Deep insights into interpretability of machine learning algorithms and applications to risk management

Jie Chen, Ph.D.
MD, Corporate Model Risk, Wells Fargo

Mar 27, 2019
AI/ML in Banking and Regulation

Rapid adoption of Artificial Intelligence (AI)/Machine Learning (ML) across Financial Institutions to improve business decisions, customer experiences and risk management

Traditionally applications of statistical techniques

- Improving performance and speed
- Benchmark/comparison
- Example: Credit decision, fraud detection, etc.

Other applications have not been touched traditionally by statistical techniques

- Natural Language Processing
- Example: Chat bots, complaint analysis, customer assistance, etc.

AI/ML techniques perform quantitative estimation (with inherent output uncertainty); thus, they are “models” according to SR11-7/OCC11-12
Examples of AI/ML Applications

Credit Risk
- Credit Scoring
- Stress Testing

Capital Market
- Derivative Pricing
- Value at Risk (VaR)

Financial Crimes
- Fraud
- Anti-Money Laundering

- Supervised ML: Random Forest (RF), Gradient Boosting Machine (GBM), Neural Nets (FNN, CNN, LSTM)
- Reinforcement Learning with pay off function
- Backward Stochastic Differential Equation Neural Network (BSDENet)
- Generative Adversarial Network (GAN)

- Supervised ML: Random Forest (RF), Gradient Boosting Machine (GBM), Neural Nets
- Unsupervised Learning/Single Label: Generative Adversarial Network
Model Risk in Machine Learning

- **Data bias and limitation**
  - Understanding and managing data bias and limitation

- **Conceptual Soundness**
  - Sound modeling framework: “trustworthiness” of ML models through its “explainability”
  - Transparency of input variables and their effects to outputs/decision

- **Model Replicability and Stability**
  - Independent replication of models and stability of variable impacts

- **Performance Monitoring**
  - On-going performance monitoring and managing model changes due to model retraining

- **Implementation Control**
  - Fail-safe mode for autonomous model use—how to detect and manage when models are gone wrong during usage
Explainable Machine Learning

**Trusted Model:**
- Do we know what the model does?
- Can we trust the model?

**Model explainability** is a key requirement for banking application
- Regulatory Requirement
- Managing Model Risk
  - Financial, Reputational, Regulatory, and Legal Risks

Can we explain ML models like traditional statistical models?
# Model Explainability Approaches

<table>
<thead>
<tr>
<th>Global Diagnostics: Effects of Inputs to Outputs Impact of correlations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Derivative-Based Approach and Variance Analysis</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Local diagnostics and Model Distillation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locally Interpretable Model</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Structured-Interpretable Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explainable Neural Networks</td>
</tr>
</tbody>
</table>
Machine Learning Interpretability: Framework and Toolbox developed by AToM, CMoR, Wells Fargo
A real data example—home lending case

- This dataset is based on a retired home lending residential mortgage model.
- We used a randomly selected subset of 1 million observations, divided into training, validation and testing sets.
- Response is an indicator variable indicating if the loan is in trouble; there are 7 raw explanatory variables listed in the table below.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>fico0</td>
<td>fico at snapshot</td>
</tr>
<tr>
<td>ltv_fcast</td>
<td>ltv forecasted</td>
</tr>
<tr>
<td>dlq_new</td>
<td>delinquency status, 1 if clean and 0 otherwise</td>
</tr>
<tr>
<td>unemprt</td>
<td>unemployment rate</td>
</tr>
<tr>
<td>totpersincyy</td>
<td>total personal income year to year ratio</td>
</tr>
<tr>
<td>h</td>
<td>horizon 1, 2, ..., 9 quarters</td>
</tr>
<tr>
<td>premod_ind</td>
<td>indicator before recession Q2 2007</td>
</tr>
</tbody>
</table>
Derivative based diagnostic tools
Global and local diagnostics

• Global interpretation is aimed at interpreting the overall relationship between input and output over the entire space.

• Local interpretation is aimed at interpreting the relationship between input and output over local region, with the idea that
  – a simple parametric model may be used to approximate the input-output relationship
  – local variable importance and input-output relationships are easily interpretable from the simple local model.
Partial dependence plots (PDP) are used to visualize the input-output relationship, proposed in Friedman (2001).

It removes the effect of other variables by the marginal integration so you get the “partial effect” of a variable.

One-dimensional partial dependence plots show possible nonlinearity.

Two-dimensional partial dependence plots and related H-statistics measurement can be used to check two-way interaction effects.

Drawback: extrapolation when covariates are correlated, the grid points for PDP calculation could be far from the data distribution, so the prediction involves extrapolation.

