Machine Learning Model Interpretation

Review by L. Querella, PhD, MSc, Data Scientist, Aug 2022

Table of Contents

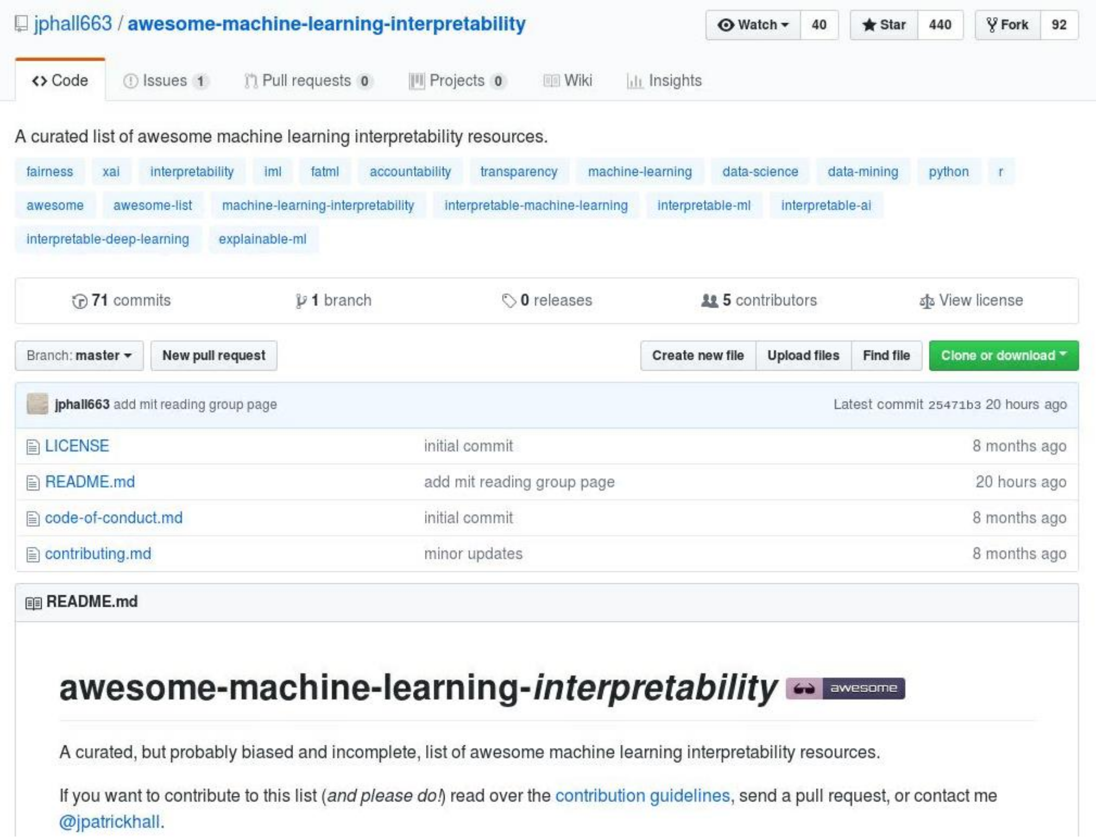
1 References

1.1 Links

1.1.1 General

1.1.1.1 Reviews

- JP Hall (online) – comprehensive and exhaustive



A curated list of awesome machine learning interpretability resources.

jphall663 add mit reading group page — Latest commit 25471b3 20 hours ago

LICENSE	initial commit	8 months ago
README.md	add mit reading group page	20 hours ago
code-of-conduct.md	initial commit	8 months ago
contributing.md	minor updates	8 months ago

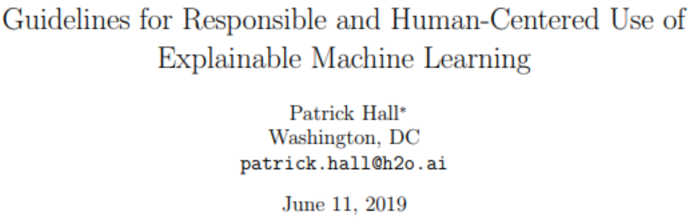
awesome-machine-learning-*interpretability*

A curated, but probably biased and incomplete, list of awesome machine learning interpretability resources.

If you want to contribute to this list (*and please do!*) read over the contribution guidelines, send a pull request, or contact me @jpatrickhall.

  https://github.com/jphall663/awesome-machine-learning-interpretability

- P Hall (06/2019)



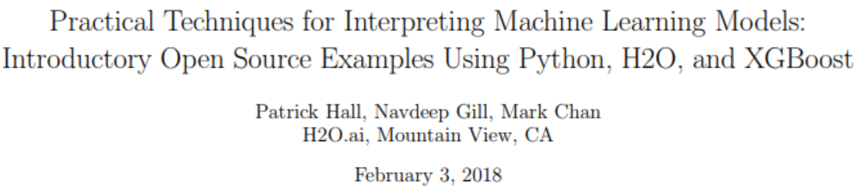
Guidelines for Responsible and Human-Centered Use of Explainable Machine Learning

Patrick Hall*
Washington, DC
patrick.hall@h2o.ai

June 11, 2019

  https://arxiv.org/abs/1906.03533

- P Hall (02/2018)



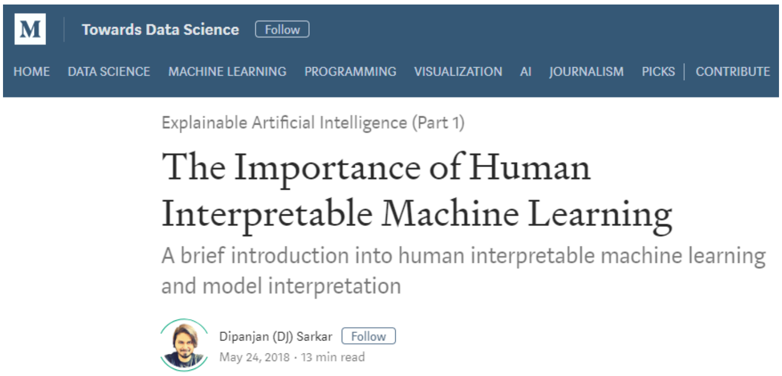
Practical Techniques for Interpreting Machine Learning Models:
Introductory Open Source Examples Using Python, H2O, and XGBoost

Patrick Hall, Navdeep Gill, Mark Chan
H2O.ai, Mountain View, CA

February 3, 2018

  https://fatconference.org/static/tutorials/hall_interpretable18.pdf

- DJ Sarkar (05/2018) – Part 1:



Explainable Artificial Intelligence (Part 1)

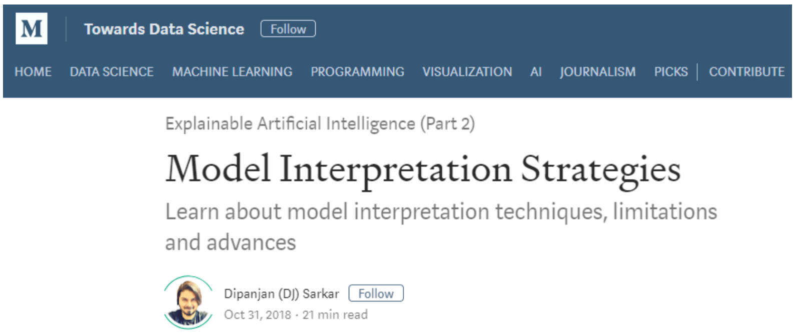
The Importance of Human Interpretable Machine Learning

A brief introduction into human interpretable machine learning and model interpretation

Dipanjan (DJ) Sarkar Follow
May 24, 2018 · 13 min read

https://towardsdatascience.com/human-interpretable-machine-learning-part-1-the-need-and-importance-of-model-interpretation-2ed758f5f476

- DJ Sarkar (10/2018) – Part 2:



Explainable Artificial Intelligence (Part 2)

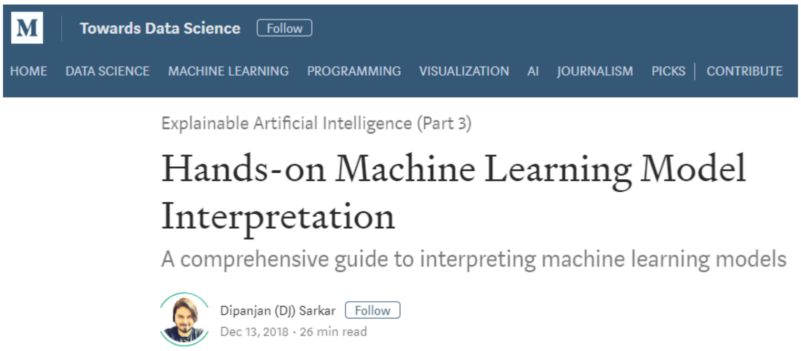
Model Interpretation Strategies

Learn about model interpretation techniques, limitations and advances

Dipanjan (DJ) Sarkar Follow
Oct 31, 2018 · 21 min read

https://towardsdatascience.com/explainable-artificial-intelligence-part-2-model-interpretation-strategies-75d4afa6b739

- DJ Sarkar (12/2018) – Part 3:



Explainable Artificial Intelligence (Part 3)

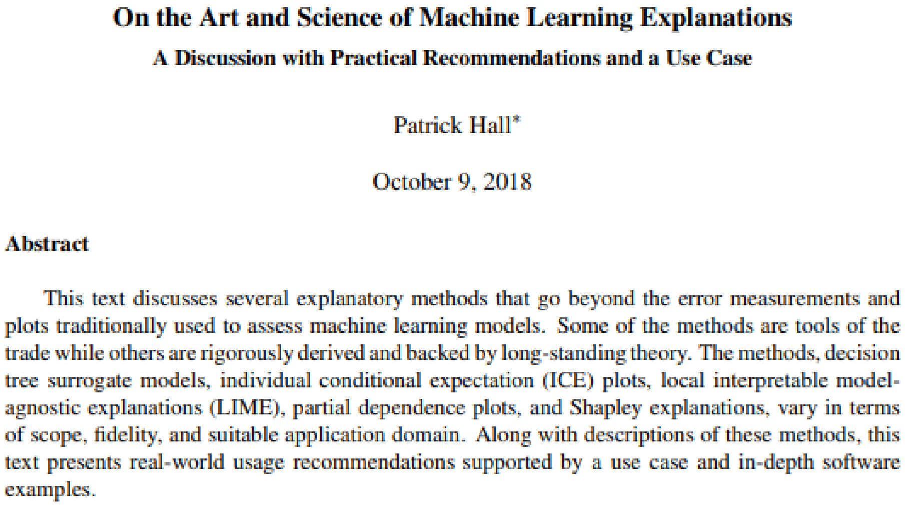
Hands-on Machine Learning Model Interpretation

A comprehensive guide to interpreting machine learning models

Dipanjan (DJ) Sarkar Follow
Dec 13, 2018 · 26 min read

https://towardsdatascience.com/explainable-artificial-intelligence-part-3-hands-on-machine-learning-model-interpretation-e8ebe5afc608

- P. Hall (10/2018)



On the Art and Science of Machine Learning Explanations

A Discussion with Practical Recommendations and a Use Case

Patrick Hall*

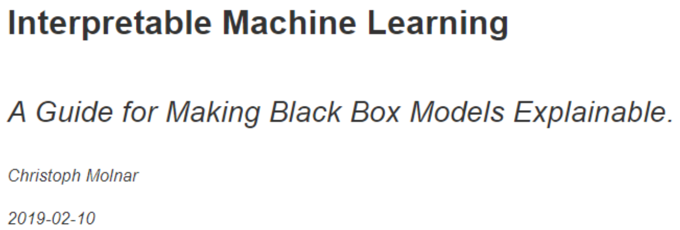
October 9, 2018

Abstract

This text discusses several explanatory methods that go beyond the error measurements and plots traditionally used to assess machine learning models. Some of the methods are tools of the trade while others are rigorously derived and backed by long-standing theory. The methods, decision tree surrogate models, individual conditional expectation (ICE) plots, local interpretable model-agnostic explanations (LIME), partial dependence plots, and Shapley explanations, vary in terms of scope, fidelity, and suitable application domain. Along with descriptions of these methods, this text presents real-world usage recommendations supported by a use case and in-depth software examples.

  https://arxiv.org/pdf/1810.02909.pdf

- C. Molnar (02/2019):



Interpretable Machine Learning

A Guide for Making Black Box Models Explainable.

