SHAP in ML Models

Xiaohe Yang

November 2020

1 Abstract

Machine learning interpretability is an important subject, since the "why" is important for people to understand the model, audit/debug, and widely accept it. As one dataset should have a fixed relationship between the variables, (i.e., the dependency between mortality rate and various health indicators should be determined when the dataset is ready), but the model does not always reveal it. This is reasonable, since the model trained may be inaccurate, or even completely wrong. To study the interpretation from various metrics: Shapley values, Local Surrogate, and Feature Importance measure, this experiment compares their interpretation results for the same model.

2 Background

Machine Learning interpretability has been of great attention for a while, and there are many existing methods. Feature Importance function is built-in with the xgboost package, to show the exact model weights for each feature. There are other methods such as global surrogate, and local surrogate, which refers to training a global or local model to explain the target model.
Another method is Shapley values. In coalitional game theory, a prediction can be explained by assuming that each feature value of the instance is a "player" in a game where the prediction is the payout, and the Shapley values are the fair distribution of the "payout" among the features.

$$\phi_i = \sum_{S \subseteq (X_1, \ldots X_p)) \setminus (X_j)} \frac{|s|!(p - |s| - 1)!}{p!} (val(S \cup (X_j)) - val(S))$$

The SHAP(SHapley Additive exPlanations) package makes uses of both the Shapley values and local surrogate, with improved efficiency. Exact calculation of Shapley values is very inefficient, since each feature requires 2^k (k:number of features) coalitions to calculate, and the total time-efficiency is exponential. With local surrogate and addictive Shapley values, the improvement is significant, especially for TreeSHAP, where it takes only D^2 (D: Tree Depth).

3 Problem Statement

Since there are various metrics for model interpretability, and they all have different mechanism, they may be explaining the same model differently. To study this effect, three metrics, local surrogate (LIME), xgboost Feature Importance, and SHAP(SHapley Additive exPlanations) are used on the same machine learning model, to compare their interpretations. The effect of the accuracy of the model to a specific dataset is also a factor, so models of different accuracy are used.

4 Experiment Setup

For the experiment, Python xgboost package is used, which is the gradient boosted decision tree. Xgboost is a powerful model because it is capable of classification and regression; and with the tree structure, we can make use of the TreeSHAP efficiency improvement. Besides, the xgboost package has the built in Feature Importance function that can show the weights of the model trained.
There are three datasets used in the experiment, two for regression, and one for classification. The first one is the Epidemiologica Follow-Up Dataset, with around 7000 data points. There are 14 features including Age, Sex, BMI, and the task is to predict the mortality rate of a person using those features. Two xgboost models are trained, of accuracy 0.5 and 0.8 respectively. The 0.5 accuracy one only used 30 data points. The similarity metrics here is self-defined, a pairwise comparison of two person about whose mortality measure is higher. So 0.5 accuracy in this metrics is as bad as a random guess.
The second dataset is California Housing, with around 14000 data points. There are 8 features including latitude, total rooms, and house age, and the task is to predict the housing price from the features. Two xgboost models are trained, of accuracy 0.54 and 0.84 respectively, with the 0.54 accuracy one only using 100 data points. The accuracy metrics used is the variance_scored_explained from sklearn package.
The third dataset is Iris - Flower Characteristics Classification, with 3 classes and 150 data points. It is a small dataset, and even the height one xgboost tree can achieve accuracy of 1.0. The 4 features used are the petal/sepal length/width.
For the three metrics used, Feature Importance is tied up to the trained model. LIME trains a local model to explain one specific prediction. Shapley values are calculated to explain one specific prediction as well, but the SHAP package plots every point we want into a summary plot, indicating the influence each feature has on the outcome.

