APPENDIX A GRAPH PRELIMINARY

Directed Graph (Digraph):

A Directed Graph $\mathcal{G}$ is defined by a pair (V, E), where V is a non-empty finite set of elements called **vertices** and E is a finite set of ordered pairs of distinct vertices called **arcs** or edges

Subdigraph:

A digraph $\mathcal{H} = (V_{\mathcal{H}}, E_{\mathcal{H}})$ is subdigraph of a digraph $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$ if $V_{\mathcal{H}} \subseteq V_{\mathcal{G}}$ and $E_{\mathcal{H}} \subseteq E_{\mathcal{G}}$ and every edge in $E_{\mathcal{H}}$ has both end-vertices in $V_{\mathcal{H}}$. One says $\mathcal{H}$ is **induced** by $V_{\mathcal{H}}$ and call $\mathcal{H}$ an induced subdigraph of $\mathcal{G}$ (Bang-Jensen & Gutin, 2008).

Degree of a Directed Graph

Given $v \in V$ the **indegree** of v is denoted as $d^-(v)$ which is the number of edges that points to v and the outdegree is denoted $d^+(v)$ which is the number of edges that points out from v to some other vertices. A node $v \in V$ is a source if $d^-(v) = 0$ and it is a **sink** if $d^+(v) = 0$ (Bang-Jensen & Gutin, 2008).

Weighted Directed Graph: It is a Directed Graph $\mathcal{G} = (V, E)$ with a mapping $W : E \to \mathbb{R}$ which assigns values to each edge. Hence, $\mathcal{G}$ can be shown as a triplet (V, E, W) (Bang-Jensen & Gutin, 2008).

Walk:

A walk in directed graph $\mathcal{G} = (V, E)$ is an alternating sequence $W = x_1a_1x_2a_2x_3 \dots x_{k-1}a_{k-1}x_k$ where $x_i \in V$, $1 \leq \forall i \leq k$ and $a_i \in E$ such that $a_i = (x_i, x_{i+1})$ (Bang-Jensen & Gutin, 2008).

Strongly connected components (SCC)

In a directed graph $\mathcal{G}$ vertex y is **reachable** from vertex x if there is walk from x to y. A directed graph $\mathcal{G}$ is strongly connected if for every pair of $x, y \in V$, x is reachable from y and vice versa.

A strongly connected component of an directed graph $\mathcal{G}$ is a maximal induced subgraph that is strongly connected.

Complete Graph. A directed graph $\mathcal{G} = (V, E)$ is complete, if for every pair $x, y \in V$, we have $(x, y), (y, x) \in E$ (Bang-Jensen & Gutin, 2008).

Cliques: A clique is complete subdigraph of a given graph (Meeusen & Cuyvers, 1975).

Quotient Graph $\mathcal{S}$

Given the graph $\mathcal{H} = (V, E_{\mathcal{H}})$, we denote $\mathcal{S} = (V_{\mathcal{S}}, E_{\mathcal{S}})$ as a quotient graph through strong connectivity equivalence relation, i.e., $i \sim j \iff i$ and j are strongly connected. More precisely:

Definition Given the graph $\mathcal{H} = (V, E_{\mathcal{H}})$, we denote $\mathcal{S} = (V_{\mathcal{S}}, E_{\mathcal{S}})$ as a reduced graph where:

- The set of vertices is the quotient set, i.e., $V_{\mathcal{S}} = V/\sim = \{SCC(v) : v \in V\}$
- Two equivalence classes $SCC(u), SCC(v) \in V_{\mathcal{S}}$ forms an edge if and only if $(u, v) \in E_{\mathcal{H}}$. In particular (Bloem et al., 2006):

$$E_{\mathcal{S}} = \{(C, C') \,|\, C \neq C' \text{ and } \exists v \in C, v' \in C' : (v, v') \in E_{\mathcal{H}}\} \tag{7}$$

Graph Density of Digraphs:

Graph density computes ratio of number of edges in the graph to the maximal number of edges, i.e.,

$$d = \frac{m}{n(n-1)} \tag{8}$$

where n is the number of nodes and m is the number of edges in the directed graph.

APPENDIX B PROOF OF THE THEOREM 1 AND COROLLARY 1.1.

We restate the Theorem 1:

B.1 THEOREM1

For $i, j, k \in D$, assume that

$$\max_{j \in S \subseteq D} |u(S \cup \{i\}) - u(S)| \leq \varepsilon_j \qquad (I)$$
$$\max_{i \in S \subseteq D} |u(S \cup \{k\}) - u(S)| \leq \varepsilon_i \qquad (II)$$

Then, the following inequalities hold:

$$|E^2(u)_{ij}| \leq \frac{d!}{2}\varepsilon_j, |E^2(u)_{ki}| \leq \frac{d!}{2}\varepsilon_i \qquad (A)$$

$$|E^2(u)_{kj}| \leq \frac{d!}{2}(2\varepsilon_j + \varepsilon_i) \qquad (B)$$

Proof. **Part A**)

Using Eq (6), $|E^2(u)_{ij}|$ is equal to:

$$|E^2(u)_{ij}| = |\sum_{j \in S \subseteq D \setminus \{i\}} \frac{|S|!\,(d - |D| - 1)!}{d!}(u(S \cup \{i\}) - u(S))| \leq \tag{9}$$

$$\sum_{j \in S \subseteq D \setminus \{i\}} \frac{|S|!\,(d - |S| - 1)!}{f!}|(u(S \cup \{i\}) - u(S))| \quad \text{triangular inequality} \tag{10}$$

$$\leq \sum_{j \in S \subseteq D \setminus \{i\}} \frac{|S|!\,(d - |S| - 1)!}{d!}\varepsilon_j = (\sum_{j \in S \subseteq D \setminus \{i\}} \frac{|S|!\,(d - |S| - 1)!}{d!})\varepsilon_j =^* \frac{d!}{2}\varepsilon_j \tag{11}$$

*:All the possible combinations of features are $d!$ but half of these times $j \in S$ and half of these times $j \notin S$, because given a sequence of features $a_1, \ldots, a_d$ where $j \in S$ as follows:

$$\underbrace{(a_1 \ldots j \ldots a_{|S|})}_{|S|} \quad i \quad \underbrace{(a_{|S|+2} \ldots a_d)}_{d-|S|-1} \tag{12}$$

There is an exact sequence on $j \notin S$ as follows:

$$\underbrace{(a_{|S|+2} \ldots a_d)}_{d-|S|-1} \quad i \quad \underbrace{(a_1 \ldots j \ldots a_{|S|})}_{|S|} \tag{13}$$

so it is $\frac{d!}{2}$ elements that $j \in S$, similarly one can derive the other inequality in part A which is $|E^2_{ki}| \leq \frac{d!}{2}\varepsilon_i$.

Part B)

We start by writing E^2_{kj} from Eq (6), i.e.:

$$\begin{aligned} |E^2_{kj}(u)| &= |\sum_{j \in S \subseteq D \setminus \{k\}} \frac{|S|!\,(d - |S| - 1)!}{d!}(u(S \cup \{k\}) - u(S))| \\ &\leq \sum_{j \in S \subseteq D \setminus \{k\}} \frac{|S|!\,(d - |S| - 1)!}{d!}|(u(S \cup \{k\}) - u(S))| \end{aligned} \tag{14}$$

where we used triangular inequality, now we look at the element inside the summation separately when $i \in S$ and $i \notin S$, note that in all cases $j \in S$, in particular we have:

- if $i \in S$, then from the assumption 2 we have $|u(S \cup \{k\}) - u(S)|$is less or equal than ε_i
- if $i \notin S$: In this case we have the following:

$$\begin{aligned} &|u(S \cup \{i\}) - u(S)| \leq \varepsilon_j, \qquad \text{we use (I)} \\ &|u(S \cup \{i\} \cup \{k\}) - u(S \cup \{i\})| \leq \varepsilon_i, \qquad i \in S \cup \{i\}, \text{ we use (II)} \\ &|u(S \cup \{i\} \cup \{k\}) - u(S \cup \{k\})| \leq \varepsilon_j \qquad j \in S \cup \{k\} \text{ , we use (I)} \end{aligned} \tag{15}$$

Using these three inequalities we have:

$$\begin{aligned} &|[u(S \cup \{i\}) - u(S)] + [u(S \cup \{i\} \cup \{k\}) - u(S \cup \{i\})] - [u(S \cup \{i\} \cup \{k\}) - u(S \cup \{k\})]| = \\ &|u(S \cup \{k\}) - u(S)| \overset{**}{\leq} |[u(S \cup \{i\}) - u(S)]| + |[u(S \cup \{i\} \cup \{k\}) - u(S \cup \{i\})]| + \\ &|[u(S \cup \{i\} \cup \{k\}) - u(S \cup \{k\})]| \leq \varepsilon_i + \varepsilon_j + \varepsilon_j \end{aligned} \tag{16}$$

where ** uses triangular inequality. Hence we have each element is at most $2\varepsilon_j + \varepsilon_i$ for both cases when $i \in S$ or $i \notin S$, thus we have:

$$\max_{j \in S \subseteq D} |(u(S \cup \{k\}) - u(S))| \leq (2\varepsilon_j + \varepsilon_i) \tag{17}$$

using the similar arguments as in part A, we have the following:

$$\begin{aligned} |E^2_{kj}| =& \leq \sum_{j \in S \subseteq D \setminus \{k\}} \frac{|S|!\ (d - |S| - 1)!}{d!} |(u(S \cup \{k\}) - u(S))| \\ &= \sum_{j \in S \subseteq D \setminus \{k\}} \frac{|S|!\ (d - |S| - 1)!}{d!} (2\varepsilon_j + \varepsilon_i) = \frac{d!}{2}(2\varepsilon_j + \varepsilon_i) \end{aligned} \tag{18}$$

□

B.2 COROLLARY 1.1.