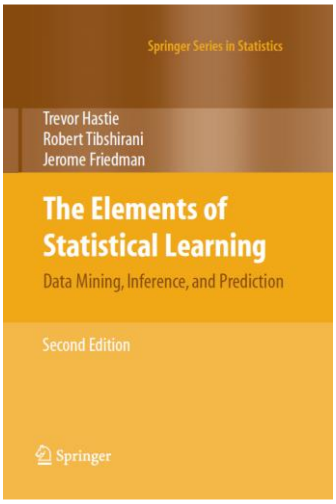
Christoph Molnar

2019-02-10

  Interpretable Machine Learning, A Guide for Making Black Box Models Explainable
  https://christophm.github.io/interpretable-ml-book/

1.1.1.2 Reference textbooks

- ESL II



Springer Series in Statistics

Trevor Hastie
Robert Tibshirani
Jerome Friedman

The Elements of Statistical Learning

Data Mining, Inference, and Prediction

Second Edition

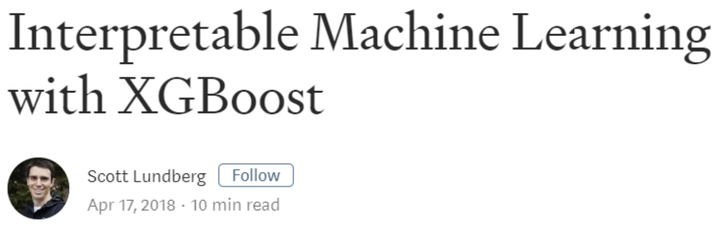
Springer

  - Feature importance:
    - §10.13.1 - Relative Importance of Predictor Variables
  - PDP:
    - §10.13.2 - Partial Dependence Plots

1.1.1.3 Specific algorithms

1.1.1.3.1 Xgboost

- S. Lundberg (04/2018)



Interpretable Machine Learning with XGBoost

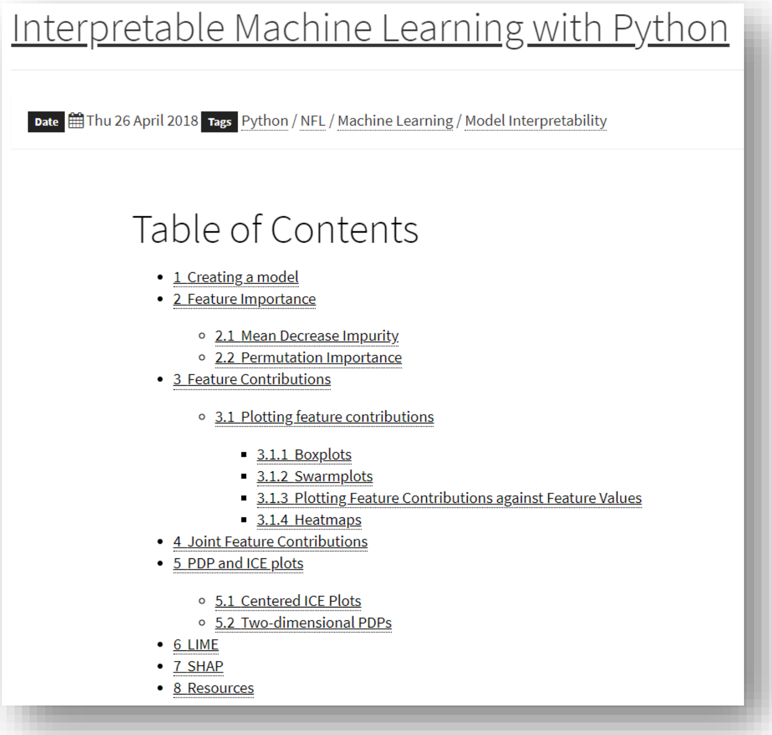
Scott Lundberg Follow

Apr 17, 2018 · 10 min read

https://towardsdatascience.com/interpretable-machine-learning-with-xgboost-9ec80d148d27

1.1.1.3.2 RandomForest

- S. Tjortjoglou (04/2018)



Interpretable Machine Learning with Python

Date Thu 26 April 2018 Tags Python / NFL / Machine Learning / Model Interpretability

Table of Contents

- 1 Creating a model
- 2 Feature Importance
  - 2.1 Mean Decrease Impurity
  - 2.2 Permutation Importance
- 3 Feature Contributions
  - 3.1 Plotting feature contributions
    - 3.1.1 Boxplots
    - 3.1.2 Swarmplots
    - 3.1.3 Plotting Feature Contributions against Feature Values
    - 3.1.4 Heatmaps
- 4 Joint Feature Contributions
- 5 PDP and ICE plots
  - 5.1 Centered ICE Plots
  - 5.2 Two-dimensional PDPs
- 6 LIME
- 7 SHAP
- 8 Resources

http://savvastjortjoglou.com/intrepretable-machine-learning-nfl-combine.html#PDP-and-ICE-plots

1.1.2 PDP (Partial Dependence Plots)

- Kaggle
  https://www.kaggle.com/dansbecker/partial-plots
- S. Tjortjoglou (04/2018)

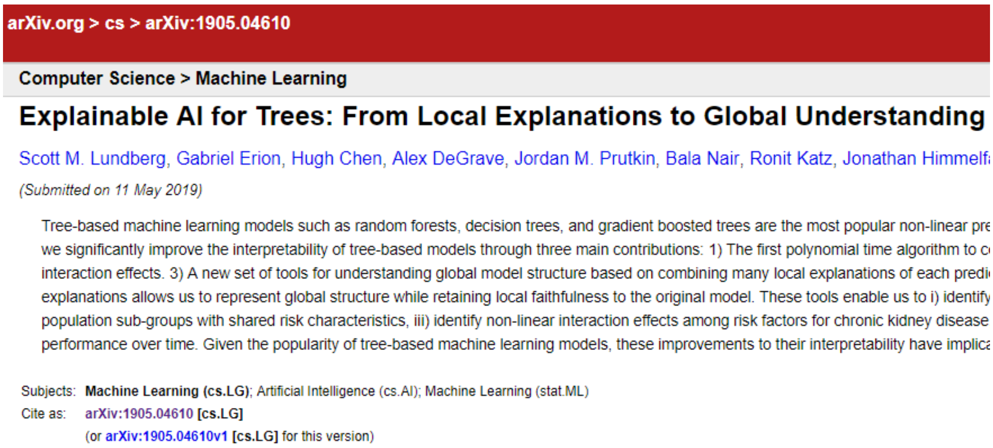
1.1.3 ICE (Individual Conditional Expectation)

- Pycebox Github
  https://github.com/AustinRochford/PyCEbox/blob/master/notebooks/PyCEBox%20Tutorial.ipynb
- S. Tjortjoglou (04/2018)

1.1.4 SHAP (Shapley Additive Explanations)

1.1.4.1 Reference articles

- S.M. Lundberg et al. (05/2019)



arXiv.org > cs > arXiv:1905.04610

Computer Science > Machine Learning

Explainable AI for Trees: From Local Explanations to Global Understanding

Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfa

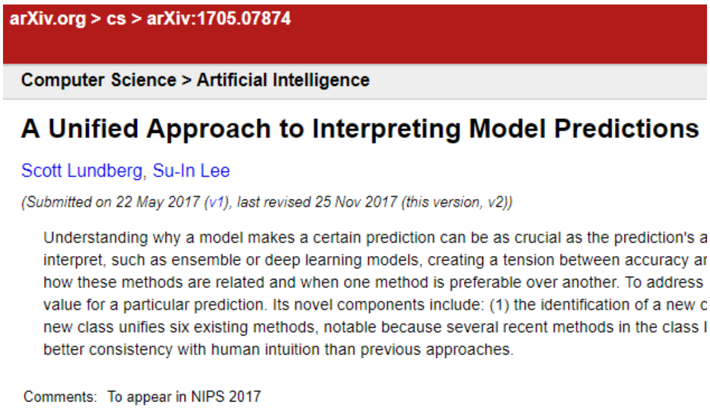
(Submitted on 11 May 2019)

Tree-based machine learning models such as random forests, decision trees, and gradient boosted trees are the most popular non-linear pre
we significantly improve the interpretability of tree-based models through three main contributions: 1) The first polynomial time algorithm to c
interaction effects. 3) A new set of tools for understanding global model structure based on combining many local explanations of each predi
explanations allows us to represent global structure while retaining local faithfulness to the original model. These tools enable us to i) identify
population sub-groups with shared risk characteristics, iii) identify non-linear interaction effects among risk factors for chronic kidney disease
performance over time. Given the popularity of tree-based machine learning models, these improvements to their interpretability have implica

Subjects: **Machine Learning (cs.LG)**; Artificial Intelligence (cs.AI); Machine Learning (stat.ML)
Cite as: arXiv:1905.04610 [cs.LG]
(or arXiv:1905.04610v1 [cs.LG] for this version)

https://arxiv.org/abs/1905.04610

- S.M. Lundberg, S-I. Lee (11/2017)



arXiv.org > cs > arXiv:1705.07874

Computer Science > Artificial Intelligence

A Unified Approach to Interpreting Model Predictions

Scott Lundberg, Su-In Lee

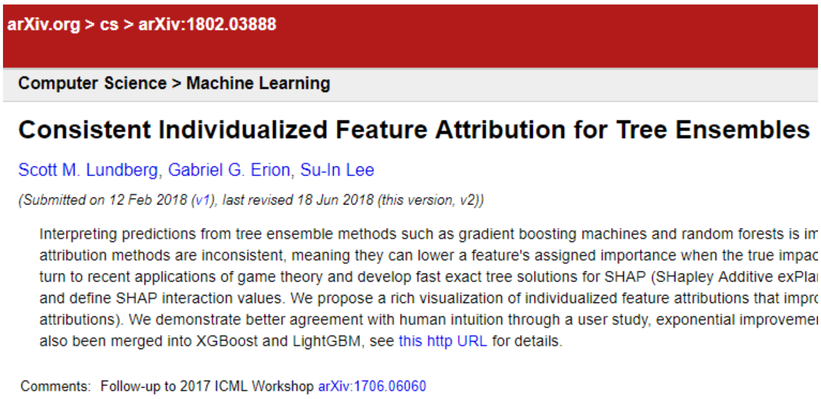
(Submitted on 22 May 2017 (v1), last revised 25 Nov 2017 (this version, v2))

Understanding why a model makes a certain prediction can be as crucial as the prediction's a
interpret, such as ensemble or deep learning models, creating a tension between accuracy ar
how these methods are related and when one method is preferable over another. To address
value for a particular prediction. Its novel components include: (1) the identification of a new c
new class unifies six existing methods, notable because several recent methods in the class l
better consistency with human intuition than previous approaches.