The original plot is from “Apley, D. W. (2016). Visualizing the effects of predictor variables in black box supervised learning models”.
1D -PDP for home lending case

- We can see some nonlinear trend for snapshot FICO, unemployment rate, prediction horizon h, and ltv_fcast.
Marginal plots and ALE plots

• Marginal plots or Bivariate plots
  – For exploratory analysis, people often plot response variable against each covariate to understand pairwise input-output relationship.
  – Techniques such as binning, LOESS and regression splines for nonparametric regression can be used for the empirical estimation.
  – In mathematical formulation, it removes the effect of other variables by the conditional integration to get the marginal effect of a targeted covariate.
  – Problem: for correlated data, response is projected to a single dimension of input space, so reflects the effects of both targeted covariate and its correlated variables.

• Accumulated Local Effects (ALE) Plots
  – The ALE plots, proposed by Apley (2016) eliminate the drawbacks for partial dependent plots and marginal plots.
  – Partial derivative: to remove the effects of correlated variables (in additive models).
  – Conditional expectation: to avoid extrapolation issue in estimation.
Unified derivative based framework

• Marginal plot decomposition
  – We propose a new interpretation technique called the accumulated total derivative effects (ATDEV) plot which is based on the total derivative of the fitted response surface.
  – ATDEV plots can be proved to be equivalent with marginal plots, up to a constant difference.
  – ATDEV plots (or equivalently marginal plots) can be decomposed into Accumulated Local Effect function (ALE) and Accumulated Cross Effects (ACE).
  – The ATDEV decomposition is consistent with the sensitivity analysis in the econometric perspectives. ATDEV plots capture the total sensitivity of the response to a specific covariate.
  – 1D-ALE represents a variable’s direct 1st order effect through its own partial derivatives; 1D-ACE represents a variable’s indirect 1st order effect through the partial derivatives of its correlated variables.

• A unified derivative based framework
  – The derivative-based approach leads to a unified framework for the PDP, ALE and Marginal functions (ATDEV plots). The three are equivalent in the independent cases, but different in the correlated cases.

\[
\frac{df(x)}{dx_j} = \frac{df(x)}{\partial x_j} + \sum_{k \neq j} \frac{df(x)}{\partial x_k} \frac{dm_k(x_j)}{dx_j}
\]

When unemployment rate is bumped by 1%, how will PD be impacted

Unemployment bump correlates with the bumps of other variables (e.g., loan characteristic variables) resulting in the change on PD in the fitted model

Direct impact on PD due to unemployment bump
How to obtain derivatives for ML algorithms

- Even when the prediction model itself does not have closed-form gradients (such as Gradient Boosting), one can fit a NN surrogate model to the prediction model scores, and get model performance comparable to the original prediction models.

- The concept of surrogate models is known as emulators in the field of computer experiments (Bastos & O'Hagan, 2009), and is referred to as model distillation (Tan, et al., 2018) or model compression (Bucilua, et al., 2006) in machine learning literature, with “born again trees” (Breiman & Shang, 1997) as one of the earliest implementations.
The diagonal plots are ALE showing the “direct” 1D effect of each variable on the response surface.

The off-diagonal plots are ACE, i.e., subplot \((k,j)\) shows the “indirect” 1D effect of each variable \(x_j\) passed through its correlated variables \(x_k\) onto the response surface.

The sum of the column is marginal plot (or ATDEV plot).

A part of sensitivity of unemployment is taken by premod_ind and LTV_forecast.
ATDEV variance heat map and correlation matrix

- ATDEV Variance heat map combines the dependence (correlations) among predictors and their influence on the response and is more useful in a regression context, while ordinary correlation matrix is unsupervised.
  - the diagonal cells represent the individual marginal contribution of each predictor on the response (i.e., ALE);
  - the off-diagonal cells represent the magnitude of the cross marginal effect of the column variable on response transferred through the row variable (i.e., ACE).
Local diagnostics and model distillation: Locally Interpretable Models and Effects based on Supervised Partitioning (LIME-SUP)
Local importance

• Describe how individual observation’s attributes affect model prediction for that observation.