For the corollary we did not mention what u is, to compute $E^2(u)$, we need u. In this corollary we assume that the utility function is monotone. In particular, Utility function $u : P(D) \to \mathbb{R}$ is monotone iff

$$\forall S, S' \text{ s.t } S \subseteq S' \subseteq P(D) \implies u(S) \leq u(S').$$

This assumption on utility states that more features given to the model does not hurt. An exmple of such utility function is mutual information, i.e., $u(S) = I(X_S; Y)$.

Corollary 1.1. *(Transitivity): If E is the Shapley explanation map and u be a monotone utility function, then graph $\mathcal{H}$ is transitive.*

Proof. If $E^2(u)_{ij} = 0$ and $E^2(u)_{ki} = 0$ we want to show $E^2(u)_{kj} = 0$

Based on the assumption u is monotone, hence every marginal gain is greater or equal than zero, i.e., $u(S \cup \{i\}) - u(S) \geq 0$, for all $S \subseteq P(D)$ an $i \in D$.

Based on the assumption we have $E^2(u)_{ij} = 0$, i.e.,

$$E^2(u)_{ij} = 0 \implies \sum_{j \in S \subseteq D \setminus \{k\}} \frac{|S|!\ (d - |S| - 1)!}{d!} (u(S \cup \{i\}) - u(S)) = 0 \tag{19}$$

But every element of the sum is greater or equal than zero hence, $\max_{j \in S \subseteq D} |u(S \cup \{i\}) - u(S)| = 0$, similarly from $E^2(u)_{ki} = 0$ we have $\max_{i \in S \subseteq D} |u(S \cup \{k\}) - u(S)| = 0$. Using Theorem 1 result we have:

$$|E^2_{kj}| \leq \frac{d!}{2}(2\varepsilon_j + \varepsilon_i) = 0 \implies E^2_{kj} = 0. \tag{20}$$

□

APPENDIX C PAGERANK

C.1 PAGERANK

PageRank (Page et al., 1999) is an algorithm used by Google search in order to give a importance ranking for web pages in their search engine. Page rank output is a probability distribution which represent the likelihood of a person random clicking on different links to end up in a specific web page form (Page et al., 1999).In here we overview the PageRank algorithm. The PageRank scores $s_i \in [0,1]$, where $\sum_{i \in V} s_i = 1$, are given as the solution of the following system of equations:

$$s_i = p_i \cdot \alpha + \sum_{j:(j,i)\in E} \frac{w_{ji}}{d_j} s_j \quad \text{for all } i \in V,$$

where $\alpha \in [0,1]$ is a dampening factor (default value of 0.85),

$$d_j = \sum_{k:(j,k)\in E} w_{jk}$$

is the outgoing weighted degree of node $j \in V$ and $[p_i]_{i\in V}$ is a probability distribution over V. In standard PageRank, $p_i = \frac{1}{|V|}$, i.e., p is the uniform distribution. In personalized pagerank, a different distribution, possibly differentiated per node, is used.

Intuitively, the PageRank scores correspond to the steady state random walk over the weighted graph with random restarts: with probability $(1-\alpha)$ the walker transitions to an edge selected with a probability proportional to neighboring edge weights. With probability α, the walker jumps to a random node in V, sampled from probability distribution p. Usually, they are via iterative applications of the above random walk transition equations, applied to a starting distribution over V(Newman, 2018; Page et al., 1999).

APPENDIX D DERIVATION OF BIVARIATE SHAPLEY EXPLANATION MAP FORMULA

To prove the equation (6), we need to compute the E^2_{ij} elements of the matrix E^2_{ij}. E^2_{ij} is the element in intersection j^{th} column and i^{th} row. Based on the definition of E^2, we j^{th} column is represented as $E(u_j)$ where u_j is defined as in eq (5). In the case of shapley explanation $E(u_j)_i$ has specific form based on Shapley value eq (4), i.e.,

$$E(u_j)_i = \sum_{S\subseteq D\setminus\{i\}} \frac{|S|!\,(d-|S|-1)!}{d!}(u_j(S\cup\{i\}) - u_j(S)) \tag{21}$$

From the definition of u_j we know it is zero if $j \notin S$, thus we can remove those from the summation, i.e.,

$$\begin{aligned} E(u_j)_i = &\sum_{j\in S\subseteq D\setminus\{i\}} \frac{|S|!\,(d-|S|-1)!}{d!}(u_j(S\cup\{i\}) - u_j(S)) + \\ &\sum_{j\notin S, i\subseteq D\setminus\{i\}} \frac{|S|!\,(d-|S|-1)!}{d!}(u_j(S\cup\{i\}) - u_j(S)) = \\ &\sum_{j\in S\subseteq D\setminus\{i\}} \frac{|S|!\,(d-|S|-1)!}{d!}(u(S\cup\{i\}) - u(S)) + 0 \end{aligned} \tag{22}$$

APPENDIX E EXTENSION OF BIVARIATE EXPLANATION MAP TO MULTIVARIATE EXPLANATION

To generalize the bivariate explanation map E^2, define $E^k : \mathcal{U} \to \mathbb{R}^{\overbrace{d\times\cdots\times d}^{\text{k times}}}$, which outputs a tensor, let T be the tensor output, each element of this tensor would be denoted as $T^{i_1\ldots i_k} \in \mathbb{R}$, hence each column would be defined as $T^{i_1\ldots i_{k-1}} \in \mathbb{R}^d$ and similar to E^2 is defined as $E(u_{i_1\ldots i_{k-1}})$, where $u_{i_1\ldots i_{k-1}}$ is defined as follows:

$$u_{i_1\ldots i_{k-1}} : P(D) \to \mathbb{R} \implies \forall S \in P(D),\, u_{i_1\cdots_{k-1}}(S) = \begin{cases} u, \text{ if } \{i_1,\ldots i_{k-1}\} \subseteq S \\ 0, \text{ if } \{i_1,\ldots i_{k-1}\} \not\subseteq S \end{cases} \tag{23}$$

Algorithm 1 Approximate Graph $\mathcal{G}$ with Shapley Sampling Algorithm

Input : Data Sample $x \in \mathcal{X} \subset \mathbb{R}^d$, Utility Function f, Number of Samples M
Output : Adjacency Matrix $\mathcal{G} \in \mathbb{R}^{d \times d}$

Initialize $\mathcal{G} = 0$
for *i=1...d* **do**
  for *m=1...M* **do**
    Create random permutation $\mathcal{O}$ of size d
    Define $\tilde{i}$ as permuted index of feature i
    Define the set of indices $s = \{\mathcal{O}_{1...\tilde{i}-1}\}$ and the set of all indices $D = \{\mathcal{O}\}$
    Sample random baseline $w \in \mathcal{X}$

    $b_1 = x_{s \cup i} \oplus w_{D \setminus \{s \cup i\}}$ \\ Symbol $\oplus$ indicates concatenation
    $b_2 = x_s \oplus w_{D \setminus s}$

    for $j \in s$ **do**
      $\mathcal{G}_{ij} = \mathcal{G}_{ij} + f(b_1) - f(b_2)$
    end
  end
end
$\mathcal{G} = \frac{1}{M}\mathcal{G}$

Return $\mathcal{G}$

APPENDIX F DETAILS OF EXPERIMENTAL SETUP AND ADDITIONAL EXPERIMENTAL RESULTS

F.1 EXPERIMENT SETUP

F.1.1 ALGORITHMS

Approximating Graph $\mathcal{G}$ with Shapley Sampling. Computation over all subsets of features is computationally expensive. In practice we can use a approximate the Bivarate Shapley value over a fixed number of samples by adapting the sampling algorithm introduced by Štrumbelj & Kononenko (Štrumbelj & Kononenko, 2014), as seen in Alg. 1. Note that computing the $\mathcal{G}$ matrix adds no complexity to the original algorithm; we simply keep track of when feature j is absent (i.e. $j \notin S$ and set the value function output to zero when this condition occurs. We can therefore calculate the bivariate and univariate shapley values concurrently. Note that since we are discarding or "filtering out" the samples where j is absent, we need to double the number of samples to achieve the same approximation accuracy as the univariate calculation.