Comments: To appear in NIPS 2017

https://arxiv.org/abs/1705.07874

- [Application to tree methods]
  S.M. Lundberg, G.E. Erion, S-I. Lee (06/2018)



arXiv.org > cs > arXiv:1802.03888

Computer Science > Machine Learning

Consistent Individualized Feature Attribution for Tree Ensembles

Scott M. Lundberg, Gabriel G. Erion, Su-In Lee

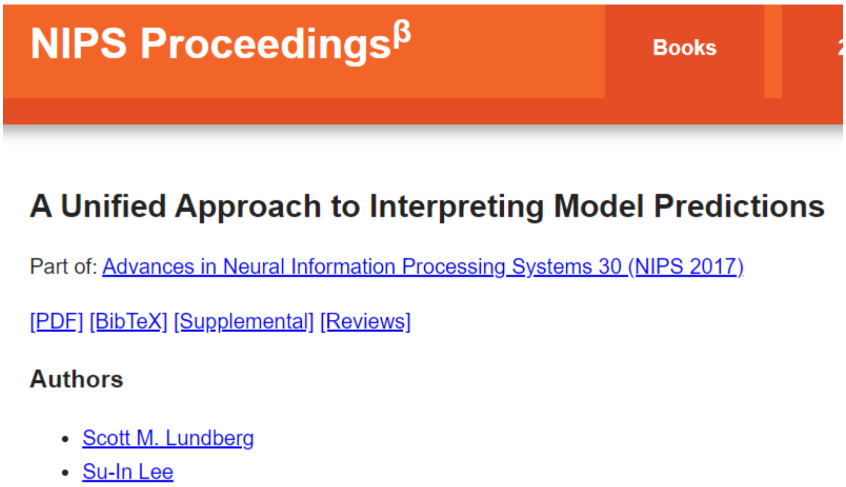
(Submitted on 12 Feb 2018 (v1), last revised 18 Jun 2018 (this version, v2))

Interpreting predictions from tree ensemble methods such as gradient boosting machines and random forests is im
attribution methods are inconsistent, meaning they can lower a feature's assigned importance when the true impac
turn to recent applications of game theory and develop fast exact tree solutions for SHAP (SHapley Additive exPla
and define SHAP interaction values. We propose a rich visualization of individualized feature attributions that impro
attributions). We demonstrate better agreement with human intuition through a user study, exponential improveme
also been merged into XGBoost and LightGBM, see this http URL for details.

Comments: Follow-up to 2017 ICML Workshop arXiv:1706.06060

https://arxiv.org/abs/1802.03888

- [Lundberg and S. Lee (2017)]



A Unified Approach to Interpreting Model Predictions

Part of: Advances in Neural Information Processing Systems 30 (NIPS 2017)

[PDF] [BibTeX] [Supplemental] [Reviews]

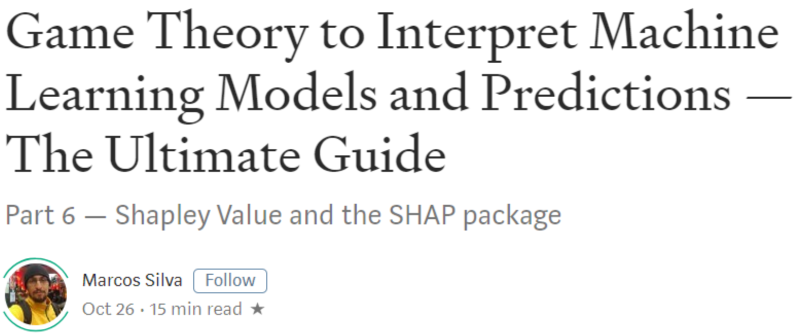
Authors

- Scott M. Lundberg
- Su-In Lee

http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions

1.1.4.2 Non-expert reviews

- M. Silva (10/2019)



Game Theory to Interpret Machine Learning Models and Predictions — The Ultimate Guide

Part 6 — Shapley Value and the SHAP package

Marcos Silva Follow
Oct 26 · 15 min read ★

https://towardsdatascience.com/the-ultimate-guide-using-game-theory-to-interpret-machine-learning-c384cbb6929

- C.M. Wild (01/2018)

One Feature Attribution Method to (Supposedly) Rule Them All: Shapley Values

Cody Marie Wild Follow
Jan 13, 2018 · 10 min read

https://towardsdatascience.com/one-feature-attribution-method-to-supposedly-rule-them-all-shapley-values-f3e04534983d

- (02/2018)

F1 PREDICTOR

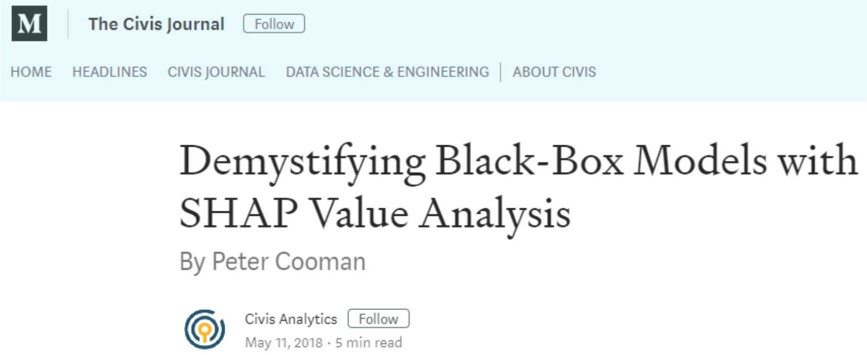
Machine-Learning Based F1 Race Prediction Engine

MODEL INTERPRETABILITY WITH SHAP

February 12, 2018 14 Comments

  http://www.f1-predictor.com/model-interpretability-with-shap/

- P. Cooman (05/2018)



The Civis Journal Follow

HOME HEADLINES CIVIS JOURNAL DATA SCIENCE & ENGINEERING ABOUT CIVIS

Demystifying Black-Box Models with SHAP Value Analysis

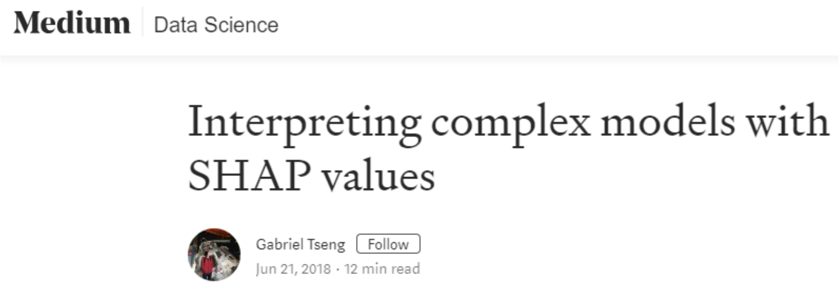
By Peter Cooman

Civis Analytics Follow

May 11, 2018 · 5 min read

  https://medium.com/civis-analytics/demystifying-black-box-models-with-shap-value-analysis-3e20b536fc80

- G. Tseng (06/2018)



Medium Data Science

Interpreting complex models with SHAP values

Gabriel Tseng Follow

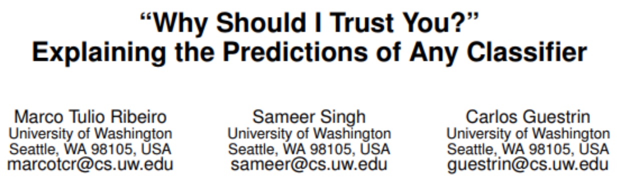
Jun 21, 2018 · 12 min read

  https://medium.com/@gabrieltseng/interpreting-complex-models-with-shap-values-1c187db6ec83

1.1.5 LIME (Local Interpretable Model-Agnostic Explanations)

1.1.5.1 Reference articles

- Ribeiro et al. (2016)
  "Why Should I Trust You?" Explaining the Predictions of Any Classifier by M. T. Ribeiro, S. Singh and C. Guestrin, SIGKDD 2016



"Why Should I Trust You?"
Explaining the Predictions of Any Classifier

Marco Tulio Ribeiro
University of Washington
Seattle, WA 98105, USA
marcotcr@cs.uw.edu

Sameer Singh
University of Washington
Seattle, WA 98105, USA
sameer@cs.uw.edu

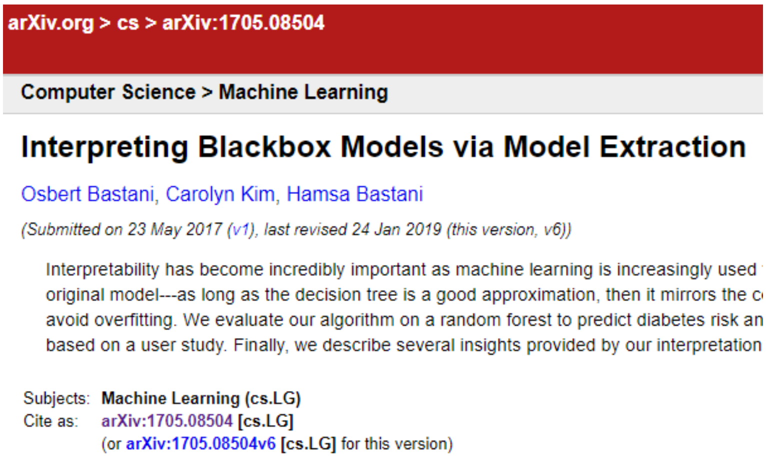
Carlos Guestrin
University of Washington
Seattle, WA 98105, USA
guestrin@cs.uw.edu

http://www.kdd.org/kdd2016/papers/files/rfp0573-ribeiroA.pdf

1.1.6 Surrogate Models – Model Extraction

1.1.6.1 Reference articles

- O Bastani et al. (01/2019)



arXiv.org > cs > arXiv:1705.08504

Computer Science > Machine Learning

Interpreting Blackbox Models via Model Extraction

Osbert Bastani, Carolyn Kim, Hamsa Bastani

(Submitted on 23 May 2017 (v1), last revised 24 Jan 2019 (this version, v6))

Interpretability has become incredibly important as machine learning is increasingly used
original model---as long as the decision tree is a good approximation, then it mirrors the c
avoid overfitting. We evaluate our algorithm on a random forest to predict diabetes risk an
based on a user study. Finally, we describe several insights provided by our interpretation

