gamma \in \{\text{'scale'}, \text{'auto'}\} \end{aligned}$$

- KNN:

$$\begin{aligned} 5 \leq & n \leq 100, \text{ with a step of } 5 \\ & weights \in \{\text{'uniform'}, \text{'distance'}\} \end{aligned}$$

- RandomForest:

$$\begin{aligned} 2 \leq & n \leq 30, \text{ with a step of } 2 \\ 2 \leq & ss_{\min} \leq 28, \text{ with a step of } 2 \\ 5 \leq & \#\{e\} \leq 95, \text{ with a step of } 5 \\ 3 \leq & d_{\max} \leq 29, \text{ with a step of } 2 \end{aligned}$$

- LogisticRegression:

$$\begin{aligned} 0.001 \leq & C \leq 0.999, \text{ with a step of } 0.002 \\ & penalty \in \{\text{'elasticnet'}, \text{'l2'}\} \\ 0.01 \leq & l1_{\text{ratio}} \leq 0.99, \text{ with a step of } 0.01 \\ & solver \in \{\text{'saga'}\} \end{aligned}$$

The reason of using elasticnet in the logistic regression is that it makes use of both L1 and L2 regularization, which can be useful to avoid overfitting. Unfortunately, saga is the only solver which supports the implementation of elasticnet.

B. Second phase: Dimensionality reduction

On the second phase DEAP [17] will be used to implement the Estimation of Distribution Algorithm. For this one, the proposed approach considered was the wrapper implementation with only the best model got from the first phase.

We consider each individual I as set compound by two sets A and B, where

$$\begin{aligned} & A = \{a | a \text{ represents a feature}\} \\ & B = \{b | b \text{ represents a model}\} \\ & \forall a, b \in \mathbb{B} \\ & I = A \cup B \end{aligned} \tag{1}$$

The parameters will be the following:

- Number of generations: 1250
- Number of individuals per generation: 500
- Subset size range=(2, 15)
- Elite fraction: 0.2
- Mutation rate: 0.1

V. Experiments and results

The implementation was made using OOP for easy management, and executed inside a COLAB notebook regarding the needed computational resources.

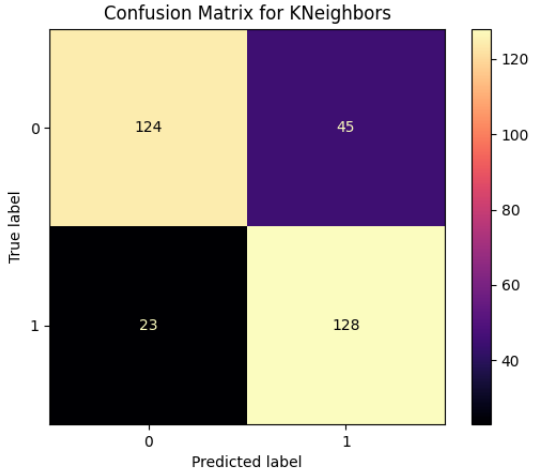

Figure 1. Confusion matrix for the KNN model.

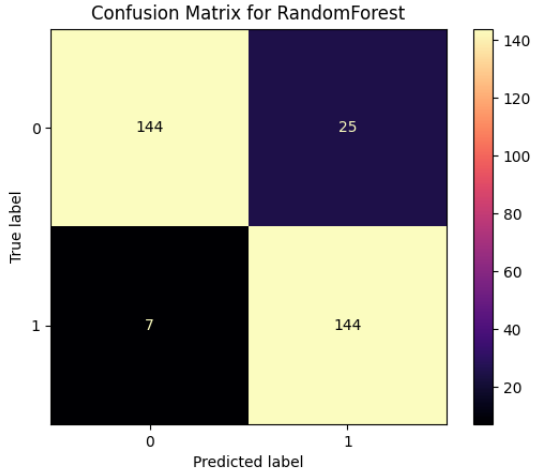

Figure 3. Confusion matrix for the Random Forest model.

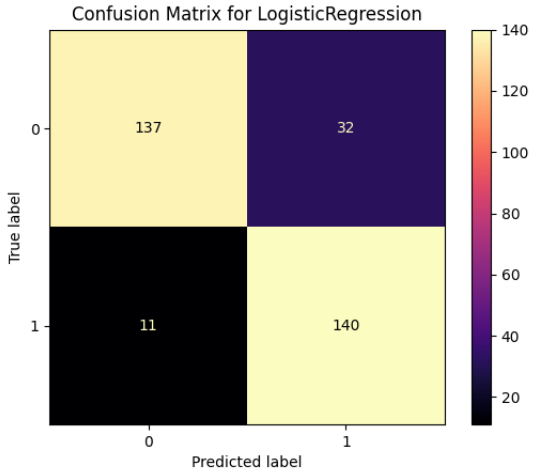

Figure 2. Confusion matrix for the Logistic Regression model.