• Important for providing reason codes for credit decisions

• Approaches
  – LIME (Local Interpretable Model-Agnostic Explanations)
  – KLIME
  – LIME-SUP
  – LOCO (leave one covariate out)
  – SHAP explanation
  – Tree interpreter
  – Quantitative input influence (QII)
  – Integrated gradients
  – DeepLIFT
  – Layer-wise Relevance Propagation (LRP)
  – Derivative based sensitivity analysis
Model distillation

• Model distillation:
  – Model distillation was originally designed to distill knowledge from a large, complex teacher model to a faster, simpler student model without significant loss in prediction accuracy.
  – We investigate model distillation for another goal—transparency, e.g., investigating if fully-connected neural networks can be distilled into models that are transparent or interpretable.
  – The purpose is to approximate the predications of the underlying model as closely as possible while retaining interpretability.
  – Literature: emulators, surrogate models, model distillation, model compression, “born again trees”

• KLIME is a variant of LIME proposed in H2o Driverless AI. It divides the input space into regions by clustering and fit a linear model in each region.
  – Unstable, unsupervised, partitions according to the Voronoi diagrams
LIME-SUP

• Locally Interpretable Models and Effects based on Supervised Partitioning (LIME-SUP) is a local interpretation method developed by CMoR. It is a supervised partitioning method using information from the machine learning model.

• The goal is to use supervised partitioning to achieve a more stable, more accurate and more interpretable surrogate model than KLIME.

• There are two implementations of LIME-SUP. One uses model based tree (LIME-SUP-R) and the other uses partial derivatives (LIME-SUP-D).
LIME-SUP-R algorithm

1. Let \( \{X_{1i}, \ldots, X_{Ki}, i = 1, \ldots, N \} \) be the set of \( K \) predictor variables used to train the original ML algorithm. We will use them as both modeling variable and partitioning variable for illustration purpose.

2. For the specified class of parametric model (say linear regression model with no interactions), fit a model-based tree to the ML predictions obtained on the training dataset.
   1) Fit an overall parametric model at the root node to the ML predictions and modeling variables.
   2) Find the best split to partition the root node into two child nodes. This is done by (again) fitting the same class of parametric models to all possible pairs of child nodes and determining the “best” partition. This involves searching over all partitioning variables and possible splits within each variable and optimizing a specified fit criterion such as MSE or logloss.
   3) Continue splitting until a specified stop criterion is met; for example, max depth or minimum number of observations in the child node is reached, or the fit is satisfactory.

3. Prune back the tree using appropriate model fit statistics such as improvement in \( R^2 \), improvement in SSE, etc. on the validation dataset, to cut off splits that have small impact.

4. Once the final tree is determined, use a regularized regression algorithm (such as LASSO) to fit a sparse version of the parametric model at each node.

5. Assess the fit on the testing dataset.
Notes

Some notes:

• The search for best split for model based tree is very different from a regular decision tree. It requires fitting linear models in the child nodes instead of a constant, and doing so for each possible split point of each splitting variable. This is much more computationally expensive than fitting a regular decision tree.

• To reduce the amount of computation, M-Fluctuation test in Zeileis et. al (2008) is used as a fast way to screen the partitioning variables.
  – M-Fluctuation test is a test for parameter instability. Its purpose is to test if the coefficients of a parametric model will change according to different segments of certain variable;
  – If the test is insignificant, a global model will fit well; otherwise, it is necessary to divide the data according to that variable and fit different child models.

• We rank the partitioning variables by the M-Fluctuation test result, and for the top variables, we do an exhaustive search over the combination of splitting variable and splitting point. This greatly reduces the amount of computation.
A real data example-home lending case

- Figures shows the tree structure and the coefficients in the terminal nodes.

- The strongest patterns in the coefficients exist for ltv_fcast, fico0 and delinquency status.

- For example, the highest coefficients of ltv_fcast at node 11 and 13 indicating the steepest slope for ltv_fast in [61.4, 92.5], and flatter at two ends.
A real data example-home lending case

• Similarly we fit KLIME with 8 clusters.
• Table below shows the MSE, Rsquare and AUC for the 5 methods.
• LIME-SUP is better than KLIME. Besides that, we see LIME-SUP-R fits slightly better than LIME-SUP-D, which is expected.

<table>
<thead>
<tr>
<th></th>
<th>LIME-SUP-R</th>
<th>LIME-SUP-D</th>
<th>KLIME-E</th>
<th>KLIME-M</th>
<th>KLIME-P</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td>0.0419</td>
<td>0.0485</td>
<td>0.0662</td>
<td>0.0677</td>
<td>0.0648</td>
</tr>
<tr>
<td>$R^2$</td>
<td>0.975</td>
<td>0.970</td>
<td>0.960</td>
<td>0.959</td>
<td>0.960</td>
</tr>
<tr>
<td>AUC</td>
<td>0.817</td>
<td>0.817</td>
<td>0.816</td>
<td>0.816</td>
<td>0.816</td>
</tr>
</tbody>
</table>
A real data example—home lending case

- Figure below provides a different view of the comparisons: values of MSE and $R^2$ computed within each of eight local regions.