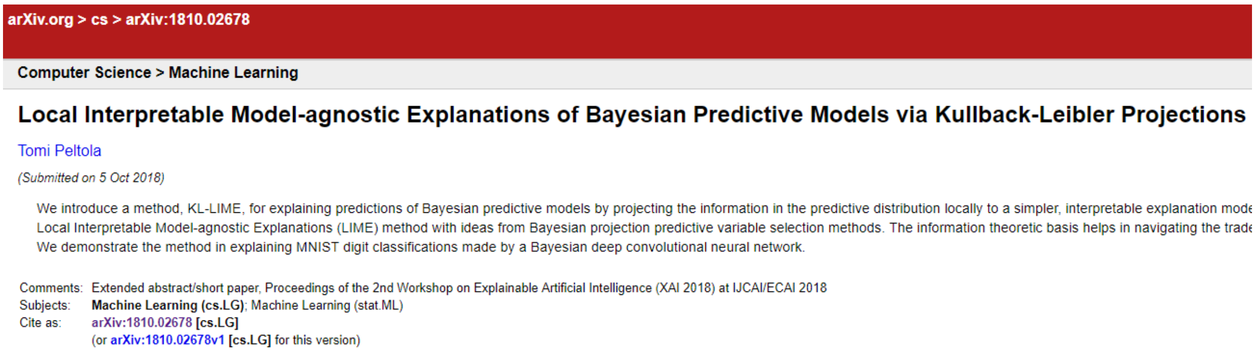
Subjects: **Machine Learning (cs.LG)**
Cite as: arXiv:1705.08504 **[cs.LG]**
(or **arXiv:1705.08504v6 [cs.LG]** for this version)

https://arxiv.org/abs/1705.08504

1.1.7 Other recent methods

1.1.7.1 LIME for Bayesian Deep Convolutional Neural Network

- T Peltola (10/2018)



arXiv.org > cs > arXiv:1810.02678

Computer Science > Machine Learning

Local Interpretable Model-agnostic Explanations of Bayesian Predictive Models via Kullback-Leibler Projections

Tomi Peltola

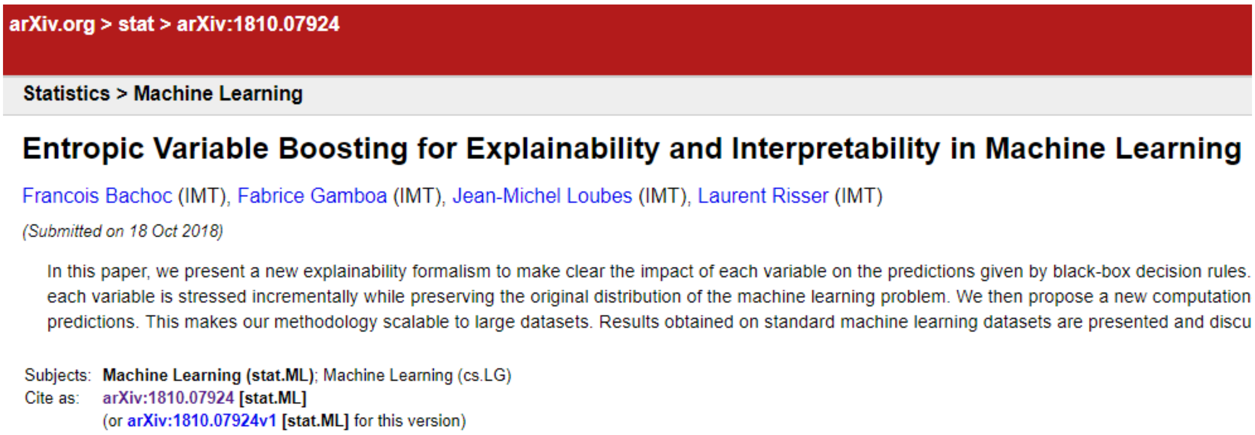
(Submitted on 5 Oct 2018)

We introduce a method, KL-LIME, for explaining predictions of Bayesian predictive models by projecting the information in the predictive distribution locally to a simpler, interpretable explanation mode
Local Interpretable Model-agnostic Explanations (LIME) method with ideas from Bayesian projection predictive variable selection methods. The information theoretic basis helps in navigating the trade
We demonstrate the method in explaining MNIST digit classifications made by a Bayesian deep convolutional neural network.

Comments: Extended abstract/short paper, Proceedings of the 2nd Workshop on Explainable Artificial Intelligence (XAI 2018) at IJCAI/ECAI 2018
Subjects: **Machine Learning (cs.LG)**; Machine Learning (stat.ML)
Cite as: arXiv:1810.02678 **[cs.LG]**
(or **arXiv:1810.02678v1 [cs.LG]** for this version)

https://arxiv.org/abs/1810.02678

- F Bachoc et al. (10/2018)



arXiv.org > stat > arXiv:1810.07924

Statistics > Machine Learning

Entropic Variable Boosting for Explainability and Interpretability in Machine Learning

Francois Bachoc (IMT), Fabrice Gamboa (IMT), Jean-Michel Loubes (IMT), Laurent Risser (IMT)

(Submitted on 18 Oct 2018)

In this paper, we present a new explainability formalism to make clear the impact of each variable on the predictions given by black-box decision rules. each variable is stressed incrementally while preserving the original distribution of the machine learning problem. We then propose a new computation predictions. This makes our methodology scalable to large datasets. Results obtained on standard machine learning datasets are presented and discu

Subjects: **Machine Learning (stat.ML)**; Machine Learning (cs.LG)
Cite as: arXiv:1810.07924 **[stat.ML]**
(or **arXiv:1810.07924v1 [stat.ML]** for this version)

https://arxiv.org/abs/1810.07924

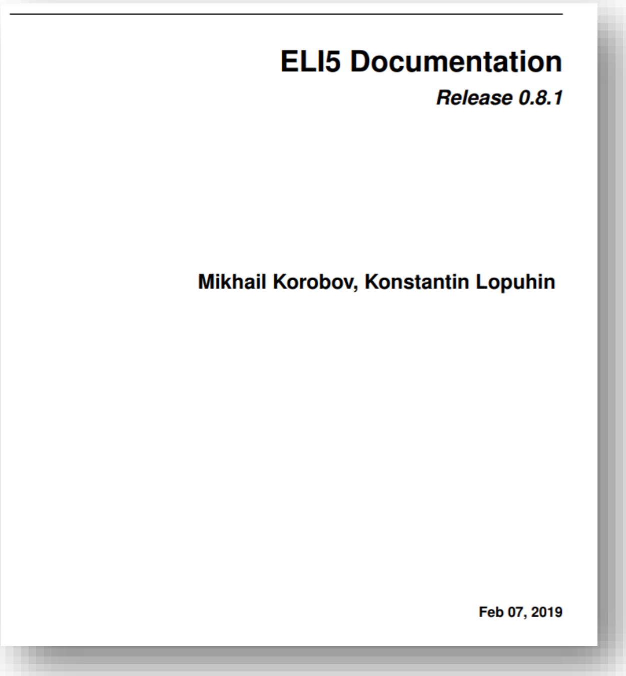
1.1.8 Python Packages

1.1.8.1 Scikit-learn

- PDP
  - https://scikit-learn.org/stable/auto_examples/ensemble/plot_partial_dependence.html

1.1.8.2 ELI5

https://media.readthedocs.org/pdf/eli5/latest/eli5.pdf



ELI5 Documentation
Release 0.8.1

Mikhail Korobov, Konstantin Lopuhin

Feb 07, 2019

1.1.8.3 Skater

- P. Choudhary (03/2018)

DATA SCIENCE

Interpreting predictive models with Skater: Unboxing model opacity

A deep dive into model interpretation as a theoretical concept and a high-level overview of Skater.

By Pramit Choudhary. March 22, 2018

https://www.oreilly.com/ideas/interpreting-predictive-models-with-skater-unboxing-model-opacity

1.1.8.4 SHAP

- S. Lundberg
  https://github.com/slundberg/shap

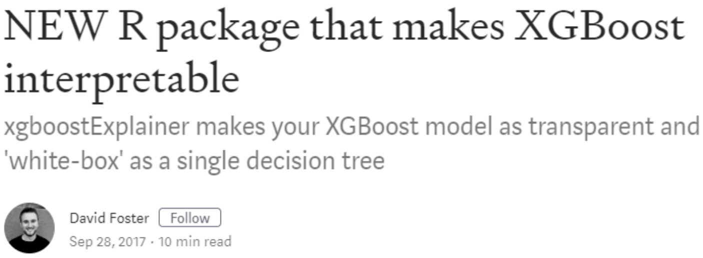
1.1.8.5 PDPBox

https://pdpbox.readthedocs.io/en/latest/

1.1.9 R Packages

1.1.9.1 xgboostExplainer

- D. Foster (09/2017) :



NEW R package that makes XGBoost interpretable

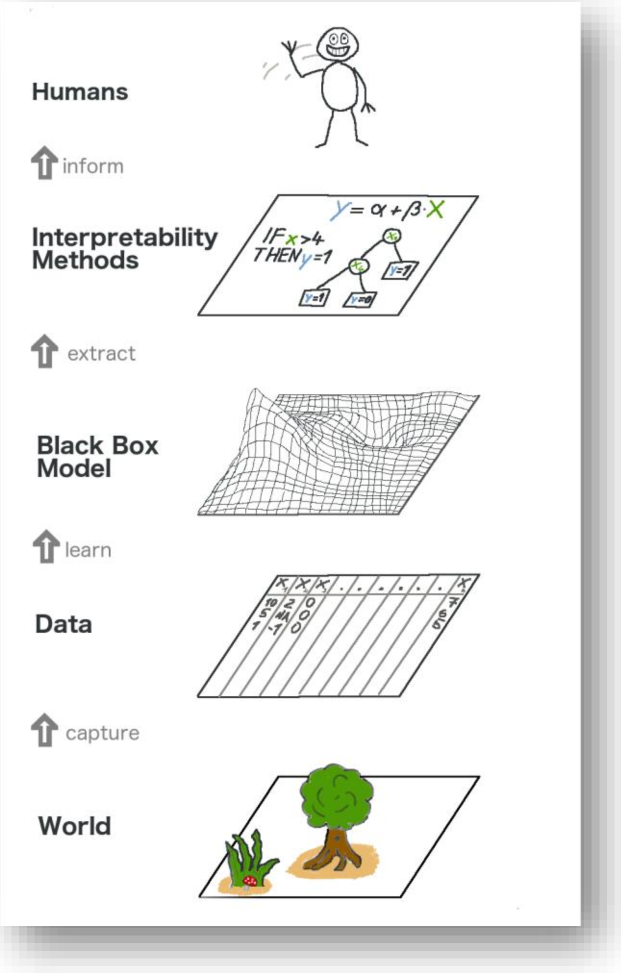
xgboostExplainer makes your XGBoost model as transparent and 'white-box' as a single decision tree

David Foster Follow
Sep 28, 2017 · 10 min read

https://medium.com/applied-data-science/new-r-package-the-xgboost-explainer-51dd7d1aa211

2 Introduction

A machine learning model by itself consists of an algorithm which tries to learn latent patterns and relationships from data without hard-coding fixed rules. Hence, explaining how a model works to the business always poses its own set of challenges. There are some domains in the industry especially in the world of finance like insurance or banking where data scientists often end up having to use more traditional machine learning models (linear or tree-based). The reason being that model interpretability is very important for the business to explain each and every decision being taken by the model. However, this often leads to a sacrifice in performance. This is where complex models like ensembles and neural networks typically give us better and more accurate performance (since true relationships are rarely linear in nature). We, however, end up being unable to have proper interpretations for model decisions. To address and talk about these gaps, I will be writing a series of articles where we will explore some of these challenges in-depth about explainable artificial intelligence (XAI) and human interpretable machine learning.