5 Experiment Result

Since LIME only analyzes one prediction, it will be used as a confirmation, while the focus is to study the other two metrics, Feature Importance and SHAP.
For the Epidemiologica Follow-Up Dataset, the three metrics don't really agree with each

other. In the accuracy 0.8 model (Figure 2), the three metrics agree that Age is the most important factor for mortality rate, which is reasonable to think about. And in the bad model of accuracy 0.5 (Figure 1), LIME and SHAP still ranked Age as the most important factor, while Feature Importance function is taking Poverty Index as the top factor. Comparing the plots produced by SHAP of the two models, they show some agreement. Age is ranked as the most important factor. Sex is ranked low in the bad model, but we can see it is because there are only 30 data points. SHAP analyzes each prediction one by one, and from the plot we can see it: Sex in the bad model is a sub-image of the good model. So the hypothesis is that, when more accurate model is trained, in this case, more data points come into the picture, SHAP will lead to one rank of feature importance.

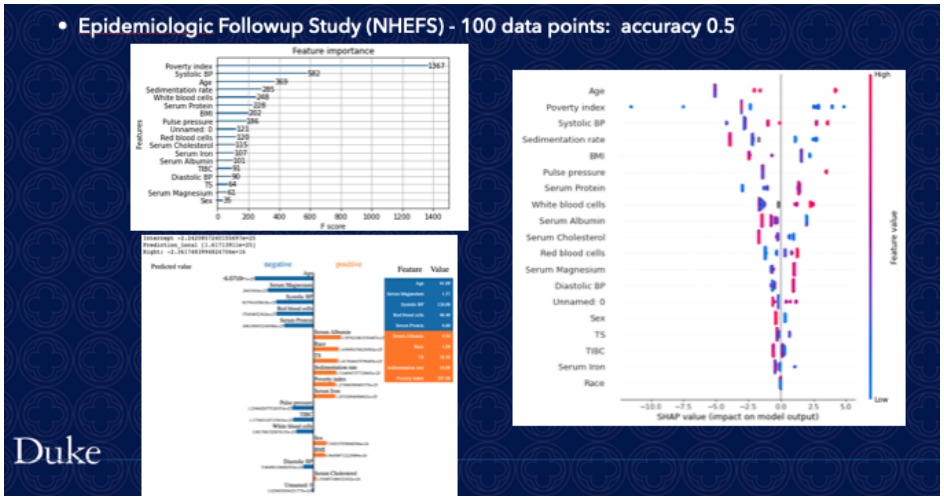

Figure 1: SHAP, LIME and Feature Importance for Epidemiologica Follow-Up, accuracy 0.5

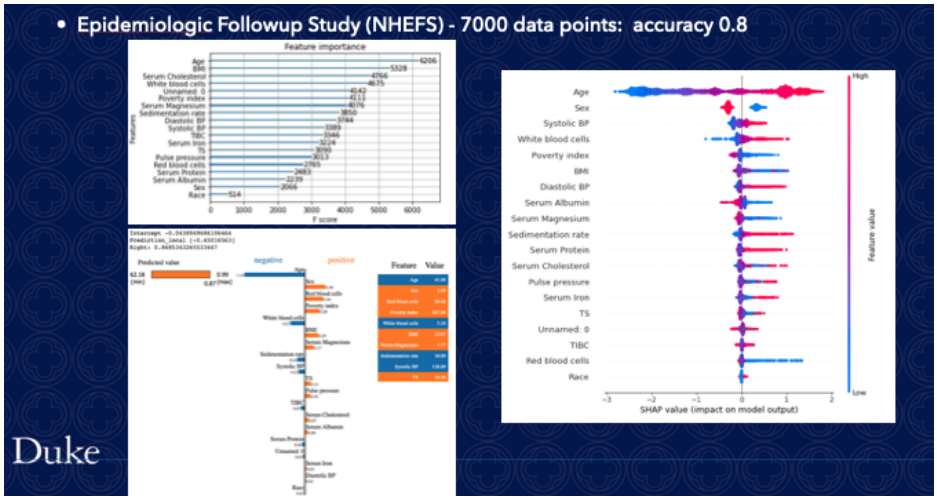

Figure 2: SHAP, LIME and Feature Importance for Epidemiologica Follow-Up, accuracy 0.8

