Correlation and Causation Analysis in Black Box Deep Learning Models

A Mathematical Framework

Bharath Keshavamurthy and Imran Pasha

CISCO Systems, Inc.

June 2019

1 System Model

- $M \triangleq$ The number of training examples, $M \in \mathbb{Z}_{++}$
- f is a trained Deep Neural Network based classifier defined as $f : \mathcal{X} \rightarrow \mathcal{Y}$
- $\mathcal{X}$ is the input feature space, $\mathcal{X} \in \mathbb{R}^K$
- $\mathcal{Y}$ is the output label space, $\mathcal{Y} \in \{0,\ 1\}$
- The Deep Neural Network based classifier is a black box to this prediction rationale determination operation and hence, we do not care about the hyper-parameters of the Neural Network.
- Let $(\vec{x} \in \mathcal{X},\ y \in \mathcal{Y})$ be a prediction instance that needs to be explained by the proposed rationale determination engine.
- $\kappa \leq K \ \triangleq$ The number of interpretable features in the prediction rationale
- Let $\vec{z} \in \mathcal{Z}$ be a perturbed instance sampled from the non-zero components (categorical or numerical) of $\vec{x} \in (\vec{x} \in \mathcal{X},\ y \in \mathcal{Y})$ where $\mathcal{Z}$ represents the sparse, interpretable feature space.
- $N \triangleq$ The number of perturbed instances sampled for prediction rationale determination, $N < M$
- $Z \equiv \{\vec{z}_i \in \mathcal{Z} | i \in \{1, 2, \ldots, N\}\}$ is the set of all perturbed instances sampled uniformly at random from the non-zero components of $\vec{x} \in (\vec{x} \in \mathcal{X},\ y \in \mathcal{Y})$

2 State-of-the-art

- A. Fisher, et. al., "All Models are Wrong but Many are Useful: Variable Importance for Black-Box, Proprietary, or Misspecified Prediction Models, using Model Class Reliance", November 2018
- S. Lundberg and S. Lee, "A Unified Approach to Interpreting Model Predictions", November 2017
- M. Ribeiro, et. al., ""Why Should I Trust You?" Explaining the Predictions of Any Classifier", 2016

3 Rationale Engine Design

The intuition is to fit a locally interpretable regression model in the neighbourhood of the sample prediction instance and use this model to explain the rationale behind that specific prediction. We can fit a Linear Model or build a Random Forests based Model from the sampled and perturbed set of interpretable feature vectors - in this paper, we stick to a linear model.

3.1 The Sparse Linear Locally Interpretable Model

Here are a few design points relevant to the linear model fit in the vicinity of the sample prediction instance.

- Model: $g(\vec{z}) = \vec{\theta}^{\intercal}\vec{z} + \phi$
- Exponential Kernel Weights:

$$e_{\vec{x}}(\vec{z}) = e^{\frac{D'(\vec{x},\vec{z})^2}{\sigma^2}}, \tag{1}$$

where

$$D'(\vec{x},\vec{z}) \triangleq \text{Cosine Similarity} = \Big(\frac{\vec{x}\cdot\vec{z}}{||\vec{x}||_2||\vec{z}||_2}\Big), \tag{2}$$

$$\sigma^2 \triangleq \text{The width of the kernel}$$

- Cost function:

$$f_0(\vec{\theta}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \vec{\theta}^{\intercal}\vec{z} - \phi)^2 \tag{3}$$

- The regularization condition modelled as an inequality constraint:

$$\sum_{k=1}^{\kappa}|\theta_k| \leq \alpha, \tag{4}$$

where, $\alpha > 0$ is a regularization parameter.

3.2 Problem Formulation

$$\vec{\theta}^*, \phi^* = \underset{\vec{\theta},\phi}{\operatorname{argmin}} \ \Big\{ \frac{1}{N} \sum_{i=1}^{N} (y_i - \vec{\theta}^\intercal \vec{z} - \phi)^2 \Big\}, \quad \text{such that, } ||\vec{\theta}||_1 \leq \alpha \tag{5}$$

Accounting for the bias term within the target and the prediction, let's rewrite the objective function as,

$$\vec{\theta}^* = \underset{\vec{\theta}}{\operatorname{argmin}} \ \Big\{ \frac{1}{N} \sum_{i=1}^{N} ((y_i - \bar{y}) - (\vec{\theta}^\intercal \vec{z} - \vec{\theta}^\intercal \bar{\mathbf{z}}))^2 \Big\}. \tag{6}$$

Assuming the targets and the predictions are normalized appropriately, we can write equation (6) as,

$$\vec{\theta}^* = \underset{\vec{\theta}}{\operatorname{argmin}} \ \Big\{ \frac{1}{N} ||\vec{y} - A\vec{\theta}||_2^2 \Big\}.$$