2.1 Understanding Machine Learning Model Interpretation

The three most important aspects of model interpretation are explained as follows.

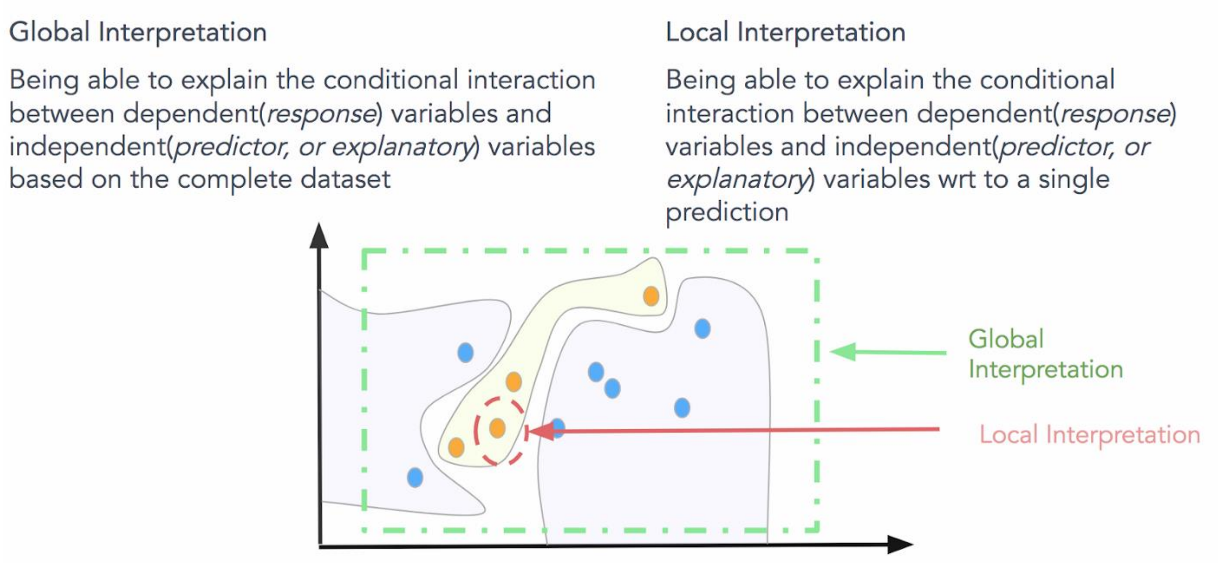
- **What drives model predictions?** We should have the ability to query our model and find out latent feature interactions to get an idea of which features might be important in the decision-making policies of the model. This ensures fairness of the model.

- **Why did the model take a certain decision?** We should also be able to validate and justify why certain key features were responsible in driving certain decisions taken by a model during predictions. This ensures accountability and reliability of the model.
- **How can we trust model predictions?** We should be able to evaluate and validate any data point and how a model takes decisions on it. This should be demonstrable and easy to understand for key stakeholders that the model works as expected. This ensures transparency of the model.

2.2 Criteria for Machine Learning Model Interpretation Methods

There are specific criteria which can be used for categorizing model interpretation methods:

- **Intrinsic or post hoc?** Intrinsic interpretability is all about leveraging a machine learning model which is intrinsically interpretable in nature (like linear models, parametric models or tree based models). Post hoc interpretability means selecting and training a black box model (ensemble methods or neural networks) and applying interpretability methods after the training (feature importance, partial dependency plots). We will focus more on post hoc model interpretable methods in our series of articles.
- **Model-specific or model-agnostic?** Model-specific interpretation tools are very specific to intrinsic model interpretation methods which depend purely on the capabilities and features on a per-model basis. This can be coefficients, p-values, AIC scores pertaining to a regression model, rules from a decision tree and so on. Model-agnostic tools are more relevant to post hoc methods and can be used on any machine learning model. These agnostic methods usually operate by analyzing (and perturbations of inputs) feature input and output pairs. By definition, these methods do not have access to any model internals like weights, constraints or assumptions.
- **Local or global?** This classification of interpretation talks about if the interpretation method explains a single prediction or the entire model behavior? Or if the scope is somewhere in between? We will talk more about global and local interpretations soon.



Global Interpretation

Being able to explain the conditional interaction between dependent(*response*) variables and independent(*predictor, or explanatory*) variables based on the complete dataset

Local Interpretation

Being able to explain the conditional interaction between dependent(*response*) variables and independent(*predictor, or explanatory*) variables wrt to a single prediction

2.3 Traditional Techniques for Model Interpretation

Model interpretation at heart, is to find out ways to understand model decision making policies better. This is to enable fairness, accountability and transparency which will give humans enough confidence to use these models in real-world problems which a lot of impact to business and society. Hence, there are techniques which have existed for a long time now, which can be used to understand and interpret models in a better way. These can be grouped under the following two major categories.

- **Exploratory analysis and visualization techniques** like clustering and dimensionality reduction.
- **Model performance evaluation metrics** like precision, recall, accuracy, ROC curve and the AUC (for classification models) and the coefficient of determination (R-square), root mean-square error, mean absolute error (for regression models)

2.3.1 Exploratory Analysis and Visualization

Dimensionality reduction techniques are very useful here since often we deal with a very large feature space (curse of dimensionality) and reducing the feature space helps us visualize and see what might be influencing a model to take specific decisions. Some of these techniques are as follows.

- **Dimensionality reduction**:
    - Principal Component Analysis (PCA)
    - Self-organizing maps (SOM)
    - Latent Semantic Indexing
- **Manifold Learning**:
    - t-Distributed Stochastic Neighbor Embedding (t-SNE)
    - Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP)
- **Variational autoencoders**:
    - An automated generative approach using variational autoencoders (VAE)
- **Clustering**:
    - Hierarchical Clustering (e.g. HDBSCAN)

2.3.2 Model Performance Evaluation Metrics

Model performance evaluation is a key step in any data science lifecycle for choosing the best model. This enables us to see how well our models are performing, compare various performance metrics across models and choose the best models. This also enables us to tune and optimize hyper-parameters in a model to obtain a model which gives the best performance on the data we are dealing with. Typically there exist certain standard evaluation metrics based on the type of problem we are dealing with.

- **Supervised Learning — Classification**: For classification problems, our main objective is to predict a discrete categorical response variable. The confusion matrix is extremely useful here from which we can derive a whole bunch of useful metrics including accuracy, precision, recall, F1-Score.
  Besides these metrics, we can also use some other techniques like the ROC curve and the AUC score. The area under the ROC curve is a very popular technique to objectively evaluate the performance of a classifier. Here we typically try to balance the True Positive Rate (TPR) and the False Positive Rate (FPR).
- **Supervised Learning — Regression**: For regression problems we can use standard metrics like the coefficient of determination (R-square), the root mean-square error (RMSE) and the mean absolute error (MAE).
- **Unsupervised Learning — Clustering**: For unsupervised learning problems based on clustering we can use metrics like the silhouette coefficient, homogeneity, completeness, V-measure and the Calinski-Harabaz index.

2.3.3 Limitations of Traditional Techniques and Motivation for Better Model Interpretation

These traditional forms of model interpretation might be easy for a data scientist to understand but since they are inherently theoretical and often mathematical, there is a substantial gap in trying to explain these to (non-technical) business stakeholders and trying to decide the success criteria of a project based on these metrics alone. Just telling the business, “I have a model with 90% accuracy” is not sufficient information for them to start trusting the model when deployed in the real world. We need human-interpretable interpretations (HII) of a model’s decision policies which could be explained with proper and intuitive inputs and outputs. This would enable insightful information to be easily shared with peers (analysts, managers, data scientists, data engineers). Using such forms of explanations, which could be explained based on inputs and outputs, might help facilitate better communication and collaboration, enabling businesses to make more confident decisions.

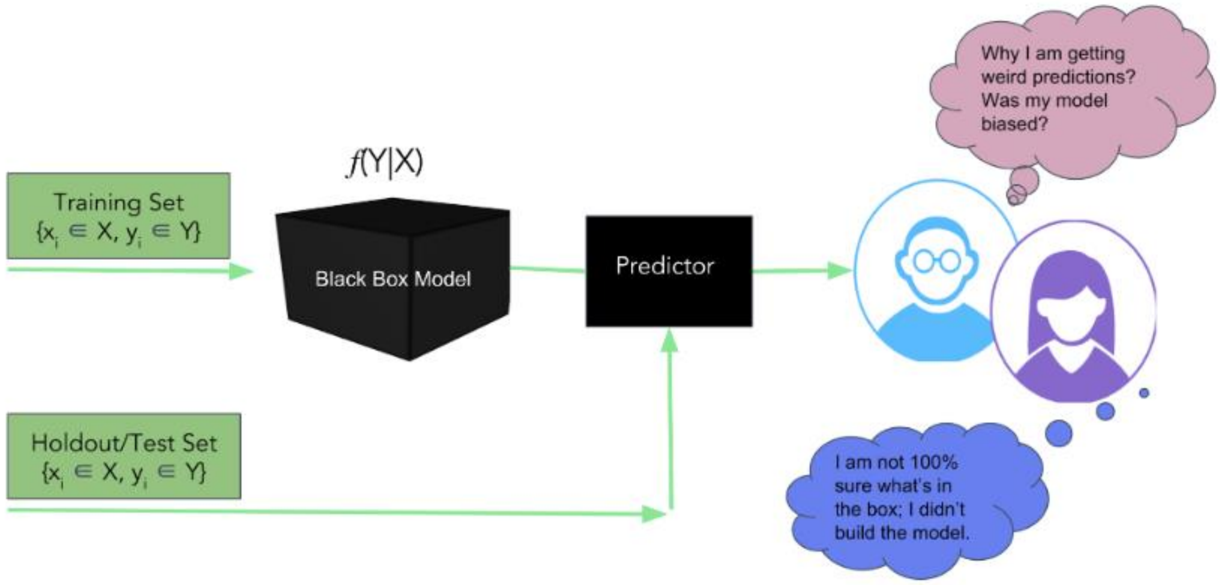

Training Set
$\{x_i \in X, y_i \in Y\}$
$f(Y|X)$
Black Box Model
Predictor
Holdout/Test Set
$\{x_i \in X, y_i \in Y\}$
Why I am getting weird predictions? Was my model biased?
I am not 100% sure what's in the box; I didn't build the model.