For the California Housing Dataset, the less accurate model is still produced by reducing the training size, and SHAP and LIME still agree with each other on the 0.5(Figure 3) and 0.8(Figure 4) accuracy model. The most important feature is recognized to be latitude on both models for SHAP and LIME, followed by median-income and longitude. For Feature Importance, the most important feature is recognized to be median-income, followed by longitude and latitude. Unlike the previous datasets, the Feature Importance explanation is more consistent, this maybe due to the bad model is still consistent with the better model. And there may be bias in the SHAP explanation plots: usually the whole dataset is used to plot the summary plot, but this dataset has 14000 data points, which causes a crush when calling the summary plot on all points. So the plots are made by just the training data points, which is one fifth of the total data points.

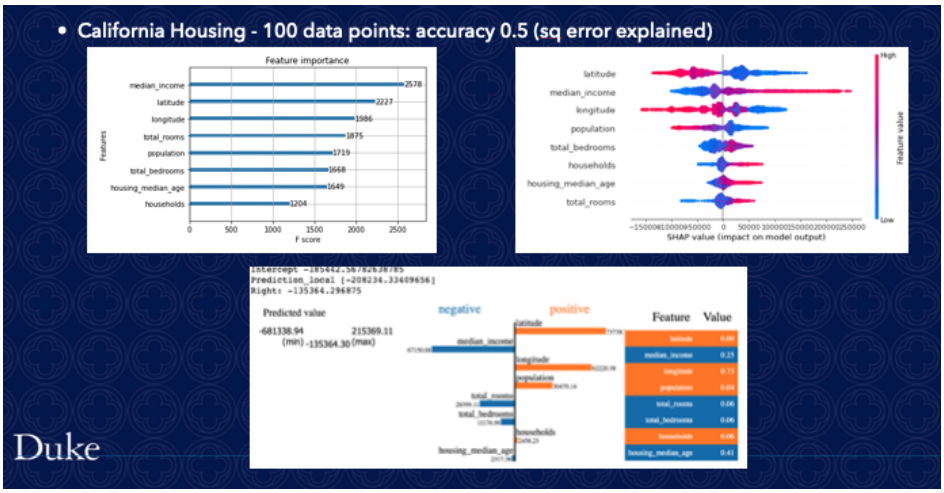

Figure 3: SHAP, LIME and Feature Importance for California Housing Price, accuracy 0.5

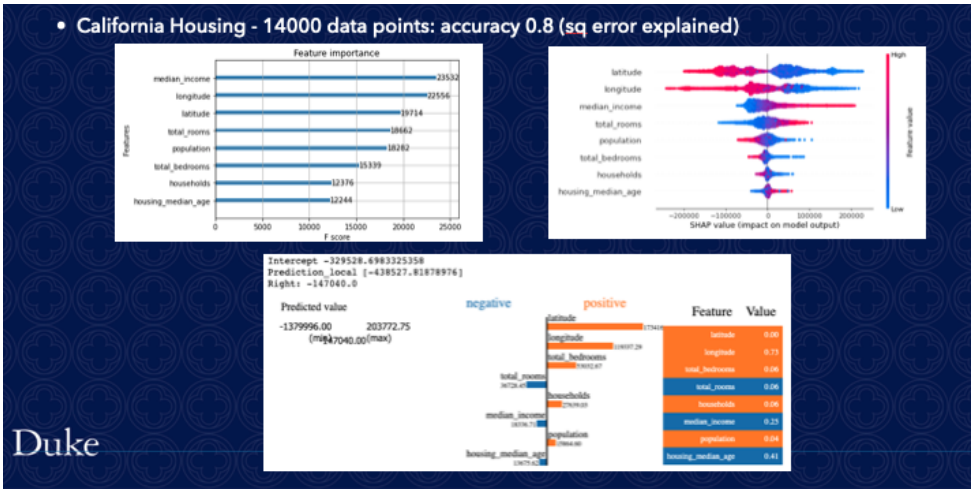

Figure 4: SHAP, LIME and Feature Importance for California Housing Price, accuracy 0.8

For the Iris - Flower Characteristics Classification, the three metrics agree with each other completely, which is different from the previous two datasets.

