096290 Explainable AI seminar

Semester 2021w

Anita Narovlyansky ID:207288291

Daniel Goman ID:207077166

Final Submission

Introduction

As data scientists, sometimes as important as accuracy in prediction is interpretability of machine learning models. In some cases, linear models are favored over more complex models because of their interpretability, even though they don't always yield the best results. For this reason, we would like to derive effective estimators of feature relevance in more complex models, such as Random Forest (RF). The solution is the algorithm of Permutation Importance (PIMP).

Despite the fact that feature importance approaches help in interpretation, there is a lack of consensus on how features are quantified, which makes the explanations unreliable. Combination of the results from multiple feature importance quantifiers reduces the variance of estimates and addressing the issue of the lack of agreement. Our assumption was that this will lead to more robust interpretations of the importance of each feature to the prediction.

Our work is based on two feature importance algorithms:

- Permutation Importance[1]
- Permutation Importance using P-value measure[2]

Approach

In our project, we implemented the PIMP, as described in each one of the papers mentioned above. The main difference between the algorithms is the feature importance ranking metric.

Afterwards, we tried to improve the final ranking list using the linear fusion[3] method.

Unfortunately, we found out that this method does not improve the results.

Literature Review

Our work is based on 5 papers:

1. Random Forests[1]

   This paper is talking about the Random Forest procedure. RF is a machine learning method for regression, classification and other tasks. It constructs a combination of decision trees and outputs the class that is the mode of the classes.

   The authors described RF's mode of operation, in particular Characterizing the Accuracy of Random Forests, different methods of initializing it and exploring the RF mechanism.

   In addition, they provided a way to calculate feature importance for RF by determining how score decreases when a feature is permuted. The method is called **Permutation Importance (PIMP)**. The permutation feature importance is defined to be the decrease in a model score when a single feature value is randomly shuffled.

2. Permutation importance: a corrected feature importance measure[2]

   In this paper, the authors talked about a heuristic for normalizing feature importance measures in order to correct feature importance bias (RF models are biased in such a way that categorical variables with a large number of categories are preferred).

   They claimed that the P-value of the observed importance provides a corrected measure of feature importance.

   In addition, they used the PIMP method in order to calculate the feature importance for two real-world case studies, where P-values computed with permutation importance were the ranking score of the importance.

3. Fusion via linear combination of scores[3]

   The authors of the paper present an analysis of the capabilities of the linear combination (LC) model for fusion of information retrieval systems.

   The LC model combines the results lists of multiple IR systems by scoring each document using a weighted sum of the scores from each of the component systems.

   In addition, they described some effects that shows the reason why should linear fusion work:

   - The skimming effect: top-ranked documents are likely to be relevant.
   - The chorus effect: agreement between lists implies relevance.

4. Towards a More Reliable Interpretation of Machine Learning Outputs for Safety-Critical Systems using Feature Importance Fusion[4]

   This paper proposes a Framework, which named as Multi-Method Ensemble for feature importance fusion with the objective of reducing variance in current feature importance estimates.

   The model consists of four stages. The first stage pre-processes the data and in the second step trains the data on multiple ML models. The third step calculate feature importance from each trained ML models using multiple feature importance methods. Finally, the fourth step fuses all feature importance generated from the third step using an ensemble strategy to generate the final feature importance values.

Their framework performed better than traditional single model-agnostic approaches, where only one feature importance method with multiple ML models is employed.

5. Random Search for Hyper-Parameter Optimization[5]

   The authors of the paper propose a way of optimizing hyper-parameters using random search. We search on a volume where each dimension represents a hyperparameter and each point represents one model configuration and randomly sample points in that domain. Compared with models configured by a pure grid search, the researchers found that random search over the same domain is able to find models that are as good or better within a small fraction of the computation time.

Methods

This section describes the design of our methods, including data generation process and the benchmark datasets.

Data Generation

This stage is inspired by the data generation process as described in Rengasamy et al.[4]

The data generated using Python's Scikit-learn library[6] with different parameters (Table 1), in order to represent the variety of the data in the real world. There are no redundant features in all the datasets, therefore we may say that the features used to train the machine learning models are orthogonal. The low correlation requirement ensures that the calculation of feature importance does not use random values because a set of features contains similar information. A combination of values from each parameter in the table creates a dataset and in total we have 15 datasets, each one contains 10,000 samples.

Parameter	**Description**	**Values**
Noise	Standard deviation of Gaussian noise (applies to the data)	0, 2, 4
#features	The total number of the features in the data	60

Table 1: Parameters used to generate the datasets.