Figure 1. Traditional methods for interpreting predictive models are not enough. Image courtesy of Pramit Choudhary.

2.4 The Accuracy vs. Interpretability Trade-off

There exists a typical Trade-off between Model Performance and Interpretability just like we have our standard Bias vs. Variance Trade-off in machine learning. In the industry, you will often hear that business stakeholders tend to prefer models which are more interpretable like linear models (linear\logistic regression) and trees which are intuitive, easy to validate and explain to a non-expert in data science. This increases the trust of people in these models since its decision policies are easier to understand. However, if you talk to data scientists solving real-world problems in the industry, they will tell you that due to the inherent high-dimensional and complex nature of real-world datasets, they often have to leverage machine learning models which might be non-linear and more complex in nature which are often impossible to explain using traditional methods (ensembles, neural networks). Thus, data scientists spend a lot of their time trying to improve model performance but in the process trying to strike a balance between model performance and interpretability.

2.5 Model Interpretation Techniques

There are a wide variety of new model interpretation techniques which try to address the limitations and challenges of traditional model interpretation techniques and try to combat the classic Intepretability vs. Model Performance Trade-off. In this section, we will take a look at some of these techniques and strategies.

2.5.1 Using Interpretable Models

The easiest way to get started with model interpretation is to use models which are interpretable out of the box! This typically includes your regular parametric models like linear regression, logistic regression, tree-based models, rule-fits and even models like k-nearest neighbors and Naive Bayes! A way to categorize these models based on their major capabilities would be:

- **Linearity**: Typically we have a linear model model if the association between features and target is modeled linearly.
- **Monotonicity**: A monotonic model ensures that the relationship between a feature and the target outcome is always in one consistent direction (increase or decrease) over the feature (in its entirety of its range of values).
- **Interactions**: You can always add interaction features, non-linearity to a model with manual feature engineering. Some models create it automatically also.

2.5.2 Feature Importance

Feature importance is generic term for the degree to which a predictive model relies on a particular feature. Typically, a feature's importance is the increase in the model's prediction error after we permuted the feature's values. Frameworks like *Skater* compute this based on an information theoretic criteria, measuring the entropy in the change of predictions, given a perturbation of a given feature. The intuition is that the more a model's decision criteria depend on a feature, the more we'll see predictions change as a function of perturbing a feature. However, frameworks like *SHAP*, use a combination of feature contributions and game theory to come up with SHAP values. Then, it computes the global feature importance by taking the average of the SHAP value magnitudes across the dataset.

The concept behind global interpretations of model-agnostic feature importance is really straightforward.

- We measure a feature's importance by calculating the increase of the model's prediction error after perturbing the feature.
- A feature is "important" if perturbing its values increases the model error, because the model relied on the feature for the prediction.
- A feature is "unimportant" if perturbing its values keeps the model error unchanged, because the model basically ignored the feature for the prediction.

The permutation feature importance measurement was introduced for Random Forests by Breiman (2001). Based on this idea, Fisher, Rudin, and Dominici (2018) proposed a model-agnostic version of the feature importance — they called it Model Reliance.

2.5.2.1 Training vs test data

Interesting discussion in Molnar (2019) (Should I Compute Importance on Training or Test Data?)

2.5.2.2 Boosted trees

Gain, Cover, Frequency

2.5.3 Feature Contributions

NB: Saabas – was shown to be inconsistent (Lundberg et al 2018).

(see Tjortjoglou 2018 for explanations on the method)

2.5.4 Partial Dependence Plots

Partial Dependence describes the marginal impact of a feature on model prediction, holding other features in the model constant. The derivative of partial dependence describes the impact of a feature (analogous to a feature coefficient in a regression model). The partial dependence plot (PDP or PD plot) shows the marginal effect of a feature on the predicted outcome of a previously fit model. PDPs can show if the relationship between the target and a feature is linear, monotonic or more complex. The partial dependence plot is a global method: The method takes into account all instances and makes a statement about the global relationship of a feature with the predicted outcome.

2.5.5 Global Surrogate Models

We have seen various ways to interpret machine learning models with feature importance, dependence plots, less complex models. But is there a way to build interpretable approximations of really complex models? Thankfully we have global surrogate models just for this purpose! A global surrogate model is an interpretable model that is trained to approximate the predictions of a black box model which can essentially be any model regardless of its complexity or training algorithm — this is as model-agnostic as it gets!

The purpose of (interpretable) surrogate models is to approximate the predictions of the underlying model as closely as possible while being interpretable. Fitting a surrogate model is a model-agnostic method, since it requires no information about the inner workings of the black box model, only the relation of input and predicted output is used. The choice of the base black box model type and of the surrogate model type is decoupled. Tree-based models are not too simplistic but interpretable and make a good choice for building surrogate models.

Typically we approximate a more interpretable surrogate model based on our base model which is treated as a black box model. We can then draw conclusions about the black box model by interpreting the surrogate model. Solving machine learning interpretability by using more machine learning!

Skater introduce the novel idea of using *TreeSurrogates* as means for explaining a model's learned decision policies (for inductive learning tasks), which is inspired by the work of Mark W. Craven described as the TREPAN algorithm.

- Choose a dataset. This could be the same dataset that was used for training the black box model or a new dataset from the same distribution. You could even choose a subset of the data or a grid of points, depending on your application.
- For the chosen dataset, get the predictions of your base black box model.
- Choose an interpretable surrogate model (linear model, decision tree, …).
- Train the interpretable model on the dataset and its predictions.
- Congratulations! You now have a surrogate model.
- Measure how well the surrogate model replicates the prediction of the black box model.
- Interpret / visualize the surrogate model.

2.5.6 Local Interpretable Model-agnostic Explanations (LIME)

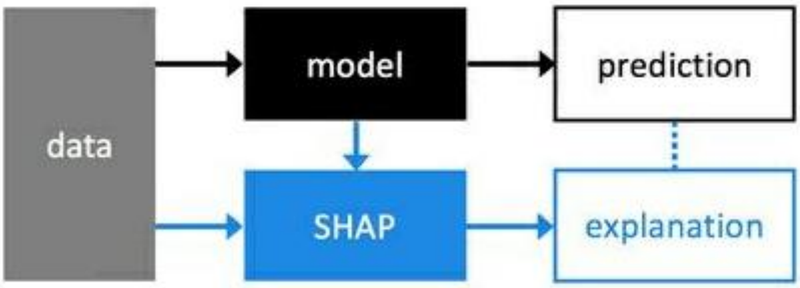
LIME is an algorithm designed to access the behavior of any base estimator (model) using local interpretable surrogate models (e.g. linear classifier/regressor). Such form of comprehensive evaluation helps in generating explanations which are locally faithful but may not align with the global behavior. Basically, LIME explanations are based on local surrogate models. These, surrogate models are interpretable models (like a linear model or decision tree) that are learned on the predictions of the original black box model. But instead of trying to fit a global surrogate model, LIME focuses on fitting local surrogate models to explain why single predictions were made.

The idea is very intuitive. To start with, just try and unlearn what you have done so far! Forget about the training data, forget about how your model works! Think that your model is a black box model with some magic happening inside, where you can input data points and get the models predicted outcomes. You can probe this magic black box as often as you want with inputs and get output predictions. Now, you main objective is to understand why the machine learning model which you are treating as a magic black box, gave the outcome it produced. LIME tries to do this for you! It tests out what happens to you black box model's predictions when you feed variations or perturbations of your dataset into the black box model. Typically, LIME generates a new dataset consisting of perturbed samples and the associated black box model's predictions. On this dataset LIME then trains an interpretable model weighted by the proximity of the sampled instances to the instance of interest. Following is a standard high-level workflow for this.

- Choose your instance of interest for which you want to have an explanation of the predictions of your black box model.
- Perturb your dataset and get the black box predictions for these new points.
- Weight the new samples by their proximity to the instance of interest.
- Fit a weighted, interpretable (surrogate) model on the dataset with the variations.
- Explain prediction by interpreting the local model.

2.5.7 Shapley Values and SHapley Additive exPlanations (SHAP)

SHAP (SHapley Additive exPlanations) is a unified approach to explain the output of any machine learning model. SHAP connects game theory with local explanations, uniting several previous methods and representing the only possible consistent and locally accurate additive feature attribution method based on what they claim! SHAP is an excellent model interpretation framework which is based on adaptation and enhancements to Shapley values which we shall explore in-depth in this section.



Typically, model predictions can be explained by assuming that each feature is a 'player' in a game where the prediction is the payout. The Shapley value — a method from coalitional game theory — tells us how to fairly distribute the 'payout' among the features. Let's take an illustrative example.

SHAP (SHapley Additive exPlanations) assigns each feature an importance value for a particular prediction. Its novel components include: the identification of a new class of additive feature importance measures, and theoretical results showing there is a unique solution in this class with a set of desirable properties. Typically, SHAP values try to explain the output of a model (function) as a sum of the effects of each feature being introduced into a conditional expectation. Importantly, for non-linear functions the order in which features are introduced matters. The SHAP values result from averaging over all possible orderings. Proofs from game theory show this is the only possible consistent approach.

3 Model-Agnostic Methods

3.1 Partial Dependence Plot (PDP)

The partial dependence plot (short PDP or PD plot) shows the marginal effect one or two features have on the predicted outcome of a machine learning model. A partial dependence plot can show whether the relationship between the target and a feature is linear, monotonous or more complex. For example, when applied to a linear regression model, partial dependence plots always show a linear relationship.

The partial dependence plot is a global method: The method considers all instances and gives a statement about the global relationship of a feature with the predicted outcome

For classification where the machine learning model outputs probabilities, the partial dependence plot displays the probability for a certain class given different values for feature(s) in S. An easy way to deal with multiple classes is to draw one line or plot per class.

3.1.1 Advantages

The computation of partial dependence plots is intuitive: The partial dependence function at a particular feature value represents the average prediction if we force all data points to assume that feature value. In my experience, lay people usually understand the idea of PDPs quickly.

If the feature for which you computed the PDP is not correlated with the other features, then the PDPs perfectly represent how the feature influences the prediction on average. In the uncorrelated case, the interpretation is clear: The partial dependence plot shows how the average prediction in your dataset changes when the j-th feature is changed. It is more complicated when features are correlated, see also disadvantages.

Partial dependence plots are easy to implement.

The calculation for the partial dependence plots has a causal interpretation. We intervene on a feature and measure the changes in the predictions. In doing so, we analyze the causal relationship between the feature and the prediction. The relationship is causal for the model – because we explicitly model the outcome as a function of the features – but not necessarily for the real world!

3.1.2 Disadvantages

The realistic maximum number of features in a partial dependence function is two. This is not the fault of PDPs, but of the 2-dimensional representation (paper or screen) and also of our inability to imagine more than 3 dimensions.

Some PD plots do not show the feature distribution. Omitting the distribution can be misleading, because you might overinterpret regions with almost no data. This problem is easily solved by showing a rug (indicators for data points on the x-axis) or a histogram

The **assumption of independence** is the biggest issue with PD plots

Heterogeneous effects might be hidden because PD plots only show the average marginal effects

3.1.3 Packages

- R
  - pdp
  - DALEX
- Python
  - PDPBox

    - Pycebox
    - Scikit-learn
    - Skater

3.2 Individual Conditional Expectation (ICE)

Individual Conditional Expectation (ICE) plots display one line per instance that shows how the instance's prediction changes when a feature changes.