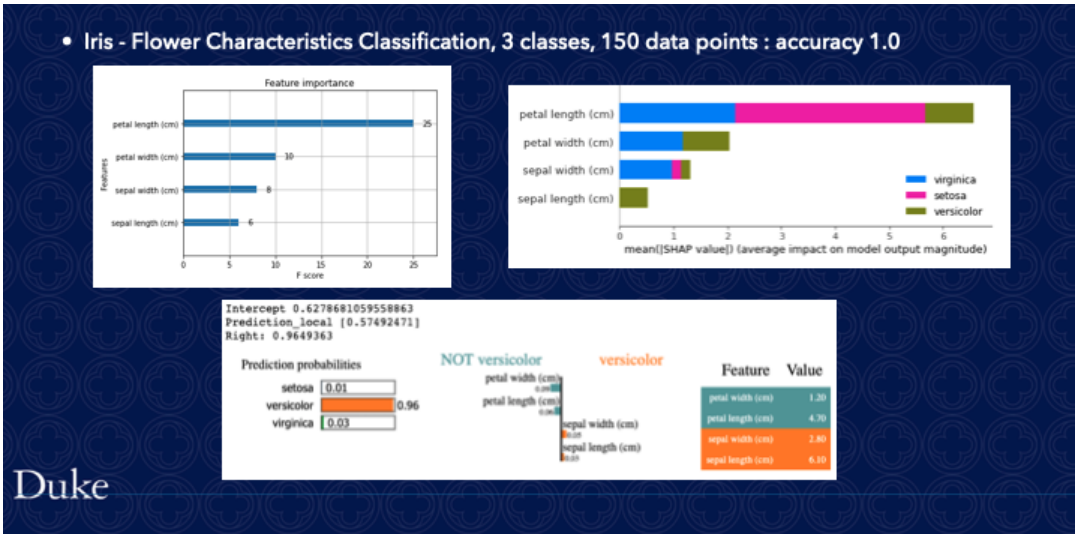

Figure 5: SHAP, LIME and Feature Importance for Iris Flower Classification

6 Analysis

There are multiple potential reasons, comparing this dataset with the previous 2 datasets. It may be the difference between regression and classification, since there are more variance for regression, and in comparison classification outcome has less variance. It may also be related to the number of features, since this dataset only has 4 features, while the previous two has 14 and 8 respectively. It may also because of the size of the dataset, since the Iris dataset has only 150 points, while the previous ones have 7000 and 14000 data points. One last potential reason is the accuracy. While the previous datasets have not really good accuracy with self-defined accuracy metrics, the Iris dataset has perfect 1.0 accuracy.

It is worth noting the credibility of Shapley values over built-in function, Feature Important function in xgboost. Recent studies show that feature attribution methods intrinsic to the model, especially gradient boost trees model like xgboost, can be inconsistent. The current feature attribution methods include gain, reduction in loss, split count, and change in expected output (Sabbas method.) There is studies showing all of the current feature attribution methods can potentially be inconsistent, which means while the model assign more importance to feature A than feature B, the feature attribution method mistakenly take feature B over A (2018, Lundberg.) Shapley values and the SHAP package, on the contrary, are regarded as the only consistent and locally accurate measure of feature importance. So in this experiment, the first dataset especially, since the accuracy metrics is self-defined to be pair-wise comparison, it may have more tendency to encounter inconsistency in feature

distribution. Also, it may be safe to assume the output from SHAP is accurate, and will stay the same, especially when we just reduce the training sample, instead of switching to a completely different machine learning model. So even in the third dataset with only 150 data points, this inconsistency of the Feature Importance function is still there; But due to some factors, the internal feature attribution method does not make a mistake, even though it is not always accurate. The reason may be that those built-in Feature Importance function is more likely to make mistake when the regression task is more complicated: for example, when the dataset has more features, more data points, and the ground truth being less obvious. A further hypothesis is that, maybe changing the accuracy metrics and the model completely (for example from Bayesaian to Random Forest) may change the SHAP output as well as changing Feature Importance. Accuracy drop due to reduction of training size, while model and dataset unchanged, is unlikely to affect SHAP.

