Quiz 8 (Interpretability)

Solutions

1. Q1:

   Technique: Anchors
   Description: A newer approach from the inventors of LIME that generates high-precision sets of plain-language rules to describe a machine learning model prediction in terms of the model's input variable values.
   Suggested usage: Anchors is currently most applicable to classification problems in both traditional data mining and pattern-recognition domains. Anchors can be higher precision than LIME and generates rules about the most important variables for a prediction, so it can be a potential replacement for Shapley values for models that don't yet support the efficient calculation of Shapley values.
   Reference: Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, "Anchors: High-Precision Model-Agnostic Explanations," The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18), April 25, 2018,
   Global or local scope: Local.
   Best-suited complexity: Low to medium. Anchors can create explanations for very complex functions, but the rule set needed to describe the prediction can become large.
   Model specific or model agnostic: Model agnostic.

   Technique: LOCO variable importance
   Description: LOCO, or even LOFO, variously stands for leave-one-"column" or "covariate" or "feature"-out. LOCO creates local interpretations for each row in a training or unlabeled score set by scoring the row of data once and then again for each input variable (e.g., column, covariate, feature) in the row. In each additional scoring run, one input variable is set to missing, zero, its mean value, or another appropriate value for leaving it out of the prediction. The input variable with the largest absolute impact on the prediction for that row is taken to be the most important variable for that row's prediction. Variables can also be ranked by their impact on the prediction on a per-row basis.
   Suggested usage: You can use LOCO to build reason codes for each row of data on which nearly any complex model makes a prediction. LOCO can deteriorate in accuracy when complex nonlinear dependencies exist in a model. Shapley explanations might be a better technique in this case, but LOCO is model agnostic and has speed advantages over Shapley both in training and scoring new data.
   Reference: Jing Lei et al., "Distribution-Free Predictive Inference for Regression," arXiv:1604.04173, 2016,
   Global or local scope: Local but can be aggregated to create global explanations.
   Best-suited complexity: Any. LOCO measures are most useful for nonlinear, nonmonotonic

response functions but can be applied to many types of machine-learned response functions.
Model specific or model agnostic: Model agnostic.

Technique: LIME
Description: Typically uses local linear surrogate models to explain regions in a complex machinelearned response function around an observation of interest.
Suggested usage: Local linear model parameters can be used to describe the average behavior of a complex machine-learned response function around an observation of interest and to construct reason codes. LIME is approximate, but has the distinct advantage of being able to generate sparse, or simplified, explanations using only the most important local variables. Appropriate for pattern recognition applications as well. The original LIME implementation may sometimes be inappropriate for generating explanations in real-time on unseen data."
Reference: Ribeiro et al., "'Why Should I Trust You?' Explaining the Predictions of Any Classifier."
Best-suited complexity: Low to medium. Suited for response functions of high complexity but can fail in regions of extreme nonlinearity or high-degree interactions.
Global or local scope: Local.
Model specific or model agnostic: Model agnostic.

Technique: Treeinterpreter
Description: For each variable used in a model, treeinterpreter decomposes some decision tree, random forest, and GBM predictions into bias (overall training data average) and component terms. Treeinterpreter simply outputs a list of the bias and individual variable contributions globally and for each record.
Suggested usage: You can use treeinterpreter to interpret some complex tree-based models, and to create reason codes for each prediction. If you would like to use treeinterpreter, make sure your modeling library is fully supported by treeinterpreter. In some cases, treeinterpreter may not be locally accurate (local contributions do not sum to the model prediction) and treeinterpreter does not consider how contributions of many variables affect one another as carefully as the Shapley approach. However, treeinterpreter can generate explanations quickly. Also most treeinterpreter techniques appear as Python packages.
Reference: Ando Saabas, "Random Forest Interpretation with scikit-learn," Diving into Data [blog], August 12, 2015
Global or local scope: Local but can be aggregated to create global explanations.
Best-suited complexity: Any. Treeinterpreter is meant to explain the usually nonlinear, nonmonotonic response functions created by certain decision tree, random forest, and GBM algorithms.
Model specific or model agnostic: Treeinterpreter is model specific to algorithms based on decision trees

Technique: Shapley explanations
Description: Shapley explanations are a Nobel-laureate technique with credible theoretical support from economics and game theory. Shapley explanations unify approaches such as LIME, LOCO, and treeinterpreter to derive consistent local variable contributions to black-box model predictions. Shapley also creates consistent, accurate global variable importance measures.
Suggested usage: Shapley explanations are accurate, local contributions of input variables and can be rank-ordered to generate reason codes. Shapley explanations have long-standing theoretical support, which might make them more suitable for use in regulated industries, but they

can be time consuming to calculate, especially outside of decision trees in H2O.ai, LightGBM, and XGBoost where Shapley is supported in low-level code and uses the efficient Tree SHAP approach.
Reference: Lundberg and Lee, "A Unified Approach to Interpreting Model Predictions."
Global or local scope: Local but can be aggregated to create global explanations.
Best-suited complexity: Low to medium. This method applies to any machine learning model, including nonlinear and nonmonotonic models, but can be extremely slow for large numbers of variables or deep trees.
Model specific or model agnostic: Can be both. Uses a variant of LIME for modelagnostic explanations. Takes advantage of tree structures for decision tree models and is recommended for tree-based models

2. Q2

$$\phi_j(\hat{f}) = \beta_j x_j - E(\beta_j X_j) = \beta_j x_j - \beta_j E(X_j)$$

$$\begin{aligned}\sum_{j=1}^{p} \phi_j(\hat{f}) =& \sum_{j=1}^{p}(\beta_j x_j - E(\beta_j X_j)) \\ =&(\beta_0 + \sum_{j=1}^{p} \beta_j x_j) - (\beta_0 + \sum_{j=1}^{p} E(\beta_j X_j)) \\ =&\hat{f}(x) - E(\hat{f}(X))\end{aligned}$$

So, the evaluation for four features is:

$$val_x(S) = val_x(\{x_1, x_3\}) = \int_{\mathbb{R}} \int_{\mathbb{R}} \hat{f}(x_1, X_2, x_3, X_4) d\mathbb{P}_{X_2 X_4} - E_X(\hat{f}(X))$$

Efficiency:
The feature contributions must add up to the difference of prediction for x and the average.

$$\textstyle\sum_{j=1}^{p} \phi_j = \hat{f}(x) - E_X(\hat{f}(X))$$

Symmetry:
The contributions of two feature values j and k should be the same if they contribute equally to all possible coalitions. If

$$val(S \cup \{x_j\}) = val(S \cup \{x_k\})$$

for all

$$S \subseteq \{x_1, \ldots, x_p\} \setminus \{x_j, x_k\}$$

then, $\phi_j = \phi_k$

Dummy
A feature j that does not change the predicted value – regardless of which coalition of feature values it is added to – should have a Shapley value of 0. If

$$val(S \cup \{x_j\}) = val(S)$$

for all

$$S \subseteq \{x_1, \ldots, x_p\}$$

then $\phi_j = 0$

Additivity:
For a game with combined payouts $val + val^+$ the respective Shapley values are as follows:

$$\phi_j + \phi_j^+$$

Suppose you trained a random forest, which means that the prediction is an average of many decision trees. The Additivity property guarantees that for a feature value, you can calculate the Shapley value for each tree individually, average them, and get the Shapley value for the feature value for the random forest.