Approximating Graph $\mathcal{G}$ with KernelSHAP.

As mentioned in Section 4, our method can be generalized to any removal-based Shapley approximation or other removal-based explainer. More concretely, each column of the $d \times d$ interaction matrix, representing interactions between d features, can be considered an independent explanation where the column feature is always present. Therefore our method is extremely flexible; the user can decide which explanation method to use based on the constraints of their intended application. However, we can also improve performance by taking advantage of different approximation methods.

Performance is improved through the use of two properties in KernelSHAP. First, we can save the sampled model outputs and reuse these values when recalculating KernelSHAP over all d features. This allows for a fixed number of model samples independent of the number of data features. Second, note that KernelSHAP attributions are calculated through a weighted linear regression mechanism. Bivariate Shapley simply changes the model output (setting the output to zero) depending on whether a feature is present or removed, which corresponds to applying a binary mask over the linear regression labels. Therefore we can save the intermediate linear regression calculation and evaluate each column

Algorithm 2 Approximate Graph $\mathcal{G}$ with KernelSHAP Algorithm

Input : Data Sample $x \in \mathcal{X} \subset \mathbb{R}^d$, Utility Function f, Number of Samples M
Output : Univariate Shapley Values $\phi \in \mathbb{R}^d$, Adjacency Matrix $\mathcal{G} \in \mathbb{R}^{d \times d}$

Initialize $\mathcal{G} \in \mathbb{R}^{d \times d}$
Initialize matrix $\tilde{X} \in \{0,1\}^{M \times d}$
Initialize matrix Π as an identity matrix $\mathbb{I}_M$
Initialize vector $Y \in \mathbb{R}^M$
Define the KernelSHAP weighting kernel $\pi(x) = \frac{(d-1)}{(d \text{ choose } |x|)|x|(d-|x|)}$

\\ Randomly draw M samples around x
for *m=1...M* **do**
  Sample random baseline $w \in \mathcal{X}$
  Sample binary vector $\tilde{x} \in \{0,1\}^d$
  Calculate perturbed labels $y = f(\tilde{x} \odot x + (1 - \tilde{x}) \odot w)$ \\ Symbol $\odot$ indicates Hadamard product

  $\tilde{X}_{m,:} \leftarrow \tilde{x}$ \\ $X_{i,j}$ indicates indices i and j in matrix X. : indicates the entire row / column.
  $Y_m \leftarrow \tilde{y}$
  $\Pi_{m,m} \leftarrow \pi(\tilde{x}_m)$
end

\\ Solve a constrained [2], weighted linear regression
Define $\Gamma = (\tilde{X}^T \Pi \tilde{X})^{-1} \tilde{X}^T \Pi$
Define $\Gamma^+ = \Gamma_{-d,:}$ \\ remove the last row of Γ
Define $\Gamma^- = \Gamma_{-1,:}$ \\ remove the first row of Γ

\\ Remove the last feature in regression calculation with the constraint that $\sum_i \phi_i = f(x) - f(\mathbb{E}[\mathcal{X}])$
Define $\phi = \Gamma^+[Y - X_{:,d} \times (f(x) - f(\mathbb{E}[\mathcal{X}]))]$
$\phi \leftarrow \phi \oplus (f(x) - f(\mathbb{E}[\mathcal{X}]) - \sum_i \phi_i)$ \\ Enforce constraint. $\oplus$ indicates concatenation.

\\ Iterate regression calculation over filtered labels
for *j=1...d* **do**
  Define $Y^+ = Y \odot \tilde{X}_{:,j}$ \\ Set $Y_m = 0$ if feature j was not selected in $\tilde{X}_{m,:}$
  Define $\phi^+ = \Gamma^+[Y^+ - X_{:,d} \times f(x)]$

  Define $Y^- = Y \odot (1 - \tilde{X}_{:,j})$ \\ Set $Y_m = 0$ if feature j was selected in $\tilde{X}_{m,:}$
  Define $\phi^- = \Gamma^-[Y^- + X_{:,d} \times f(\mathbb{E}[\mathcal{X}])]$

  $\phi^+ \leftarrow \phi^+ \oplus (\phi_d - \phi^-_{d-1})$ \\ Utilize the property that $\phi = \phi^- + \phi^+$
  $\mathcal{G}_{:,j} \leftarrow \phi^+$
end

Return ϕ, $\mathcal{G}$

of the interaction matrix with two matrix multiplications with the sparse labels. This results in significant speed improvements and allows scaling to datasets with large number of features, such as the COPD dataset with 1,077 features.

Mutual Redundancy on Graph $\mathcal{H}$. Given the unweighted graph $\mathcal{H}$, we want to find groups of mutually redundant features as investigated in Fig. 2. These features are identified as strongly connected nodes within the graph. We use the package NetworkX (**?**), which implements Tarjan's

[2]Note that $\pi(x) = \infty$ when $|x| \in \{0, d\}$, therefore in practice we remove two variables during the linear regression calculation and enforce the following two constraints. 1) $\phi_0 = f(\mathbb{E}[\mathcal{X}])$, where ϕ_0 is defined as the bias / intercept term in the regression, and 2) $\sum_i \phi_i = f(x) - f(\mathbb{E}[\mathcal{X}])$.

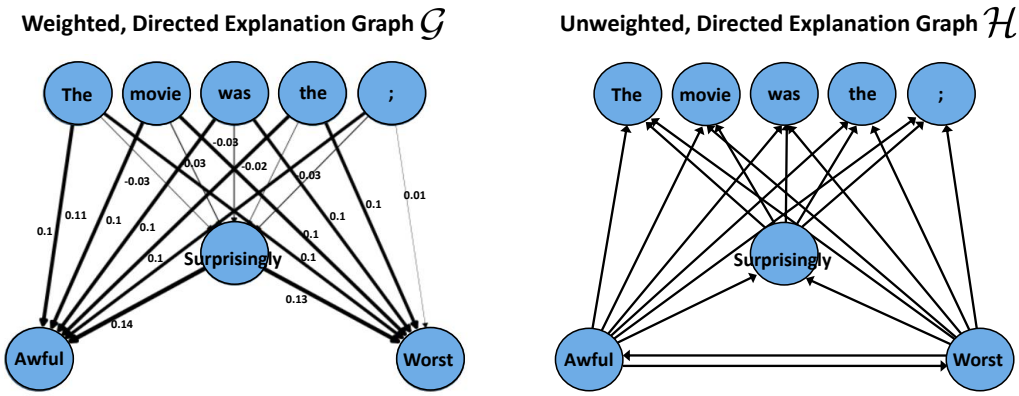

Figure 5: Comparison of graph $\mathcal{G}$ and graph $\mathcal{H}$ for the the given IMDB example "The movie was the worst; surprisingly awful", which is classified as negative sentiment. Note that the sinks and sources of graph $\mathcal{G}$ and graph $\mathcal{H}$ are reversed in terms of influential and redundant features. I.E. the **source** nodes of graph $\mathcal{G}$ represent redundant features, whereas the **sink** nodes of graph $\mathcal{H}$ represent (directionally) redundant features.

algorithm (Tarjan, 1972) to identify such nodes. Tarjan's algorithm is a depth-first search that runs in linear time. This algorithm for identifying mutually redundant features is outlined in Alg. 3.

To generate the results of Fig. 2, we apply a binary mask for each sample such that a given percentage of its mutually redundant features are set to their baseline value. Note that the number of features masked for a given percentage may vary between samples, since the $\mathcal{H}$ is calculated on an instance-wise basis. We then record the accuracy for the set of masked samples.

Algorithm 3 Mutual Redundancy on Graph $\mathcal{H}$

Input : Unweighted Directed Graph $\mathcal{H}$
Output : Groups of Mutually Redundant Features

Define $\mathcal{S} = \{s_1, ..., s_m\}$ as the set of strongly connected subgraphs in $\mathcal{H}$

Return $\mathcal{S}$

Directional Redundancy on Graph $\mathcal{H}$. Directional redundancy is defined in terms of $\mathcal{H}$-sinks and $\mathcal{H}$-sources on graph $\mathcal{H}$, which we investigate in Table 2. There are a number of methods to identify source and sink nodes on a graph; in our implementation we use the PageRank algorithm (Page et al., 1999) and take the maximally and minimally-ranked node as the sink and source, respectively (Alg. 4). Note that using PageRank in such manner will only identify singular sinks and sources, therefore we first separate graph $\mathcal{H}$ into its connected subgraphs and apply PageRank to the condensation graph of each subgraph. We use the PageRank implementation in the Scikit-Network package (Bonald et al., 2020) with no personalization and default damping $= 0.85$. In practice, we found that changing the damping parameter had no effect on identified features.