- The conclusions are similar as before. LIME-SUP does better almost on all local regions, except LIME-SUP-D has slightly larger MSE than KLIME occasionally.
Structured interpretable Model
--explainable neural network
Linear Models: Regression and Splines

Linear Model

\[ f(x) = w_0 + w x \]

If \( f(x) \) nonlinear, we can still employ linear model: Spline

\[ f(x) = w_0 + \sum_{j=1}^{k} w_j B_j(x) \]

Where \( B_j(.) \) are basis functions such as constant or simple hinge function \( \max(0, x-c_j) \)

\( c_j \): knot locations

\( k \): #knots
Neural Networks

\[ f(x) = w_0 + \sum_{j=1}^{k} w_j B_j(v_j x) \]

\( B_j(.) \) with simple hinge functions are called ReLU (Rectifier Linear Units), \( \max(0, v_j x - c_j) \)

\( c_j \) "knot locations" are called "bias weights"

Unlike spline approach where the knot locations, \( c_j \), are pre-determined, all the weights in Neural Networks are optimized through gradient descent (backpropagation) algorithm

- Knot locations are optimized simultaneously among all input variables
- Unlike linear model in the traditional statistical approach, the knot location is optimized on scaled \( x \), the \( (v_j x) \), instead of \( x \)
Dealing with More Complex Functions

Simple “hinged” basis function requires a lot of knots to approximate more complicated functions

• Statistical approach
  • Higher order splines or other more complicated non parametric approach including ridge function estimation, $B_j(.)$: Projection Pursuit Regression/Additive Index Model

• Neural network approach:
  • Splines on splines: additional hidden layer (narrow deep network instead of wide shallow network)
Higher Dimension: Projection Approach

Projection Splines and single hidden layer Neural Networks have the similar form:

\[ f(x) = w_0 + \sum_{j=1}^{k} w_j B_j(v_j^T x) \]

Additive ‘Index’ Model:

\[ f(x) = \gamma_1 h_1(\beta_1^T x) + \gamma_2 h_2(\beta_2^T x) + \ldots + \gamma_k h_k(\beta_k^T x) \]

Structured NN | Model Structure
--- | ---
AIM-Net(xNN) | AIM \[ y = \sum_{j=1}^{M} h_j(\alpha_j^T x) + \epsilon \]
GAM-Net | GAM \[ g(y) = \sum_{j=1}^{p} f_j(x_j) + \epsilon \]
PLM-Net | PLM \[ y = \beta^T x + h(z) + \epsilon \]
SIM-Net | SIM \[ y = h(\alpha^T x) + \epsilon \]
LIN-Net | LM \[ y = \beta^T x + \epsilon \]
Additive Index Models

- Formulation:
  \[ f(x) = h_1(\alpha_1'x) + \ldots + h_M(\alpha_M'x) \]
  
  - Ridge functions \( \{h_j, j = 1, \ldots, M\}; \text{ Usually } M < p \)
  
  - Projection coefficients (directions) \( \alpha'_j, j = 1, \ldots, M \)
  
  - Ridge functions are additive

- Model is inherently **interpretable**:  
  
  - Linear projections are understandable  
  
  - (Nonlinear) Ridge functions are easily graphed

- Has a similar “universal approximation” theorem

- Familiar model from Projection Pursuit Regression

- Some potential identifiability concerns:
  
  - As formulated, may not be identifiable  
  
  - Conditions exist (Yuan, 2011) but some may not be reasonable or applicable in credit modelling context
Explainable Neural Network (xNN)

- Structured network architecture designed to learn an AIM.
- Network structures chosen to match features of AIM:
- Two Key Structures:
  - **Projection Nodes**: Nodes with linear activation functions. Used for projections and final sums.
  - **Subnetwork**: Collection of nodes that:
    - Internally, are fully connected, multi-layer, and use nonlinear activation functions.
    - Externally, are only connected to rest of the network through a univariate input and univariate output.
    - Used to learn ridge functions, $h_i(\cdot)$
$y = 0.5x_1 + 0.5x^2_2 + 0.5x_3x_4 + 0.3x^2_5 + \epsilon$

**Learned Ridge Functions:**

- Subnet 1:
  - $y_1$
  - $y_2$
  - $y_3$
  - $y_4$

- Subnet 2:
  - $y_1$
  - $y_2$
  - $y_3$
  - $y_4$

**Learned Projection Coefs:**

- Coefs for net1:
  - $\beta_1$
  - $\beta_2$
  - $\beta_3$
  - $\beta_4$

- Coefs for net2:
  - $\beta_1$
  - $\beta_2$
  - $\beta_3$
  - $\beta_4$

- Coefs for net3:
  - $\beta_1$
  - $\beta_2$
  - $\beta_3$
  - $\beta_4$

- Coefs for net4:
  - $\beta_1$
  - $\beta_2$
  - $\beta_3$
  - $\beta_4$