Automatic Feature Engineering And Selection Tabular Data Science

Ron Ben Shimol

Bar Ilan University, Ramat Gan, Israel
ron.ben.shimol@gmail.com

Abstract. Feature engineering techniques provide a process to extract features and use these extra features to improve the quality of results from a machine learning process, compared with supplying only the raw data to the machine learning process. Feature selection techniques provide a process to reduce the number of input variables by eliminating redundant or irrelevant features and narrowing down the set of features to those most relevant to the machine learning model. The process helps to reduce overfitting, improves accuracy and reduces training time. In this paper, we propose a solution to improve the process of designing and selecting features for machine learning models, which is a critical step in developing effective models. We aim to improve model training and performance, by automatic feature engineering and selection, using state of the art methods. We tested the results on four different datasets using Random Forest classifier. on each dataset the accuracy of the model increased.

Problem Description

The two parts of the data science pipeline that we chose to improve are Feature Engineering and Feature Selection. As we mentioned in the Abstract section, the main problem is that it takes a lot of time and effort to research a specific tabular data even though the techniques which will be deployed will be almost identical for most of the tabular datasets. Our vision is to improve these elements and save time and effort by creating a generic library that will automate the process of feature engineering and feature selection to improve the model accuracy.

Solution Overview

we applied the operations we saw in class. We separated the dataset to categorical and numeric columns as the methods used for each column type are different. for one or more columns we applied oneiric and binary operations.

The numeric transformations we used are: imputation - replace missing values using the median along each column. minmax scaling - transform features by scaling each feature to a given range Polynomial Features - Generate a new feature matrix consisting of all polynomial combinations of the features.

the categorical transformations we used are: imputation - replace missing values using the median along each column. one hot encoding - Encode categorical features as a one-hot numeric array.

After performing the transformations we have created a new tabular dataset with the modified and the additional columns. The next step was performing an automated feature selection on the new dataset. Performing feature selection on the new dataset is essential due to the high amount of the new features that were added to the dataset. Our tool compared two algorithms of feature selection and the algorithm with better results was chosen in order to optimize the model training afterwards. the feature selection algorithms we used are: select k best - Select features according to the k highest scores. SelectFromModel - a Meta-transformer for selecting features based on importance weights with the combination of a Random Forest Classifier as an estimator. In addition the tool we created is Sklearn-compatible, therefore it can be used easily in other projects without any effort.

Experimental evaluation

We tested our tool on 4 datasets: Breast Cancer, Iris, Wine, Titanic. For the titanic dataset, in order to run the classifier on the baseline dataset we needed to perform one-hot encoding to convert all the categorical columns. in addition there are missing values in the input data and there was a need to impute the missing values. this is exactly what our solution solves, without automating the process there was a need to perform all the mentioned actions manually. Therefore for the Titanic dataset we didn't perform a further comparison as it's clear that our solution is the undisputed winner. For the rest of the datasets we ran a Random Forest Classifier on both the baseline datasets and the datasets enriched by our tool, then we compared the performance of the trained models. The results were great, our tool enhanced the accuracy of the models and reduced both the false positive and false negative of the models. In the following figure, you can see the results comparison:

dataset: Wine			
	accuracy	**precision**	**recall**
baseline	0.84	0.91	0.83
auto fe	0.91	1	0.92

Fig. 1. Performance comparison on the Wine dataset

dataset: Breast Cancer			
	accuracy	**precision**	**recall**
baseline	0.9	0.95	0.88
auto fe	0.94	0.97	0.94

Fig. 2. Performance comparison on the Breast Cancer dataset

dataset: Iris			
	accuracy	**precision**	**recall**
baseline	0.97	1	0.91
auto fe	1	1	1

Fig. 3. Performance comparison on the Iris dataset

Code. Code can be found at: github link

Related Work

The field of feature engineering and selection has been an active area of research and practice in machine learning and data science.
Venkatesh and Anuradha[3] provide a comprehensive overview of the various feature selection and engineering techniques, including filter, wrapper, and embedded methods. They discuss the pros and cons of each approach and provide insights into their effectiveness for different types of data and models.
Chandrashekar et al.[2] review the different feature selection methods, including statistical, clustering-based, and optimization-based techniques. They also discuss the challenges of feature selection, such as the curse of dimensionality and feature redundancy.
Feature Engineering and Selection: A Practical Approach for Predictive Models[1] by Kuhn and Johnson provides a resource for understanding the concepts of feature engineering and selection in predictive modeling. The authors emphasize the importance of finding better predictor representations and provide illustrative examples using real datasets. The paper includes R programs for reproducing the analyses presented and requires prior exposure to regression modeling and familiarity with the tidyverse ecosystem of R packages.

Conclusion

After working on our tool and learning the methods available in the fields of feature engineering and feature selection, we were amazed by the great variety of the available options . In addition, we saw how time consuming and hard it is to perform all the mentioned actions manually. By performing these actions automatically we were able to improve our datasets without any additional efforts, and we are sure we could improve

the models even more by adding more techniques to our tool. In addition, we created our tool as Sklearn-compatible so others would be able to embed it in their project and benefit from its capabilities. After seeing the results of our project, we understand the need for such tools as by using such tools the data scientist will be able to focus on other elements in the pipeline and will get much better results in less effort and time.

References

[1] Brandon Butcher and Brian J. Smith. "Feature Engineering and Selection: A Practical Approach for Predictive Models". In: *The American Statistician* 74.3 (2020), pp. 308–309. DOI: 10.1080/00031305.2020.1790217. eprint: https://doi.org/10.1080/00031305.2020.1790217. URL: https://doi.org/10.1080/00031305.2020.1790217.

[2] Girish Chandrashekar and Ferat Sahin. "A survey on feature selection methods". In: *Computers Electrical Engineering* 40.1 (2014). 40th-year commemorative issue, pp. 16–28. ISSN: 0045-7906. DOI: https://doi.org/10.1016/j.compeleceng.2013.11.024. URL: https://www.sciencedirect.com/science/article/pii/S0045790613003066.

[3] Jorge R. Vergara and Pablo A. Estévez. "A review of feature selection methods based on mutual information". In: *Neural Computing and Applications* 24.1 (Mar. 2013), pp. 175–186. DOI: 10.1007/s00521-013-1368-0. URL: https://doi.org/10.1007%2Fs00521-013-1368-0.