In Table 2 we show the results of completely masking all $\mathcal{H}$-sources and $\mathcal{H}$-sinks. We apply a binary mask, setting the value of all $\mathcal{H}$-source or $\mathcal{H}$-sink features to their baseline values, then record the sample accuracy.

Redundancy Ranking on Graph $\mathcal{G}$. We want to create a continuous ranking of feature redundancy given graph $\mathcal{G}$, as investigated in Fig. 3. We first add $\epsilon = 10^{-70}$ to each element in $\mathcal{G}$ to eliminate disconnected subgraphs. Note that for certain value functions, such as those used in our experiments, the graph G can contain negative values. We normalize these negative values by applying an element-wise Softplus function: $\text{Softplus}(x) = \ln(1 + e^x)$. We then directly apply the PageRank algorithm

Algorithm 4 Directional Redundancy on Graph $\mathcal{H}$

Input : Unweighted Directed Graph $\mathcal{H}$
Output : Source Nodes and Sink Nodes

Define $\mathcal{W} = \{w_1, ..., w_m\}$ as the set of weakly connected subgraphs in $\mathcal{H}$
for *i = 1,...,m* **do**
  Create condensed graph c_i = Condensation(w_i)
  Source Node α_i = argmax PageRank (c_i)
  Sink Node ω_i = argmin PageRank (c_i)
end
Source Nodes = Condensation$^{-1}(\{\alpha_1, ..., \alpha_m\})$
Sink Nodes = Condensation$^{-1}(\{\omega_1, ..., \omega_m\})$

Return Source Nodes, Sink Nodes

from Scikit-Network to obtain feature rankings. We again use the default damping parameter of 0.85 for all datasets.

One issue we observed during testing was the occurrence of nodes with identical PageRank scores, indicating a similar level of redundancy. With no other information, this would necessitate random selection when generating the feature ranking. With this motivation, we experiment with using the univariate shapley values as personalization values. In personalized PageRank, the personalization values dictate the distribution over nodes for which a random jump will land. With no personalization, a random jump will land in each node with equal probability; i.e. the personalization is assumed to be uniform. By setting the personalization to the univariate shapley values, we bias the stationary distribution towards nodes that have high shapley values. Therefore, nodes of similar redundancy would be further ranked by their respective univariate shapley values. In practice, using personalization slightly improves post-hoc accuracy results in Fig. 3 for larger masking percentages. The full algorithm for generating the redundancy ranking of features is outlined in Alg. 5.

Algorithm 5 Directional Redundancy Ranking on Graph $\mathcal{G}$.

Input : Weighted Directed Graph $\mathcal{G}$, *optional* Univariate Feature Ranking $R \in \mathbb{R}^d$
Output : Score vector $S \in \mathbb{R}^d$, representing relative feature importance for each feature.

Define A as the adjacency matrix for Graph $\mathcal{G}$

$\tilde{A} = A + 10^{-70}$ \\ Add $\epsilon \approx 0$ to ensure all nodes are connected
$\tilde{A} = \text{SoftPlus}(A)$ \\ Element-wise Softplus function to normalize negative values

if *Personalization* **then**
  S = PageRank($\tilde{A}$) with Personalization Values R
else
  S = PageRank($\tilde{A}$)
end
Return S

F.1.2 Implementation Details for Bivariate Shapley and Comparisons Models.

Unless otherwise specified, we use the default parameters when implementing comparison methods using publicly available code.

Removal-based methods typically assign a value to act as a proxy for a feature's absence during feature removal. This value is commonly referred to as a baseline, or reference value, and is often assigned to be some *a priori* neutral value. While different removal-based methods may have different

baseline values as default, we assign a single baseline value used for all methods for a given dataset. This is to maintain comparability, since the objective of our experiments is to evaluate the explanation calculation rather than the choice of baseline value. For tabular data, we define the value for all removed features to be zero, except the Divorce dataset where a value of '3' indicates the average response, and the Census dataset where we fix the baseline to be the average value for each feature. For images, we use a pixel value of zero. For text, we set the word embedding for the selected feature to be the zero vector.

Bivariate Shapley - Sampling We apply Bivariate Shapley on a variety of prediction models (detailed in section F.1.3), using a value function $v(S) = \mathbb{E}_{w\sim\mathcal{B}}[P(Y = \hat{y}|X = x_S \cup w_{\bar{S}})]$, where $\hat{y}$ is the model's predicted class, $\bar{S}$ is the complement of S, and w represents samples drawn from a baseline distribution $\mathcal{B}$. As previously discussed, this baseline distribution is fixed to a value that is dependent on the given dataset. We set m, the number of samples drawn in alg. 1 to be 1000.

Bivariate Shapley - Kernel We utilize the algorithm described in sec.F.1.1 by adapting the publicly available package for kernelSHAP (Lundberg & Lee, 2017). We keep the same default parameters as KernelSHAP, except we double the number of default samples to account for the Bivariate Shapley filtering.

Shapley Excess. Shapley Excess refers to the surplus value from contribution of players in a coalition game grouped in a singleton coalition as compared to their individual contributions. This can be written formally as:

$$\phi_s - \sum_{i\in s} \phi_i$$

where ϕ_s is the shapley value of a group of players when considered as a singleton player. We implement this formula using the KernelSHAP approximation by combining features and evaluating the resulting excess Shapley value.

Shapley Interaction Index. Introduced by (Grabisch & Roubens, 1999), Shapley Interaction Index has gained popularity due to the efficient implementation by (Lundberg et al.) on tree-based prediction models. In order to apply this method efficiently with the entirety of the datasets in our experiments, we use the KernelSHAP approximation to calculate Shapley Interaction Index. This implementation results in significantly faster calculations compared to Shapley Sampling approximations (as seen in tbl 7. We use the default parameters of KernelSHAP, applied $2 \times d$ times per sample, where d is the number of features.

Shapley Excess. Shapley Excess refers to the surplus value from contribution of players in a coalition game grouped in a singleton coalition as compared to their individual contributions. This can be written formally as:

$$\phi_s - \sum_{i\in s} \phi_i$$

where ϕ_s is the shapley value of a group of players when considered as a singleton player. We implement this formula using the KernelSHAP approximation by combining features and evaluating the resulting excess Shapley value.

Shapley Taylor Index. Introduced by Sundararajan et al. (2020b). As of this writing, there is no publicly available code for the Shapley Taylor Index. Therefore we build our own implementation using the Shapley Sampling approximation as outlined in the original paper. We choose a sample size of $m = 200$ for each element of the interaction matrix.

GNNExplainer. Introduced by Ying et al. (2019), GNNExplainer is a method for explaining a GNN-based black-box model. It can be used on a variety of GNN tasks, such as node or graph classification, and identifies a compact subgraph and subset of node features that best explains the GNN output. This is accomplished through the use of a soft mask on the edges and node features of the input graph. Specifically, GNNExplainer trains a neural network to generate the edge and node feature masks, with the objective of maximizing mutual information between the black-box output of the masked graph and the label.

While GNNExplainer was originally intended to explain GNN models, it can be used in conjunction with non-GNN models. In our implementation, our objective is to identify the important edges between the features of a data sample. Therefore we define a fully-connected graph with the features as the graph nodes. When applying GNNExplainer to this fully-connected graph, GNNExplainer

Domain	**Genetics**	**Image**		**Text**	**Tabular**		
Dataset	COPDGene	CIFAR10	MNIST	IMDB	Census	Divorce	Drug
Classes	2	10	10	2	2	2	2
Train/Test Samples	1,641/407	50k/10k	60k/10k	25k/25k	26k/6.5k	102/68	1413/472
Model	4-Layer MLP	Resnet18	2-Layer CNN	1-Layer GRU	XGBoost	3-Layer MLP	Random Forest
Model Accuracy	88.2	89.8	99.0	88.1	87.3	98.5	85.3

Table 3: Summary of the datasets and models in our investigation

returns edge importance values for the given data sample. This output can be converted to a subgraph using specifying a threshold, below which the edges are removed. In our experiments, we directly use the edge importance values as the weights of a directed, weighted graph. This resulting graph is then evaluated and compared with Bivariate Shapley using the same algorithms for identifying mutually redundant features, directionally redundant features, and feature redundancy ranking, as outlined in App. F.1.1. We implement GNNExplainer using the Pytorch Geometric package (Fey & Lenssen, 2019) with default parameters.