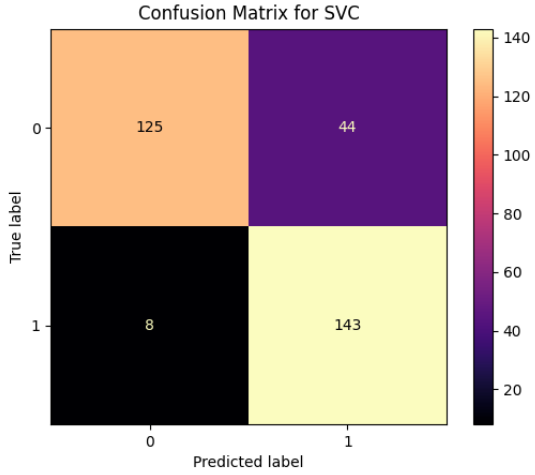

Figure 4. Confusion matrix for the Support Vector Classifier model.

A. *First phase results*

- RandomForest
  - best_accuracy: 0.896875
  - best_f1_macro: 0.8966678007931858
  - best_model:
    - clf__max_depth: 19
    - clf__min_samples_split: 2
    - clf__n_estimators: 75
  - best_estimator: Pipeline(steps=[('scaler', StandardScaler()), ('clf', RandomForest-Classifier(max_depth=19, n_estimators=75))])
  - score: 0.9
- **RandomForest**
  - `best_accuracy`: 0.896875
  - `best_f1_macro`: 0.8966678007931858
  - `best_params`:
    - `max_depth`: 19
    - `min_samples_split`: 2
    - `n_estimators`: 75
    - `scaler`: StandardScaler
  - `test_score`: 0.9

em **Logistic Regression**

  - `best_accuracy`: 0.8836
  - `best_f1_macro`: 0.8830
  - `best_params`:
    * `C`: 0.093
    * `penalty`: elasticnet
    * `l1_ratio`: 0.0
    * `solver`: saga
    * `scaler`: StandardScaler
  - `test_score`: 0.8656
- **SVC**
  - `best_accuracy`: 0.8336
  - `best_f1_macro`: 0.8300
  - `best_params`:
    * `C`: 0.071
    * `kernel`: sigmoid
    * `gamma`: scale
    * `scaler`: StandardScaler
  - `test_score`: 0.8375
- **KNeighbors**
  - `best_accuracy`: 0.8156
  - `best_f1_macro`: 0.8144
  - `best_params`:
    * `n_neighbors`: 10
    * `weights`: uniform
    * `scaler`: StandardScaler
  - `test_score`: 0.7875

B. *Second phase results*

After that, those models were used to perform the feature selection. But when looking at the history, it seems that each population gave the same best individual. Here's a plot of the evolution of the best individual.

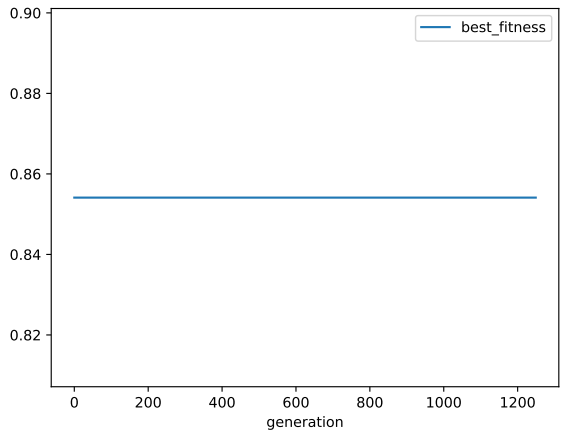

Figure 5. Evolution of the best individual, per generation.

About the feature reduction performance vs. full dataset performance, we can see that there is no significant difference between the two, but that the low accuracy was reduced in around 2% compared to the full dataset.

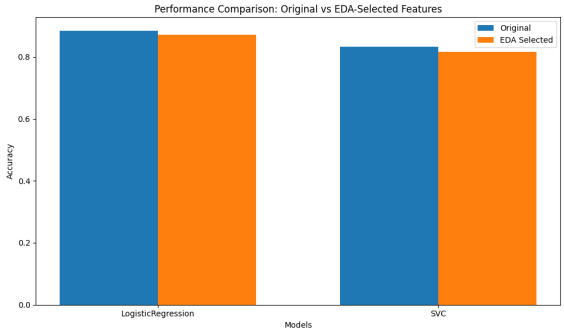

Figure 6. Performance comparison between full dataset and feature reduced dataset.

Table I
Performance comparison between full dataset and feature reduced dataset.