In other words,

$$\vec{\theta}^* = \underset{\vec{\theta}}{\operatorname{argmin}} \ \Big\{ \frac{1}{N} \sum_{i=1}^{N} (y_i - \vec{\theta}^\intercal \vec{z})^2 \Big\}. \tag{7}$$

Finally, let's add the exponential kernel coefficient as a neighbourhood weight to equation (7) and re-write the optimization problem as,

$$\vec{\theta}^* = \underset{\vec{\theta}}{\operatorname{argmin}} \ \Big\{ \frac{1}{N} \sum_{i=1}^{N} e_{\vec{x}}(\vec{z}_i) \ (y_i - \vec{\theta}^\intercal \vec{z})^2 \Big\}, \quad \text{such that, } ||\vec{\theta}||_1 \leq \alpha. \tag{8}$$

3.3 Problem Solution

The optimization problem detailed in equation (8) is a standard convex optimization problem due to the following reasons:

- The inequality constraint encapsulates a feasible set defined as

  $$\mathcal{F} \equiv \{\vec{\theta} \in \mathbb{R}^\kappa | \mathbf{1}^\intercal \vec{\theta} \leq \alpha\}.$$

  The feasible set $\mathcal{F}$ is a convex set because it is a half-space and all half-spaces are convex sets.

- The objective function,

  $$f_0(\vec{\theta}) = \frac{1}{N} \sum_{i=1}^{N} e_{\vec{x}}(\vec{z}_i) \ (y_i - \vec{\theta}^\intercal \vec{z})^2,$$

  is a convex function due to the following reasons:

– The domain of the function denoted by $dom(f_0) \equiv \mathbb{R}^{\kappa}$ is a convex set.

– The objective function can be re-written as follows.
Pushing the weight in, we get,

$$f_0(\vec{\theta}) = \frac{1}{N}\sum_{i=1}^{N}((e_{\vec{x}}(\vec{z}_i))^2 y_i - (e_{\vec{x}}(\vec{z}_i))^2 \vec{\theta}^{\intercal}\vec{z})^2$$

$$f_0(\vec{\theta}) = \frac{1}{N}||\vec{y}' - A'\vec{\theta}||_2^2$$

where,

$$\vec{y}' = [(e_{\vec{x}}(\vec{z}_i))^2 y_i | i \in \{1, 2, \ldots, N\}]^{\intercal}$$

$$A' = [(e_{\vec{x}}(\vec{z}_i))^2 z_{ij}],\ \forall i \in \{1, 2, \ldots, N\} \text{ and } j \in \{1, 2, \ldots, \kappa\}$$

The L2-norm operation of a vector ($h(.) = ||.||_2$) is convex because of the properties imbibed by the Triangle Inequality as explained below. Please note that this holds true for any $p - norm,\ p \in \mathbb{Z}_{++},\ p \geq 1$. For $0 \leq \beta \leq 1$ and for any $\vec{x}, \vec{y} \in dom(g)$,

$$||\beta\vec{x} + \bar{\beta}\vec{y}||_p \leq ||\beta\vec{x}||_p + ||\bar{\beta}\vec{y}||_p$$

$$||\beta\vec{x} + \bar{\beta}\vec{y}||_p \leq \beta||\vec{x}||_p + \bar{\beta}||\vec{y}||_p$$

This satisfies the Jensen's inequality requirement for convexity of functions, i.e.,

$$h(\beta\vec{x} + \bar{\beta}\vec{y}) \leq \beta h(\vec{x}) + \bar{\beta}h(\vec{y})$$

Furthermore, the norm-square operation in the objective function is convex due to the composition of functions. More details are given below. For any $\vec{x}, \vec{y} \in dom(h) \cap dom(l)$ where $h(.)$ denotes the norm function and $l(.)$ denotes the square function, we can write,

$$l(h(\beta\vec{x} + \bar{\beta}\vec{y})) \leq l(\beta h(\vec{x}) + \bar{\beta}h(\vec{y})) \leq \beta l(h(\vec{x})) + \bar{\beta}l(h(\vec{y}))$$

Finally, the affine transformation of a convex function (norm-square of a vector) is a convex function.
Therefore, the cost function minimization performed within the locally interpretable regression model is a convex optimization problem.

- Now that we have a constrained convex optimization problem on our hands, we can use Projection Gradient Descent to solve for the optimal parameters ($\vec{\theta}$) in the locally interpretable linear model.

3.4 The Algorithm

The optimization algorithm is detailed below. Note that **Algorithm 1** is just an optimization algorithm used to determine the weights of the numerous constructed locally interpretable models. The fully integrated causation analysis algorithm is detailed in **Algorithm 2**.