Note that while GNNExplainer can indeed be applied to non-GNN models, these models may not be able to incorporate the graph structure in its predictions. For example, even though GNNExplainer applies an edge mask to the input graph, this edge information is meaningless if the black-box model is not designed to use this structure in its prediction. In this case, the GNNExplainer will receive non-informative black-box outputs in its mutual information maximization objective.

F.1.3 Datasets and Models

COPDGene. The COPDGene dataset is an observational study with a cohort of 10,000 participants designed to identify the genetic risk factors for COPD. The study contains participants with and without COPD; COPD diagnosis, subtyping, and progression are monitored using high-resolution CT scans. We are interested in investing the relation between gene expression and smoking status (see section F.2.6 for details). The dataset contains RNA-sequencing count data for 1,077 genes and the associated binary label for smoking status. We use a neural network with 4 fully-connected layers of 200 hidden units, batch normalization, and relu activation. The model is trained using Adam (Kingma & Ba, 2015) with learning rate 10^{-3} for 800 epochs, achieving a test accuracy of 88.2%.

CIFAR10. CIFAR10 (Krizhevsky, 2009) consists of 60k images of dimension 32×32 with RGB channels. We train a Convolution Neural Network (CNN) to classify the 10 different classes, using a Resnet18 architecture (He et al., 2016) with default parameters. We apply color jittering and horizontal flip data augmentations, as well as data normalization. The model is trained using Adam with learning rate 10^{-3} for 80 epochs, achieving a test accuracy of 89.8%.

While it is possible to use individual pixels when calculating Bivariate Shapley, we choose to use superpixels to reduce computation and improve the interpretability of results. Superpixels are contiguous clusters of pixels that are treated as a single feature for feature importance purposes; i.e. all individual pixels within the superpixel are masked or selected jointly. We use the *simple linear iterative clustering* (SLIC) algorithm (Achanta et al., 2012) in our image experiments. SLIC divides the image into similarly sized superpixels based on clustering in the CIELAB color space. For CIFAR10, we use SLIC with 255 superpixels and minimal smoothing ($\sigma = 5$).

MNIST. MNIST (LeCun & Cortes, 2010) consists of 28×28 greyscale images with the handwritten numerals $0 - 9$. We train a CNN with two convolution layers and a single batch normalization layer. Each convolution uses a 6×6 kernel size, stride 2, and a 200 channel mapping. We train the model using stochastic gradient descent (SGD) with learning rate 10^{-2} for 20 epochs, achieving a test accuracy of 99.0%. We again use SLIC to create superpixels; for MNIST we use 196 superpixels and $\sigma = 5$.

IMDB. The Large Movie Review Dataset (IMDB) (Maas et al., 2011) consists of 50k movie reviews which we use for the task of sentiment analysis. We train a Recurrent Neural Network (RNN) classifier with a single Gated Recurrent Unit (GRU) (Cho et al., 2014) layer of 500 hidden units to predict either positive or negative sentiment. We tokenize each review using the NLTK package (Loper & Bird, 2002) and map each token to a pretrained word embedding. We use the 300-dimensional GloVe (Pennington et al., 2014) embedding with 840B tokens, pretrained on the Common Crawl dataset.

	Insertion AUC (Higher is better)							Deletion AUC (Lower is better)						
Dataset	COPD	CIFAR10	MNIST	IMDB	Census	Divorce	Drug	COPD	CIFAR10	MNIST	IMDB	Census	Divorce	Drug
Ours-SS	0.48	0.75	0.85	0.45	0.43	0.30	0.30	0.01	0.05	0.03	0.02	0.32	0.05	0.10
Ours-K	0.49	0.65	0.84	0.43	0.42	0.30	0.30	0.00	0.08	0.03	0.02	0.32	0.05	0.10
Sh-Sam	0.48	0.75	0.85	0.45	0.43	0.30	0.30	0.01	0.05	0.03	0.02	0.32	0.05	0.10
kSHAP	0.42	0.48	0.77	0.29	0.42	0.29	0.30	0.09	0.17	0.17	0.03	0.36	0.05	0.11
Sh-Int	0.20	0.35	0.46	0.32	0.43	0.16	0.17	0.23	0.31	0.52	0.29	0.33	0.14	0.20
Sh-Tay	–	0.34	0.78	0.34	0.06	0.17	0.29	–	0.27	0.19	0.19	0.06	0.10	0.11
Sh-Exc	–	0.32	0.51	0.30	0.37	0.15	0.08	–	0.31	0.48	0.29	0.38	0.15	0.29
GNNExp	0.25	0.15	0.25	0.30	0.38	0.25	0.26	0.25	0.15	0.25	0.30	0.38	0.25	0.27

Table 4: Influential Feature Evaluation through Insertion and Deletion AUC. We calculate a feature ranking by applying PageRank on the $\mathcal{G}$ graph, iteratively removing the most influential feature, then evaluating AUC on the resulting curve. Note that we cannot run Sh-Tay and Sh-Exc methods on the COPD dataset due to their computational issues with the large number of features.

We limit the vocabulary to 10k tokens and text sample length to 400 tokens. The model was trained using Adam with learning rate 10^{-4} for 15 epochs, achieving test accuracy of 88.1%.

Census. The UCI Census Income dataset aggregates data from the 1994 census dataset. We use 12 features, including both continuous and discrete data, to predict whether an individual has an annual income greater than $\$50k$. Our model is trained using XGBoost (Chen & Guestrin, 2016) with a maximum of 5000 trees, $\eta = 0.01$, and subsample = 0.5, achieving 87.3%test accuracy.

Divorce. The UCI Divorce Predictors dataset (Yöntem et al., 2019) consists of a 54-question survey with 170 participants regarding various activities and attitudes towards their partners. Each question is answered with a ranking on a scale from $1 - 5$. We train a 3-layer MLP with relu activation, predicting if the participant was divorced. Each hidden layer contained 50 hidden units. The model was trained using SGD with learning rate 0.1 and achieved test accuracy of 98.5%. During the Bivariate Shapley calculation, we use a baseline value of 3 to indicate a feature's absence, as this represents the value representing a neutral response.

Drug. The UCI Drug Consumption dataset (Fehrman et al., 2015) consists of 1,885 responses to an online survey concerning the consumption habits of various drugs. We use binary features for the six drugs nicotine, marijuana, cocaine, crack, ecstasy, and mushrooms, indicating whether the respective drug has been previously consumed. We build a model to predict whether the participant has also consumed a seventh drug, LSD. We use a random forest model with 100 trees, achieving a test accuracy of 85.3%

F.1.4 Licenses for COPDGene Data

All participants provided their informed consent, and IRB approval was obtained from all concerned institutions. IRB information will be provided once anonymity has been lifted.

F.2 Additional Experimental Results

F.2.1 Insertion and Deletion AUC

Insertion AUC (iAUC) and Deletion AUC (dAUC), introduced by (Petsiuk et al.), quantify the ability for an explainer to find the most influential features of a given black-box model. We use iAUC and dAUC as a supplementary metric to evaluate the redundancy-based ranking we explore in figure 3.

To summarize, dAUC iteratively removes the highest-ranked features of a given image and measures the change in model output compared to the baseline prediction, as summarized by the area under the curve. Lower dAUC values indicate that the explainer can accurately assess the features most influential towards the model output. Conversely, iAUC starts with an uninformative baseline sample then iteratively inserts the highest-ranked features, then measures change in model output through AUC. Higher values of iAUC indicate better performance. We evaluate Bivariate Shapley, as well as a variety of popular univariate and bivariate black-box explainers, on these two metrics in table 4.

	10% Features Mask					50% Features Mask				
Dataset	Ours	Shap Sampl	Int	KernelSHAP	L2X	Ours	Shap Sampl	Int	KernelSHAP	L2X
COPD	100 ± 0.0	100 ± 0.0	82.8 ± 1.9	99.3 ± 0.4	92.6 ± 1.3	100 ± 0.0	100 ± 0.0	68.3 ± 2.3	100 ± 0.0	86.0 ± 1.7
CIFAR10	99.4 ± 0.3	99.0 ± 0.4	70.2 ± 2.0	86.6 ± 1.0	71.4 ± 2.0	93.0 ± 1.1	92.4 ± 1.2	32.8 ± 2.1	54.9 ± 1.4	23.2 ± 1.9
MNIST	100 ± 0.0	100 ± 0.0	84.6 ± 1.6	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	62.8 ± 2.2	99.9 ± 0.4	100 ± 0.0
IMDB	100 ± 0.0	100 ± 0.0	92.6 ± 1.2	100 ± 0.0	94.0 ± 1.2	100 ± 0.0	100 ± 0.0	64.4 ± 2.1	100 ± 0.0	57.9 ± 2.2
Census	100 ± 0.0	100 ± 0.0	100 ± 0.0	96.0 ± 0.9	96.6 ± 0.8	96.8 ± 0.8	96.8 ± 0.8	94.8 ± 1.0	90.0 ± 1.3	84.8 ± 1.6
Divorce	100 ± 0.0	100 ± 0.0	98.5 ± 1.5	100 ± 0.0	100 ± 0.0	100 ± 0.0	100 ± 0.0	58.8 ± 6.0	98.5 ± 1.5	98.5 ± 1.5
Drug	100 ± 0.0	100 ± 0.0	91.7 ± 1.3	100 ± 0.0	100 ± 0.0	99.2 ± 0.4	99.2 ± 0.4	77.1 ± 1.9	100 ± 0.0	75 ± 2.0

Table 5: Accuracy results for masking redundant features as identified using PageRank on graph G. These results mirror Figure 3 but with the result variance, as represented by $\pm$ standard deviation. Note that for datasets with < 10 features, the given feature mask percentage is approximate.