Metrics	**LogisticRegression**	**SVC**
Original Accuracy	0.8836	0.8336
EDA Accuracy	0.8706	0.8156
Improvement	-0.0130	-0.0180
Improvement (%)	-1.47%	-2.16%

VI. Discussion

As shown on the results, the EDA convergence gets stuck on the same individual, no evolution, no mutations. There is work to do, about code depuration, algorithm improvements, hyperparameters tuning, among others.

VII. Conclusion

This experiment haven't demonstrated the capabilities of this Decoupled Approach combining Estimation of Distribution Algorithm and GridSearchCV methods, it's needed more work to improve this research and experimenting with different variations to confirm or reject the fact that this approach is particularly unuseful.

References

[1] *Automated Machine Learning: Methods, Systems, Challenges*. Springer International Publishing, 2019, isbn: 9783030053185. doi: 10.1007/978-3-030-05318-5 [Online]. Available: http://dx.doi.org/10.1007/978-3-030-05318-5

[2] M. V. M. Valentim Realinho, *Predict students' dropout and academic success*, 2021. doi: 10.24432/C5MC89 [Online]. Available: https://archive.ics.uci.edu/dataset/697

[3] T. Anderson and J. Adelusi, "Data cleaning and preprocessing for noisy datasets: Techniques, challenges, and solutions," Oct. 2024.

[4] D. H. Wolpert, "The lack of a priori distinctions between learning algorithms," *Neural Computation*, vol. 8, no. 7, pp. 1341–1390, 1996. doi: 10.1162/neco.1996.8.7.1341

[5] R. E. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

[6] *Automated Machine Learning: Methods, Systems, Challenges*. Springer International Publishing, 2019, ch. 4.3, isbn: 9783030053185. doi: 10.1007/978-3-030-05318-5 [Online]. Available: http://dx.doi.org/10.1007/978-3-030-05318-5

[7] F. Pedregosa et al., "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, no. null, pp. 2825–2830, Nov. 2011, issn: 1532-4435.

[8] Y. Saeys, T. Abeel, and Y. Van de Peer, "Robust feature selection using ensemble feature selection techniques," in *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 2008, pp. 313–325, isbn: 9783540874812. doi: 10.1007/978-3-540-87481-2_21 [Online]. Available: http://dx.doi.org/10.1007/978-3-540-87481-2_21

[9] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1, pp. 273–324, 1997, Relevance, issn: 0004-3702. doi: https://doi.org/10.1016/S0004-3702(97)00043-X [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000437029700043X

[10] R. Panthong and A. Srivihok, "Wrapper feature subset selection for dimension reduction based on ensemble learning algorithm," *Procedia Computer Science*, vol. 72, pp. 162–169, 2015, issn: 1877-0509. doi: 10.1016/j.procs.2015.12.117 [Online]. Available: http://dx.doi.org/10.1016/j.procs.2015.12.117

[11] S. S. Durbha, R. L. King, and N. H. Younan, "Wrapper-based feature subset selection for rapid image information mining," *IEEE Geoscience and Remote Sensing Letters*, vol. 7, no. 1, pp. 43–47, 2010. doi: 10.1109/LGRS.2009.2028585

[12] H. Choi, D. Yeo, S. Kwon, and Y. Kim, "Gene selection and prediction for cancer classification using support vector machines with a reject option," *Computational Statistics & Data Analysis*, vol. 55, no. 5, pp. 1897–1908, May 2011, issn: 0167-9473. doi: 10.1016/j.csda.2010.12.001 [Online]. Available: http://dx.doi.org/10.1016/j.csda.2010.12.001

[13] H. Mühlenbein and G. Paaß, "From recombination of genes to the estimation of distributions i. binary parameters," in *Parallel Problem Solving from Nature — PPSN IV*. Springer Berlin Heidelberg, 1996, pp. 178–187, isbn: 9783540706687. doi: 10.1007/3-540-61723-x_982 [Online]. Available: http://dx.doi.org/10.1007/3-540-61723-X_982

[14] *Estimation of Distribution Algorithms*. Springer US, 2002, isbn: 9781461515395. doi: 10.1007/978-1-4615-1539-5 [Online]. Available: http://dx.doi.org/10.1007/978-1-4615-1539-5

[15] E. Cantú-Paz, "Feature subset selection by estimation of distribution algorithms," in *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO'02, New York City, New York: Morgan Kaufmann Publishers Inc., 2002, pp. 303–310, isbn: 1558608788.

[16] A. Quemy, "Two-stage optimization for machine learning workflow," *Information Systems*, vol. 92, p. 101 483, 2020, issn: 0306-4379. doi: https://doi.org/10.1016/j.is.2019.101483 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306437919305356

[17] F.-M. De Rainville, F.-A. Fortin, M.-A. Gardner, M. Parizeau, and C. Gagné, "Deap: A python framework for evolutionary algorithms," Jul. 2012, pp. 85–92. doi: 10.1145/2330784.2330799