Data Pre-Processing

We normalize the input data to the range between zero to one using MinMax normalization. Normalization allows equal weighting for all the features and therefore reduces the bias in the learning process.

Machine Learning Model – Random Forest

As our project focuses on Random Forest, the ML approach employed in our experiment is RF. The optimal hyperparameters are shown in Table 2. We optimize the model using random search for hyperparameters[5], where the data is the generated datasets of the upper and the lower limits of the parameters. The hyperparameters are constant since we measure feature importance.

Hyperparameters	**Values**
#trees	700
Maximum trees' depth	7
Minimum samples before split	2
Bootstrap	True

Table 2: Hyperparameters value for Random Forest.

Evaluation Metrics

We normalize the outputs from each of the models using SoftMax normalization to have them sum up to 1, as the importance of features as we defined it is split among the features and sums up to 1.

In order to evaluate the performance of the different feature importance methods, we employ the evaluation metric: Mean Square Error (MSE).

Results

Table 3 summarizes the results of experiments. In order to compare between the different PIMP methods, we evaluated the classic PIMP, the p-value PIMP and the fusion on the data described above, where informative level is 20%.

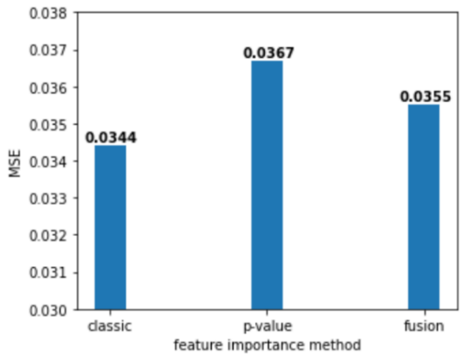

Table 3: MSE values on PIMP methods.

We see that the classic PIMP ranking method gave us the best results. The p-value based PIMP has the worst results, and the result of our fusion method is between them.

Discussion

We have tried to improve the classic PIMP algorithm using Linear Fusion method. Our findings are that the fusion method outperforms the p-value PIMP, but is still inferior to the classic PIMP.

We used Linear Fusion, which takes a weighted average over the output lists. The downfall of using such methodology, as we are generating an "intermediate model", is that relatively to the weak model, the results are elevated, but comparing to the stronger model, the results are decreased.

The idea of using Linear Fusion comes from the Information Retrieval field. We thought the contribution of using such model would follow from the "Chorus Effect" (TODO: reference to chorus effect), which combines sets of good results into a better final result.

We believe that due to this fallacy we failed to obtain improved results over all models involved.

Another potential issue that might have caused this is not feeding a large enough variety of models to our fusion model, which led to fusing between similar models that obtain their results in a similar manner, and thus the variance was too small to achieve any improvement.

During this project, we have learned how to work with data when the goal is not to predict the target feature, but to estimate the ground truth of the importance of each of those explaining features.

Also, we learned how to explore literature which is relevant to our needs.

Future Work

The algorithm and methodology we described can be improved in several ways:

Different Fusion Methods:

We used a specific fusion method – the Linear Fusion. One idea to experiment is to try using different fusion methods, such a CombSum (TODO reference) or ProbFuse (TODO reference). The above fusion methods are more advanced and has shown to achieve better results compared to Linear Fusion, which is why they might serve as adequate alternatives.

More Models:

In our work we did not have much variety among the feature importance algorithms we incorporated in our fusion model, which might have led to getting a model which has narrow understanding of feature importance in the specific dataset we generated.
Therefore, a way to tackle this weakness to feed a larger variety of importance-generating models to our final fusion model.

Appendix

Link to our github rep: https://github.com/DanielGoman/Fusion-Permutation-Importance

[1] Breiman L. Random Forests, Mach. Learn., 2001, vol. 45 (pg. 5-32)
[2] Altmann A, Toloşi L, Sander O, Lengauer T. Permutation importance: a corrected feature importance measure. Bioinformatics 2010, 26:1340–1347.
[3] 1. C. C. Vogt and G. W. Cottrell: “Fusion via linear combination of scores.” Information Retrieval, 1(3) pages 151–173, 1999.(From Diamond T. “Information retrieval using dynamic evidence combination”. Unpublished Ph.D. Thesis proposal, School of Information Studies, Syracuse University, 1998.)
[4] Rengasamy, Divish, Benjamin Rothwell, and Grazziela Figueredo. "Towards a More Reliable Interpretation of Machine Learning Outputs for Safety-Critical Systems using Feature Importance Fusion." arXiv preprint arXiv:2009.05501 (2020).
[5] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, Journal of machine learning research 13 (Feb) (2012) 281–305.
[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel,M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Pas-sos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12(2011) 2825–2830.