Bivariate Shapley-S				
	PH-Accy		% Feat Masked	
Dataset	Sink Masked	Source Masked	Sink Masked	Source Masked
COPD	99.5	62.7	1.5	98.5
CIFAR10	94.6	15.0	6.2	93.8
MNIST	100.0	13.4	77.7	22.3
IMDB	100.0	54.0	3.5	96.5
Census	100.0	82.0	23.8	76.2
Divorce	100.0	51.5	22.2	77.8
Drug	100.0	48.5	43.5	56.5

Bivariate Shapley-K				
	PH-Accy		% Feat Masked	
Dataset	Sink Masked	Source Masked	Sink Masked	Source Masked
COPD	97.3	62.7	13.6	86.4
CIFAR10	82.6	19.4	10.4	89.6
MNIST	100.0	17.6	13.6	86.4
IMDB	97.2	54.0	23.7	76.3
Census	100.0	82.0	33.3	66.7
Divorce	100.0	51.5	22.0	78.0
Drug	100.0	48.5	43.5	56.5

Table 6: Posthoc-accy of BivShap-S and BivShap-K after masking $\mathcal{H}$-source nodes, representing features with minimal redundancies, and $\mathcal{H}$-sink nodes, representing directionally redundant features.

F.2.2 Sampling Variance of Post-Hoc Accuracy Results

The Bivariate Shapley method, like other shapley-based methods, does not involve any training or optimization of weights. Therefore it does not suffer from issues related to data variability. In addition, the quantitative results from our experiments are averaged over ≈ 500 test samples (less for divorce and drug, due to dataset size). We show the variance results for Figure 3 in Table 5.

F.2.3 BivShap-K results for Sink and Source Masking on Graph $\mathcal{H}$

The BivShap-K results for sink and source masking on graph $\mathcal{H}$ are omitted in table 2 due to space constraints. We present the full full results in table 6.

F.2.4 Sensitivity of Graph $\mathcal{H}_\gamma$ to γ

In Section 4.1 we define a relaxed version of the redundancy graph $\mathcal{H}_\gamma = (V_\mathcal{H}, E^\gamma_\mathcal{H})$ where $V_\mathcal{H} = V_\mathcal{G}$ and $E^\gamma_\mathcal{H} = \{(i,j) \in E_\mathcal{G} : |W_\mathcal{G}(i,j)| \leq \gamma\}$. Intuitively, $\gamma \in \mathcal{R}^+$ acts as a threshold to define redundant edges in $\mathcal{H}_\gamma$. As γ increases, the number of edges in $\mathcal{H}_\gamma$ also increases, resulting in larger mutually redundant clusters and a higher sensitivity to directional redundancy. From the perspective of accurately representing the black-box model, the choice of γ presents a tradeoff akin to sensitivity and specificity: larger γ values more easily identify true redundancies within the data (increased sensitivity), at the cost of potentially mislabeling non-redundancies (reduced specificity).

While it is trivial to choose γ through cross-validation using post-hoc accuracy (or equivalent metric), such methods are not ideal for instance-wise explanation purposes where the practitioner may not have access to a sufficient number of validation samples. We therefore attempt to establish guidelines for choosing γ. In particular we investigate the effect of γ on graph density (Figure 6). Note that increasing γ also increases the density of graph $\mathcal{H}$. We can see that at a certain density, post-hoc accuracy exhibits a sharp decrease, suggesting that the identified redundancies are not truly redundant. The ideal γ depends on the level of redundancy in the dataset, therefore the value should be chosen based on the given task. For experimental purposes, we use a constant $\gamma = 10^{-5}$ for all datasets.

We also explore how increasing γ affects the identification of mutually redundant features in Figure 7. We similarly see that increasing γ increases the number of edges in graph $\mathcal{H}$, which correspondingly increases the number of mutually redundant features identified. For datasets with inherently low mutual redundancy, such as CIFAR10, this has the effect of reducing Post-hoc accuracy when γ is

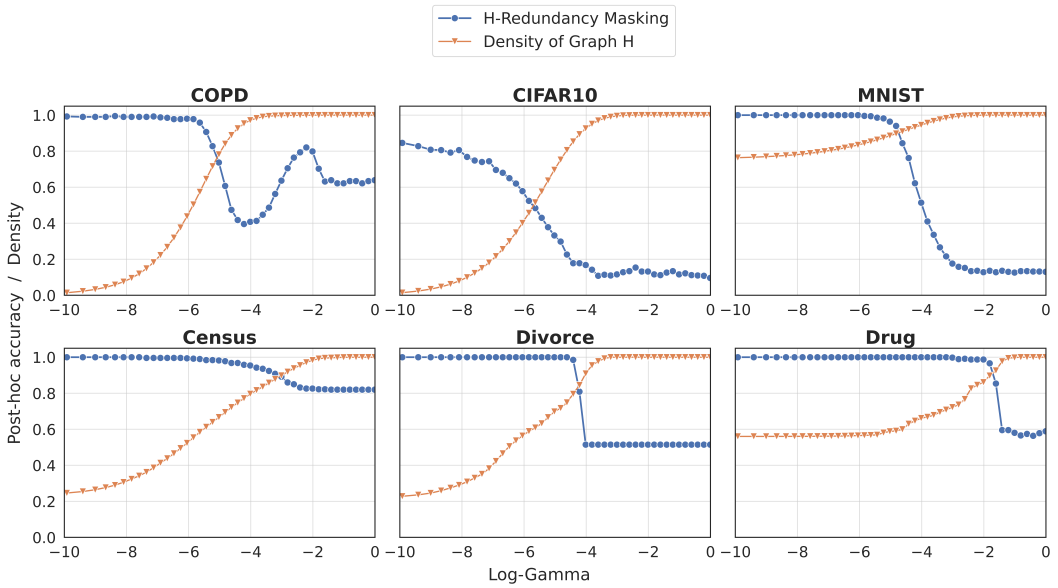

Figure 6: Sensitivity analysis of graph $\mathcal{H}$ to parameter γ. We compare the density of $\mathcal{H}$ as γ increases to the post-hoc accuracy after masking all directionally redundant features found in $\mathcal{H}$.

		Bivariate Methods					Univariate Methods			GNN Methods
Dataset	# Features	Ours-SS	Ours-K	Sh-Int	Sh-Tay	Sh-Exc	Sh-Sam	kSHAP	L2X	GNNExp
COPD	1077	5942	36	2877	112900*	838200*	3047	1.4	0.00	10.9
CIFAR10	255	218	2.5	101	2819*	6267*	140	0.65	0.00	0.79
MNIST	196	116	1.5	48	1194*	2350*	57	0.34	0.00	0.42
IMDB	$\leq$400	207	1.9	160	1279*	1796*	103	0.40	0.00	0.73
Census	12	2.7	0.20	2.6	11.6	5.3	1.6	0.83	0.00	0.17
Divorce	54	18.2	0.34	6.5	63.2	93.3	11.3	0.16	0.00	0.15
Drug	6	2.3	0.07	1.21	181	0.96	1.26	0.10	0.00	1.54

Table 7: Time comparison in seconds per data sample for the methods used for the post-hoc accuracy and AUC calculations. Fields indicated by * which were averaged over 5 samples due to computational cost, otherwise time calculations were averaged over 500 samples (or the total number of test samples if fewer than 500)

increased past a certain threshold. Therefore in practice γ should be selected either through cross validation, or by examining the density curve as in Figure 6.

F.2.5 Time complexity Details with Univariate Comparison.

Table 7 includes the full feature attribution timing results. Note that L2X requires an initial training stage for neural network-based explainer model, which is not included in these results. Once this explainer model is trained, the topk features are obtained through single forward pass, which is the activity measured in Table 7. All experiments are performed on an internal cluster equipped with Intel Gold 6132 CPUs. The evaluations on CIFAR10, MNIST, IMDB datasets were calculated using GPUs (Nvidia Tesla V100), whereas the other datasets were trained without a GPU. Finally, the calculated times were averaged over all samples used in the experiments (500 samples, unless the dataset has less than 500 samples total), except for the fields indicated by * which were averaged over 5 samples due to computational cost.