The partial dependence plot for the average effect of a feature is a global method because it does not focus on specific instances, but on an overall average. The equivalent to a PDP for individual data instances is called individual conditional expectation (ICE) plot (Goldstein et al. 2017)

An ICE plot visualizes the dependence of the prediction on a feature for *each* instance separately, resulting in one line per instance, compared to one line overall in partial dependence plots. A PDP is the average of the lines of an ICE plot. The values for a line (and one instance) can be computed by keeping all other features the same, creating variants of this instance by replacing the feature's value with values from a grid and making predictions with the black box model for these newly created instances. The result is a set of points for an instance with the feature value from the grid and the respective predictions.

What is the point of looking at individual expectations instead of partial dependencies? Partial dependence plots can obscure a heterogeneous relationship created by interactions. PDPs can show you what the average relationship between a feature and the prediction looks like. This only works well if the interactions between the features for which the PDP is calculated and the other features are weak. In case of interactions, the ICE plot will provide much more insight.

3.2.1 Packages

- Python
    - Pycebox

4 Python libraries

We cover usage of the following model interpretation frameworks in our tutorial.

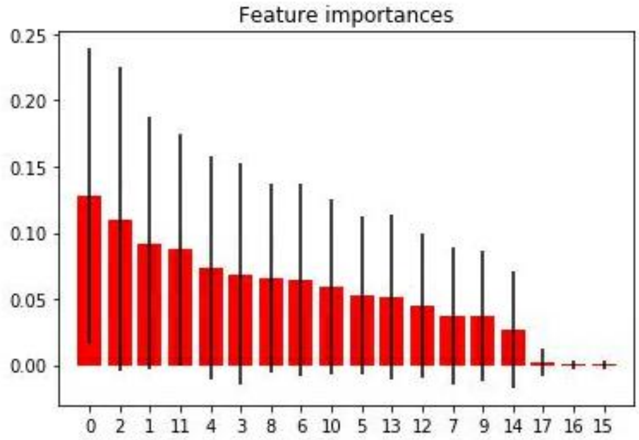
- ELI5
- Skater
- SHAP

The major model interpretation techniques we will be covering in this tutorial include the following.

- Feature Importances
- Partial Dependence Plots
- Model Prediction Explanations with Local Interpretation
- Building Interpretable Models with Surrogate Tree-based Models
- Model Prediction Explanation with SHAP values
- Dependence & Interaction Plots with SHAP

4.1 Scikit-learn

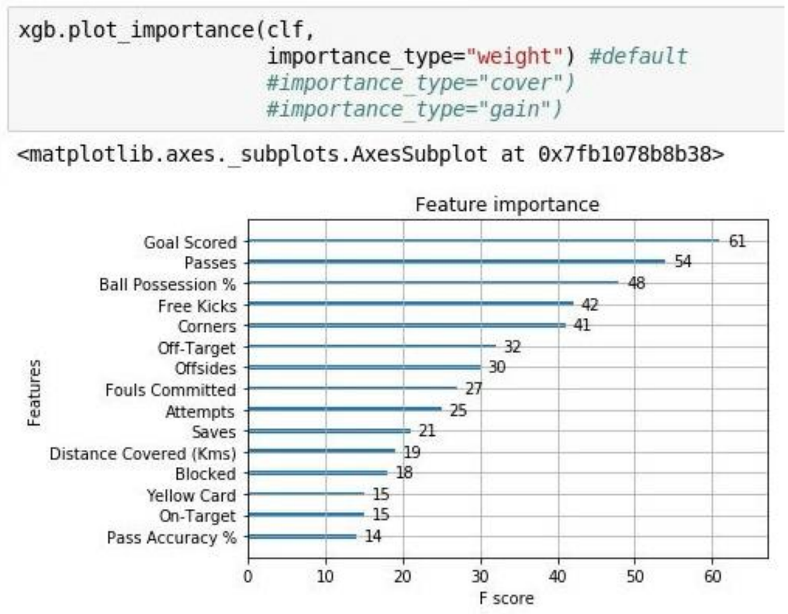
4.1.1 RandomForestClassifier



4.1.2 Xgboost.XGBClassifier

There are three options for measuring feature importance in XGBoost:

- Weight
  - The number of times a feature is used to split the data across all trees.
- Cover
  - The number of times a feature is used to split the data across all trees weighted by the number of training data points that go through those splits.
- Gain
  - The average training loss reduction gained when using a feature for splitting.



```
xgb.plot_importance(clf,
                    importance_type="weight") #default
                    #importance_type="cover")
                    #importance_type="gain")
```

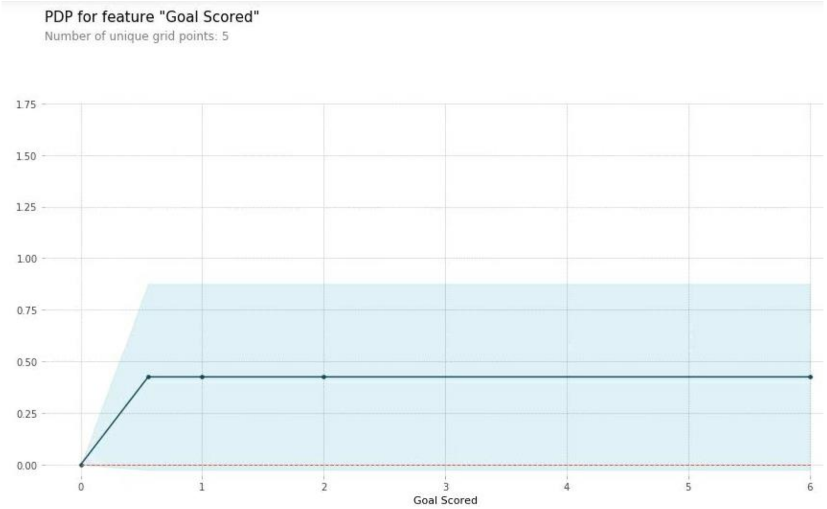
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb1078b8b38>
```

4.2 PDP

4.2.1 PDPBox

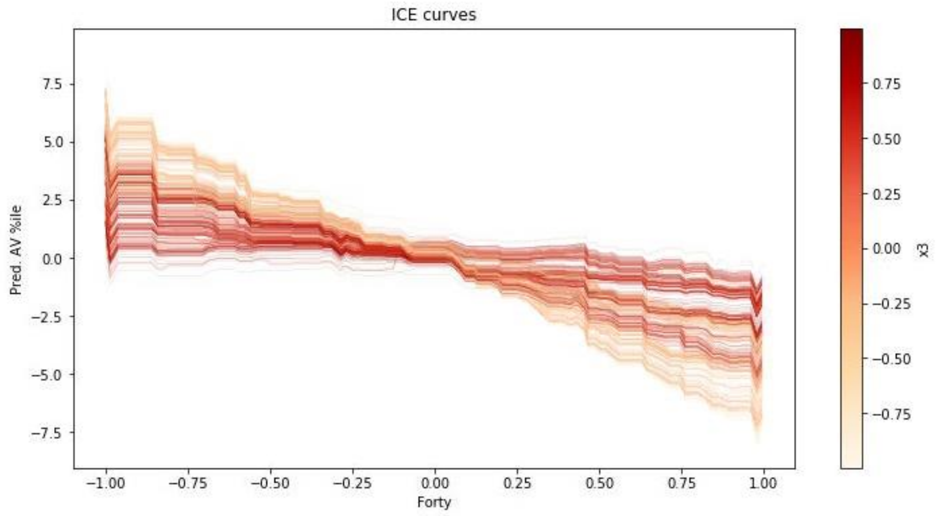
https://towardsdatascience.com/introducing-pdpbox-2aa820afd312

- The y axis is interpreted as change in the prediction from what it would be predicted at the baseline or leftmost value.
- A blue shaded area indicates level of confidence



4.2.2 Pycebox

https://github.com/AustinRochford/PyCEbox



4.3 ELI5

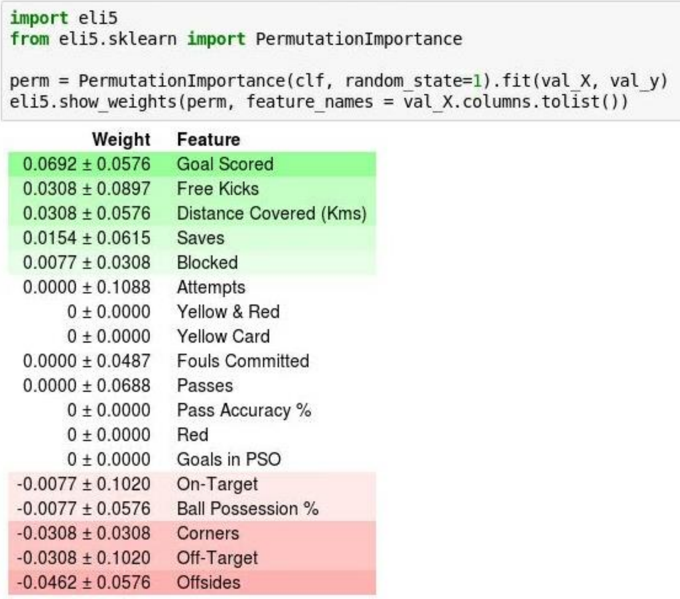
4.3.1 Permutation Importance

Permutation importances or mean decrease accuracy (MDA) is an alternative to mean decrease impurity (for randomforest) that can be applied to any model. The basic idea of permutation importance is to permute the values of each feature and measure how much that permutation negatively impacts the scoring metric.

Randomly re-ordering a single column should cause less accurate predictions, since the resulting data no longer corresponds to anything observed in the real world. Model accuracy especially suffers if we shuffle a column that the model relied on heavily for predictions.

Steps:

- Get a trained model
- Shuffle the values in a single column, make predictions using the resulting dataset. Use these predictions and the true target values to calculate how much the loss function suffered from shuffling. That performance deterioration measures the importance of the variable you just shuffled.
- Return the data to the original order (undoing the shuffle from step 2.) Now repeat step 2 with the next column in the dataset, until you have calculated the importance of each column.