Algorithm 1 Projection Gradient Descent Algorithm

Initialization: Pick an initial point $\vec{\theta_0} \in \mathcal{F}$;
Pick a step-size, $\gamma = 0.1$;
while $\vec{\theta}_{t+1} != [\vec{\theta_t} - \gamma \nabla f_0(\vec{\theta}_t)]^+$ **do**
  $\vec{\theta}_{t+1} = [\vec{\theta_t} - \gamma \nabla f_0(\vec{\theta}_t)]^+$
end

Here, the $[\vec{\theta}]^+$ operation refers to the projection algorithm which is implemented in Python. In this document, we explain the **Vector Projection Logic** for a linear model in two dimensions.
The component of a vector $\vec{a} \in \mathbb{R}^2$ along another vector $\vec{b} \in \mathbb{R}^2$ is given by,

$$a_b = \frac{\vec{a} \cdot \vec{b}}{||\vec{b}||}$$

The projection of $\vec{a}$ along $\vec{b}$ is then given by,

$$\vec{a}_b = a_b \frac{\vec{b}}{||\vec{b}||}$$

Taking the collection of line segments constituting the boundary of the feasible set, we find the $<$*smallest_ distance, closest_ point*$>$ pair for the out-of-bounds point with respect to each of these line segments and then find the smallest distance among them. The point corresponding to the smallest distance among all the line segments is the projection of the out-of-bounds point in the feasible set.
The final, integrated causation determination algorithm is outlined below.

Algorithm 2 Integrated Causation Determination Algorithm

Build, Compile, and Train a Neural Networks based Classifier $f : \mathcal{X} \rightarrow \mathcal{Y}$;
Predict a sample case from the test data and call it a *prediction rationale instance* $p(\vec{x}^{(p)}, y^{(p)})$;
Now, the task at hand is to explain/interpret the prediction made by the NN-Classification engine;
Based on the hyper-parameters N=the number of perturbed samples and κ=the number of features in the interpretable model, create perturbed samples $\vec{z}_i,\ \forall i \in \{1, 2, \ldots, N\}$ by sampling κ random features of the prediction rationale instance p's feature vector $\vec{x}^{(p)}$:
samplesCollection = [];
$j = 0$;
for all possible κ-feature combinations, i.e. **while** $j < \binom{K}{\kappa}$ **do**
    $i = 0$;
    samples = [];
    while $i < N$ **do**
        Perturbed sample $\vec{z}_i$ = randomly sample $(x_1, x_2, \ldots, x_\kappa)_j$ components of the prediction rationale instance's feature vector $\vec{x}^{(p)}$;
        Construct the bare-metal perturbed sample (stripped-down version for optimization) $\vec{z}_i'$;
        Find the cosine similarity of the perturbed sample - $D'(\vec{x}^{(p)}, \vec{z}_i) = \frac{\vec{x}^{(p)} \cdot \vec{z}_i}{||\vec{x}^{(p)}||_2 ||\vec{z}_i||_2}$;
        Evaluate the weight (exponential kernel) of this perturbed sample - $e_{\vec{x}^{(p)}}(\vec{z}_i) = \exp(\frac{D'(\vec{x}^{(p)}, \vec{z}_i)^2}{\sigma^2})$;
        Determine the prediction made by the classifier for this perturbed sample - $y_{\vec{z}_i} = f(\vec{z}_i)$;
        samples$[i] = (\vec{z}_i, \vec{z}_i', e_{\vec{x}^{(p)}}(\vec{z}_i), y_{\vec{z}_i})$;
    end
    samplesCollection$[j]$ = samples;
end
evaluationResults = [];
$j = 0$;
for samples in samplesCollection, i.e. **while** $j < \binom{K}{\kappa}$ **do**
    Perform Projection Gradient Descent to determine the minimum of the loss function and the corresponding optimal weights:
    Initialization: Pick an initial point $\vec{\theta}_0 \in \mathcal{F}$;
    Pick a step-size, $\gamma = 0.1$;
    $f_0^j(\vec{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \left(e_{\vec{x}}(\vec{z}_i)\ (y_i - \vec{\theta}^\intercal \vec{z})^2 \right)$;
    while $\vec{\theta}_{t+1}^j\ != \ [\vec{\theta}_t^j - \gamma \nabla f_0^j(\vec{\theta}_t^j)]^+$ **do**
        $\vec{\theta}_{t+1}^j \ = \ [\vec{\theta}_t^j - \gamma \nabla f_0(\vec{\theta}_t^j)]^+$
    end
    Optimized Result: evaluationResults$[j] = (\vec{\theta}^{j*}, f_0(\vec{\theta}^{j*}))$;
end
Output: Causation for the prediction rationale instance - $Causation(p(\vec{x}^{(p)}, y^{(p)}))$ = min(evaluationResults, **lambda** $x :\ x[1])[0]$;