As previously mentioned in Sec F.1.1, the Bivariate Shapley method can be applied naively to any removal-based explanation method repeating the explainer's calculations d times, where d is the number of data features. It follows that our method's time complexity is dependent on the choice explanation method and, when implemented naively, linearly scales that method's complexity by the number of features. Certain methods, such as KernelSHAP, can be adapted to realize even more

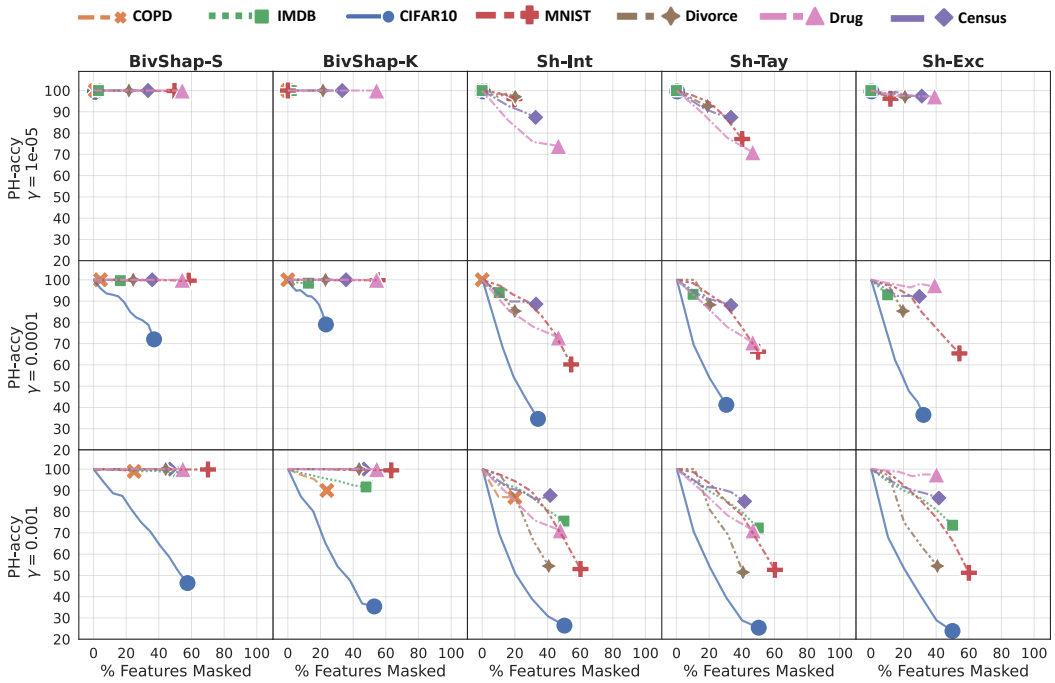

Figure 7: Posthoc Accuracy evaluated on Mutual Redundancy masking derived from graph $\mathcal{H}$. Strongly connected nodes in $\mathcal{H}$ are randomly masked with increasing mask sizes until a single node remains, represented by the final marker for each dataset. Each row represents a different selection of threshold parameter γ. Note that we cannot run Sh-Tay and Sh-Exc methods on the COPD dataset due to their computational issues with the large number of features.

efficient implementations of Bivariate Shapley, which we outline in App F.1.1. We provide time comparisons to competing methods in Table 1.

F.2.6 Gene Ontology Enrichment Analysis of COPDGene Dataset

Chronic Obstructive Pulmonary Disease (COPD) is a chronic inflammatory lung disease. The relation between COPD and smoking is well-established; it has been shown that smoking increases the risk of developing lung disease through a variety of ways, such as increasing lung inflammation (Arnson et al., 2010). Here, we investigate the relation between gene expression data and smoking status in COPDGene data. We show the interpretation power of our methods by relating our most influential genes to biological pathways which correspond to smoking. We performed Gene Set Enrichment Analysis (GSEA) using the GenePattern web interface (Reich et al., 2006) on the ranking of influential features, which we generate as follows. We first calculate graph $\mathcal{G}$ locally, as in Alg. 1. We then create the global $\mathcal{G}$ graph for each subgroup, smokers and non-smokers, by averaging the $\mathcal{G}$ adjacency matrix over all samples within each subgroup (Fig. 8). We directly apply the ranking algorithm outlined in Section F.1.1 to obtain subgroup-specific importance scores. We use the list of 1,079 unique gene names with their associated importance score as input into the GenePattern interface. Gene set enrichment for these two groups was calculated using the GSEAPreranked module with 1000 permutations, using the Hallmark (h.all.v7.4.symbols.gmt) and Immunologic gene sets (c7.all.v7.4.symbols.gmt). We observed genetic pathways corresponding to Macrophages as statistically significant at a q-value ≤ 0.05 (the pathway table is in the App., Table. 8). Macrophages are a type of immune cells that can initiate inflammation, and they also involve the detection and destruction of bacteria in the body. The relation between such cells and smoking has been observed in biological domain; many studies have observed that smoking induces changes in immune cell function in COPD patients (Yang & Chen, 2018; Strzelak et al., 2018).

Group	Pathway Name	Genes	q-value
Smoker	**GSE25123_WT_VS_PPARG_KO_MACROPHAGE_IL4_STIM_DN**	**23**	**0.02**
	GSE32986_GMCSF_VS_GMCSF_AND_CURDLAN_LOWDOSE_STIM_DC_UP	24	0.16
	GSE45365_WT_VS_IFNAR_KO_CD11B_DC_UP	18	0.20
	GSE22886_NAIVE_VS_MEMORY_TCELL_DN	32	0.20
	GSE40274_FOXP3_VS_FOXP3_AND_LEF1_TRANSDUCED_ACTIVATED_CD4_TCELL_UP	25	0.21
	GSE32986_UNSTIM_VS_CURDLAN_LOWDOSE_STIM_DC_DN	28	0.32
NonSmoker	**GSE25123_WT_VS_PPARG_KO_MACROPHAGE_IL4_STIM_DN**	**23**	**0.01**
	GSE32986_UNSTIM_VS_CURDLAN_LOWDOSE_STIM_DC_DN	28	0.13
	GSE32986_GMCSF_VS_GMCSF_AND_CURDLAN_LOWDOSE_STIM_DC_UP	24	0.21

Table 8: GO enrichment results for the redundancy-based ranking of graph $\mathcal{G}$ for Smoker and Nonsmoker subgroups. Gene pathways with q-value < 0.05 are bolded.

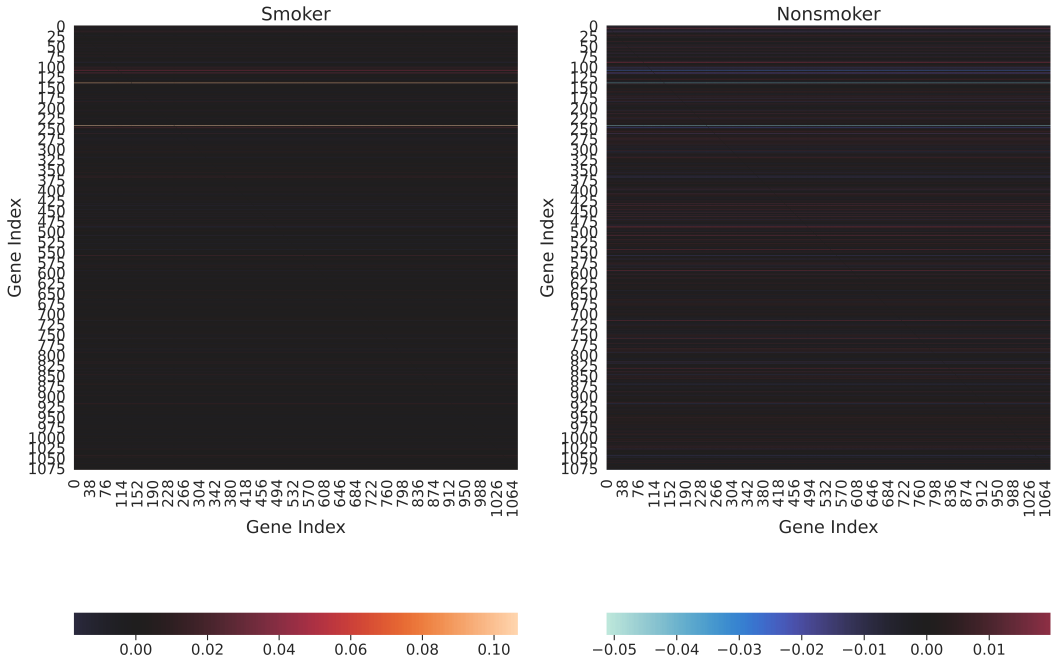

Figure 8: Adjacency matrix for graph $\mathcal{G}$, averaged over Smoker and Nonsmoker subgroups and displayed as a heatmap.