```
import eli5
from eli5.sklearn import PermutationImportance

perm = PermutationImportance(clf, random_state=1).fit(val_X, val_y)
eli5.show_weights(perm, feature_names = val_X.columns.tolist())
```

Weight	Feature
0.0692 ± 0.0576	Goal Scored
0.0308 ± 0.0897	Free Kicks
0.0308 ± 0.0576	Distance Covered (Kms)
0.0154 ± 0.0615	Saves
0.0077 ± 0.0308	Blocked
0.0000 ± 0.1088	Attempts
0 ± 0.0000	Yellow & Red
0 ± 0.0000	Yellow Card
0.0000 ± 0.0487	Fouls Committed
0.0000 ± 0.0688	Passes
0 ± 0.0000	Pass Accuracy %
0 ± 0.0000	Red
0 ± 0.0000	Goals in PSO
-0.0077 ± 0.1020	On-Target
-0.0077 ± 0.0576	Ball Possession %
-0.0308 ± 0.0308	Corners
-0.0308 ± 0.1020	Off-Target
-0.0462 ± 0.0576	Offsides

The values towards the top are the most important features, and those towards the bottom matter least. The first number in each row shows how much model performance decreased with a random shuffling (in this case, using "accuracy" as the performance metric). Like most things in data science, there is some randomness to the exact performance change from a shuffling a column. We measure the amount of randomness in our permutation importance calculation by repeating the process with multiple shuffles. The number after the ± measures how performance varied from one-reshuffling to the next.

You'll occasionally see negative values for permutation importances. In those cases, the predictions on the shuffled (or noisy) data happened to be more accurate than the real data. This happens when the feature didn't matter (should have had an importance close to 0), but random chance caused the predictions on shuffled data to be more accurate. This is more common with small datasets, like the one in this example, because there is more room for luck/chance.

In our example, the most important feature was Goals scored. That seems sensible. Soccer fans may have some intuition about whether the orderings of other variables are surprising or not.

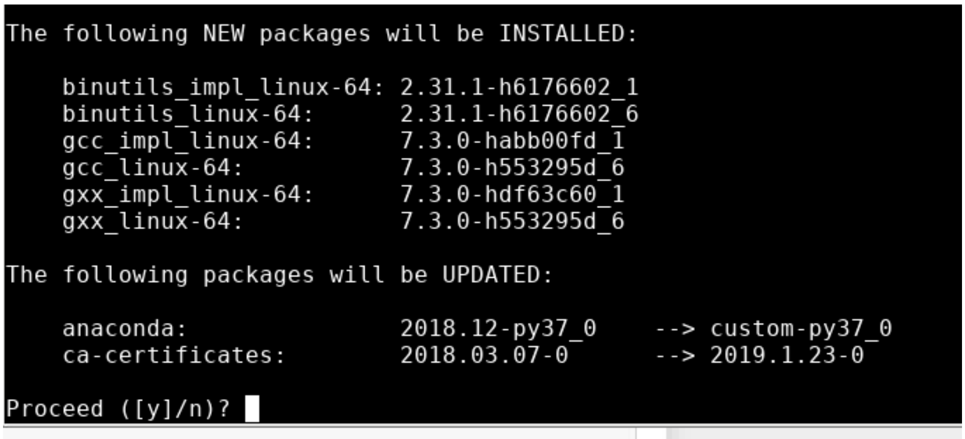
4.4 Skater

4.4.1 Install from conda/env py37

```
Issue with tensorflow install

ImportError: /lib64/libm.so.6: version `GLIBC_2.23' not found (required by
/home/l0461064@azure.cloud.corp.local/anaconda3/lib/python3.7/site-
packages/tensorflow/python/_pywrap_tensorflow_internal.so)

conda install gxx_linux-64
```



```
The following NEW packages will be INSTALLED:

    binutils_impl_linux-64: 2.31.1-h6176602_1
    binutils_linux-64:      2.31.1-h6176602_6
    gcc_impl_linux-64:      7.3.0-habb00fd_1
    gcc_linux-64:           7.3.0-h553295d_6
    gxx_impl_linux-64:      7.3.0-hdf63c60_1
    gxx_linux-64:           7.3.0-h553295d_6

The following packages will be UPDATED:

    anaconda:               2018.12-py37_0    --> custom-py37_0
    ca-certificates:        2018.03.07-0      --> 2019.1.23-0

Proceed ([y]/n)?
```

```
Y ➔ sort of overriding the conda env 3.7

TEST install gcc

NO FIX probably due to version of centos too old

conda install -c conda-forge Skater
```

pip install

NOT sudo because goes to 2.7!!

4.5 SHAP

The combination of a solid theoretical justification and a fast practical algorithm makes SHAP values a powerful tool for confidently interpreting tree models such as XGBoost's gradient boosting machines.

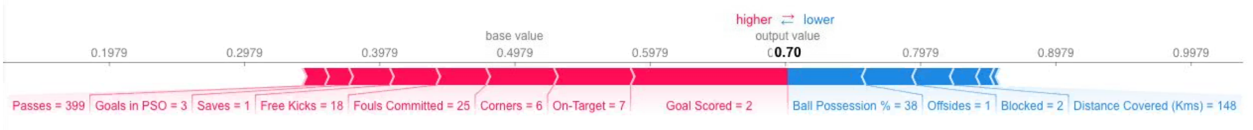
https://github.com/slundberg/shap

- Tree ensemble example with shap.TreeExplainer(XGBoost/LightGBM/CatBoost/scikit-learn models) –FAST
- Deep learning: shap.DeepExplainer() – "Slow"
- Any model: shap.KernelExplainer() – "slow"

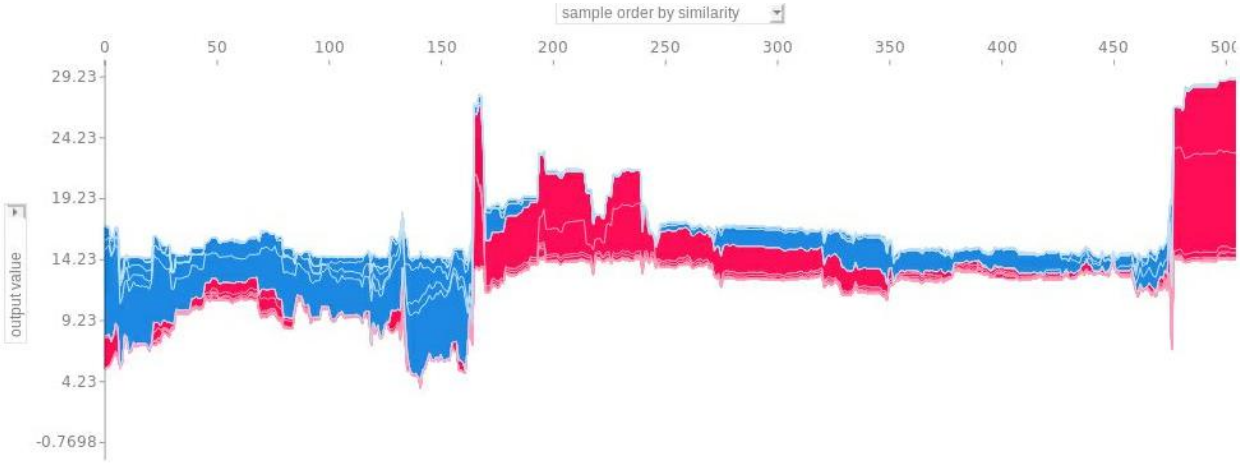
4.5.1 TreeExplainer

4.5.1.1 Shap Values Review

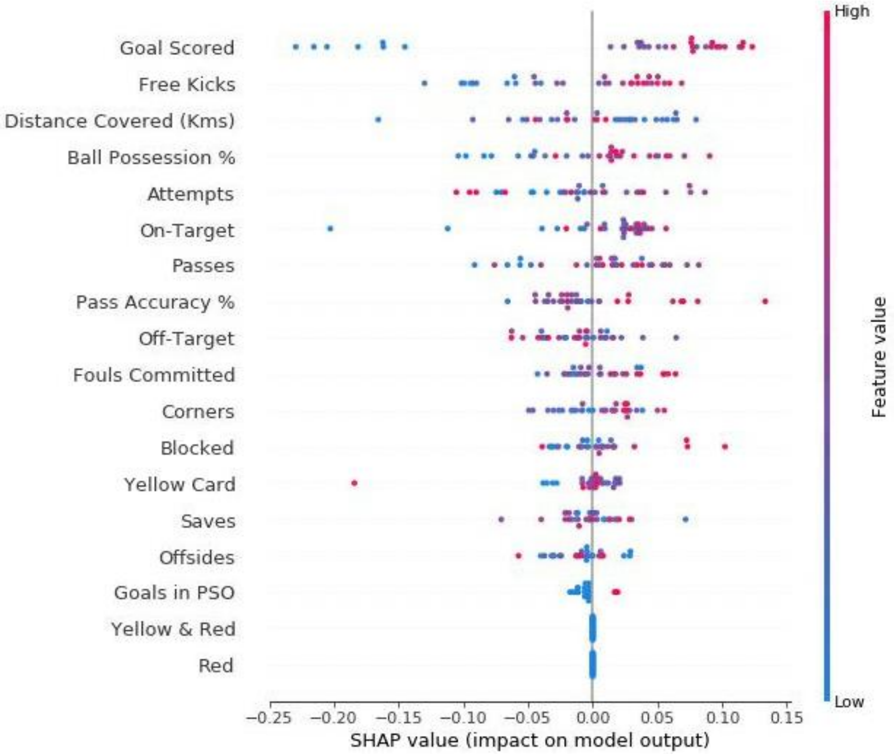
Single instance:



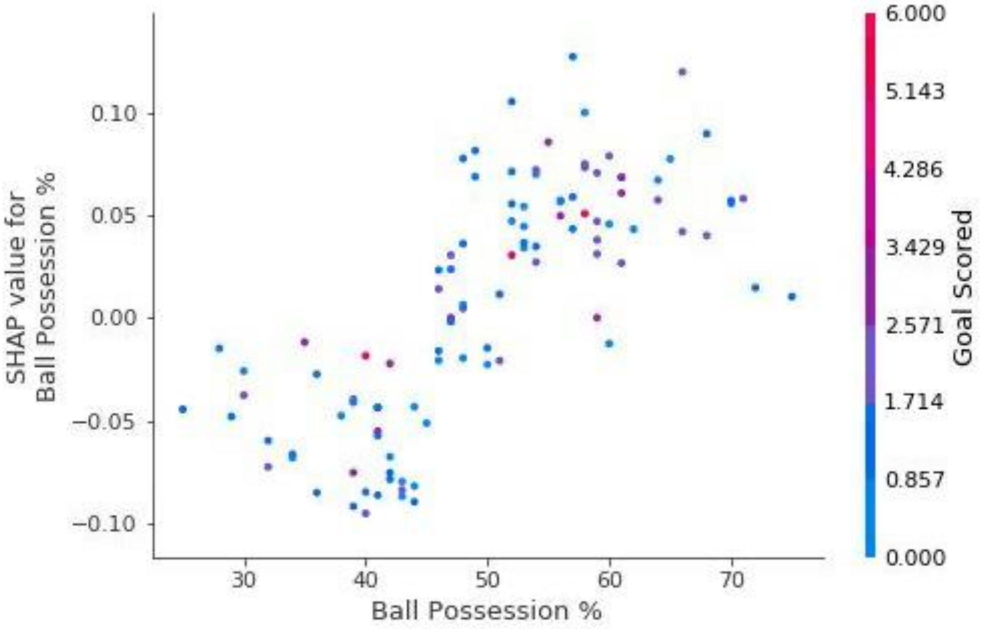
Full dataset: (interactive plot)



4.5.1.2 Summary Plots



4.5.1.3 Shap Dependence Contribution Plots



Issue with tensorflow install

```
pip install --upgrade --force-reinstall
```