7 Notes and Future Work

The code and California housing dataset can be found here:
`https://github.com/littlelittlelotus/CS590-7-ComputationalEcon`.
The Iris dataset and Epidemiologica Follow-Up dataset are builtin in sklearn and SHAP packages.
For the second dataset, there are 14000 data points to be plotted, and it is taking forever and finally caused kernel crush on Jupyter Notebook. However, with around 10000 data points it is doing fine. According to the reference posted by the developer of SHAP, the summary plot function for TreeSHAP should plot within 1 minute for 10000 data points. One other potential reason maybe the effect of the number of features. And as I looked at the other available functions in SHAP, I couldn't find a way to solve this issue. Even though using too many data points to analyze the weight of each feature can be messy to show in one graph, it would be good to have a way to save the partial graphs produced, even though all data points just cannot be plotted onto one graph. It would also be great if there is another plotting function that can plot directly on the previously produced plot. The requirement should be it has to be the same dataset and model, and it will be really convenient. This function can be used to plot large dataset by breaking it into smaller sub-datasets, and it is useful for comparison as well.

There are also more functions in the SHAP package to do experiments on, such as image analyzing DeepSHAP that uses Deep Learning. Other than DeepSHAP, LIME is also capable of analyzing images. Both of them will create features of the images by detecting pixel groups. It should be very interesting to study as well. Unlike xgboost, however, there does not seem to have a built-in function with deep learning packages that can explain the model. While Feature Importance is a very useful function to have in xgboost package, for neural networks it may not be possible, since it is known for the complex internal structure. Due to this reason, predictions made by neural networks are mostly not explainable, at least the model producer package doesn't have any built-in explainer. However, since deep learn-

ing methods are hard to understand, there may be a higher need to explain it, compared to the gradient boosted trees.

It would also be useful to study the limitations of SHAP package. Recent studies focus on the advantage and convenience it brought, and not the limitations. It is good to know when to use the built-in interpretability tools such as Feature Importance and when to use LIME and SHAP. As in the future, everything that can be automated will be automated(Zuboff, 1988), the process of machine learning model training will likely be automated. And at that time, we will be explaining the model, not training the model, so interpretability is an important subject.

8 Reference

https://christophm.github.io/interpretable-ml-book/shap.html
https://github.com/slundberg/shap/blob/e3efee5919afd2c62d3e7c651b3a100a82ba1790/README.md
https://arxiv.org/abs/1705.07874
https://www.aitimejournal.com/@jonathan.hirko/intro-to-classification-and-feature-select
https://www.kaggle.com/goldenoakresearch/us-household-income-stats-geo-locations?select=kaggle_income.csv
https://mljar.com/blog/feature-importance-xgboost/
http://gael-varoquaux.info/interpreting_ml_tuto/content/02_why/04_black_box_interpretation.html
https://www.kaggle.com/yohanb/explaining-xgb-model-with-lime
https://blog.dominodatalab.com/shap-lime-python-libraries-part-2-using-shap-lime/
https://www.kaggle.com/manisood001/california-housing-optimised-modelling
https://xgboost.readthedocs.io/en/latest/parameter.html
https://www.kaggle.com/prashant111/xgboost-k-fold-cv-feature-importance
https://marcotcr.github.io/lime/tutorials/Lime%20-%20multiclass.html
https://github.com/marcotcr/lime/issues/334
https://arxiv.org/abs/1802.03888
https://shoshanazuboff.com/book/books/in-the-age-of-the-smart-machine/