REFERENCES

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and Sabine Süsstrunk. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, November 2012. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2012.120.

Yoav Arnson, Yehuda Shoenfeld, and Howard Amital. Effects of tobacco smoke on immunity, inflammation and autoimmunity. *Journal of autoimmunity*, 34(3):J258–J265, 2010.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

Jørgen Bang-Jensen and Gregory Z Gutin. *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.

Roderick Bloem, Harold N Gabow, and Fabio Somenzi. An algorithm for strongly connected component analysis in n log n symbolic steps. *Formal Methods in System Design*, 28(1):37–56, 2006.

Thomas Bonald, Nathan de Lara, Quentin Lutz, and Bertrand Charpentier. Scikit-network: Graph analysis in python. *Journal of Machine Learning Research*, 21(185):1–6, 2020. URL http://jmlr.org/papers/v21/20-412.html.

Howard D Bondell and Brian J Reich. Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar. *Biometrics*, 64(1):115–123, 2008.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 883–892. PMLR, 10–15 Jul 2018. URL http://proceedings.mlr.press/v80/chen18j.html.

Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, August 2016. doi: 10.1145/2939672.2939785.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. URL http://arxiv.org/abs/1406.1078.

Smita Chormunge and Sudarson Jena. Correlation based feature selection with clustering for high dimensional data. *Journal of Electrical Systems and Information Technology*, 5(3):542–549, 2018.

Ian Covert, Scott Lundberg, and Su-In Lee. Feature removal is a unifying principle for model explanation methods. *CoRR*, abs/2011.03623, 2020a. URL https://arxiv.org/abs/2011.03623.

Ian Covert, Scott Lundberg, and Su-In Lee. Understanding global feature contributions through additive importance measures. *arXiv preprint arXiv:2004.00668*, 2020b.

Ian Covert, Scott M Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 17212–17223. Curran Associates, Inc., 2020c. URL https://proceedings.neurips.cc/paper/2020/file/c7bf0b7c1a86d5eb3be2c722cf2cf746-Paper.pdf.

Tianyu Cui, Pekka Marttinen, and Samuel Kaski. Learning global pairwise interactions with bayesian neural networks. *arXiv preprint arXiv:1901.08361*, 2019.

Anupam Datta, Shayak Sen, and Yair Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pp. 598–617. IEEE, 2016.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Alexandre Duval and Fragkiskos D Malliaros. Graphsvx: Shapley value explanations for graph neural networks. *arXiv preprint arXiv:2104.10482*, 2021.

E. Fehrman, A. K. Muhammad, E. M. Mirkes, V. Egan, and A. N. Gorban. The Five Factor Model of personality and evaluation of drug consumption risk. June 2015.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric, 2019. URL http://arxiv.org/abs/1903.02428. cite arxiv:1903.02428.

Christopher Frye, Colin Rowat, and Ilya Feige. Asymmetric shapley values: incorporating causal knowledge into model-agnostic explainability. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1229–1239. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/0d770c496aa3da6d2c3f2bd19e7b9d6b-Paper.pdf.

Michel Grabisch and Marc Roubens. An axiomatic approach to the concept of interaction among players in cooperative games. *International Journal of game theory*, 28(4):547–565, 1999.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Joseph D Janizek, Pascal Sturmfels, and Su-In Lee. Explaining explanations: Axiomatic feature interactions for deep networks. *Journal of Machine Learning Research*, 22(104):1–54, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL http://arxiv.org/abs/1412.6980.

Jon M Kleinberg. Hubs, authorities, and communities. *ACM computing surveys (CSUR)*, 31(4es): 5–es, 1999.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*, 2002.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent Individualized Feature Attribution for Tree Ensembles. pp. 9.

Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):56–67, 2020.

Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *arXiv preprint arXiv:2011.04573*, 2020.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Aria Masoomi, Chieh Wu, Tingting Zhao, Zifeng Wang, Peter Castaldi, and Jennifer Dy. Instance-wise feature grouping. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 13374–13386. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/9b10a919ddeb07e103dc05ff523afe38-Paper.pdf.

W Meeusen and L Cuyvers. Clique detection in directed graphs: A new algorithm. *Journal of Computational and Applied Mathematics*, 1(3):185–203, 1975.

Mark Newman. *Networks*. Oxford university press, 2018.

Guillermo Owen. Multilinear extensions of games. *Management Science*, 18(5-Part-2):64–79, 1972. URL https://EconPapers.repec.org/RePEc:inm:ormnsc:v:18:y:1972:i:5-part-2:p:64-79.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162.

Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models.

Elizabeth A. Regan, John E. Hokanson, James R. Murphy, Barry Make, David A. Lynch, Terri H. Beaty, Douglas Curran-Everett, Edwin K. Silverman, and James D. Crapo. Genetic epidemiology of COPD (COPDGene) study design. *COPD*, 7(1):32–43, February 2010. ISSN 1541-2563. doi: 10.3109/15412550903499522.

Michael Reich, Ted Liefeld, Joshua Gould, Jim Lerner, Pablo Tamayo, and Jill P. Mesirov. GenePattern 2.0. *Nature Genetics*, 38(5):500–501, May 2006. ISSN 1061-4036. doi: 10.1038/ng0506-500.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

Patrick Schwab and Walter Karlen. Cxplain: Causal explanations for model interpretation under uncertainty. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/3ab6be46e1d6b21d59a3c3a0b9d0f6ef-Paper.pdf.

Lloyd S Shapley. *17. A value for n-person games*. Princeton University Press, 2016.

Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2008. doi: 10.1017/CBO9780511811654.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pp. 3145–3153. PMLR, 2017.

Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014.

Agnieszka Strzelak, Aleksandra Ratajczak, Aleksander Adamiec, and Wojciech Feleszko. Tobacco smoke induces and alters immune responses in the lung triggering inflammation, allergy, asthma and other lung diseases: a mechanistic review. *International journal of environmental research and public health*, 15(5):1033, 2018.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pp. 3319–3328. PMLR, 2017.

Mukund Sundararajan, Kedar Dhamdhere, and Ashish Agarwal. The shapley taylor interaction index. In *International Conference on Machine Learning*, pp. 9259–9268. PMLR, 2020a.

Mukund Sundararajan, Kedar Dhamdhere, and Ashish Agarwal. The Shapley Taylor Interaction Index. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 9259–9268. PMLR, November 2020b.

R. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1:146–160, 1972.

Michael Tsang, Dehua Cheng, and Yan Liu. Detecting statistical interactions from neural network weights. *arXiv preprint arXiv:1705.04977*, 2017.

Michael Tsang, Youbang Sun, Dongxu Ren, and Yan Liu. Can i trust you more? model-agnostic hierarchical explanations. *arXiv preprint arXiv:1812.04801*, 2018.

Michael Tsang, Sirisha Rambhatla, and Yan Liu. How does this interaction affect me? interpretable attribution for feature interactions. *arXiv preprint arXiv:2006.10965*, 2020.

Jiaxuan Wang, Jenna Wiens, and Scott Lundberg. Shapley flow: A graph-based approach to interpreting model predictions. In *International Conference on Artificial Intelligence and Statistics*, pp. 721–729. PMLR, 2021.

David C Yang and Ching-Hsien Chen. Cigarette smoking-mediated macrophage reprogramming: mechanistic insights and therapeutic implications. *Journal of nature and science*, 4(11), 2018.

Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32:9240, 2019.

Mustafa Kemal Yöntem, Kemal Adem, Tahsin İlhan, and Serhat Kiliçarslan. DIVORCE PREDICTION USING CORRELATION BASED FEATURE SELECTION AND ARTIFICIAL NEURAL NETWORKS. *Nevşehir Hacı Bektaş Veli Üniversitesi SBE Dergisi*, 9(1):259–273, June 2019.

Jinsung Yoon, James Jordon, and Mihaela van der Schaar. Invase: Instance-wise variable selection using neural networks. In *International Conference on Learning Representations*, 2018.

Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *CoRR*, abs/2012.15445, 2020. URL https://arxiv.org/abs/2012.15445.

Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. *arXiv preprint arXiv:2102.05152*, 2021.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer, 2014.

Xiangrong Zeng and Mário AT Figueiredo. Solving oscar regularization problems by fast approximate proximal splitting algorithms. *Digital Signal Processing*, 31:124–135, 2014.

Hao Zhang, Yichen Xie, Longjie Zheng, Die Zhang, and Quanshi Zhang. Interpreting multivariate shapley interactions in dnns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 10877–10886, 2021.