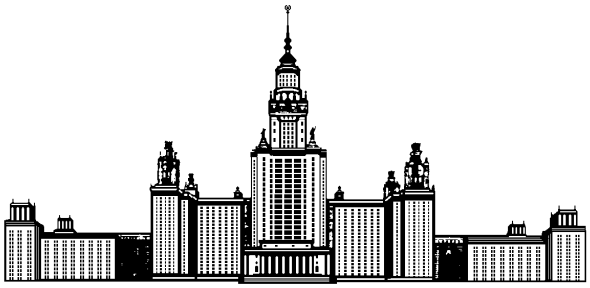

Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Кузнецов Михаил Константинович

Важность признаков

КУРСОВАЯ РАБОТА

Научный руководитель:
д.ф.-м.н., профессор
А. Г. Дьяконов

Москва, 2021

§1. Введение

С течением времени модели, помогающие решать непростые задачи, становятся всё сложнее и сложнее. Иногда нам важно не только, как хорошо решена задача с точки зрения качества, но и умение объяснить полученные ответы. Большое количество параметров и нелинейных связей внутри являются главной причиной плохой интерпретации. Существует три наиболее известные категории *важности признаков* (feature importance): filter, embedded, wrapper.

Filter методы опираются на знание о самих данных, например, коэффициенты корреляции, взаимная информация. Поиск взаимосвязи между признаками и целевой переменной является основной задачей.

Embedded методы используют внутреннее представление модели. Примерами могут служить веса модели, information gain в деревьях. Существенным недостатком является ограниченность их применения, выигрышем — более конкретное представление о степени взаимодействия признаков.

Wrapper методы — наиболее общий способ определения важности, так как он не зависит от устройства модели (model-agnostic), а только от ее ответов. Shapley values находят значимость, исходя из индивидуального вклада признака, входящего в подмножество исходных «фич».

Mixed методы — смесь вышеперечисленных. В основном, нейросетевые.

§2. Постановка задачи

Пусть $(\mathbf{X}, \mathbf{Y}) = (\mathbf{X_1}, \mathbf{X_2}, ..., \mathbf{X_p}, \mathbf{Y})$ — случайный вектор, $(x, y) = (x_1, x_2, ..., x_p, y)$ — его реализация. Совокупность упорядоченных x_i формирует эмпирический аналог признака — X_i, а упорядоченных y — Y. Множество значений переменной Z равно $rng(Z)$.
Тогда $\mathcal{X} = rng(X_1) \times rng(X_2) \times \ldots \times rng(X_p)$, $\mathcal{Y} = rng(Y)$. Допустим мы обучаем модель $\mathbf{a}$. Тройка $(\mathbf{X}, \mathbf{Y}, \mathbf{a})$ формирует конкретную ситуацию. Обозначим за $\mathcal{S}$ набор из всевозможных таких троек.

Тогда задача выглядит в общем случае следующим образом: необходимо задать функцию $\phi_i : \mathcal{S} \to \mathbb{R},\ i = 1, \ldots, p$. Назовём ее значение на конкретной тройке *важностью i-ого признака*. На практике, мы не знаем распределения признаков, поэтому тройка $(\mathbf{X}, \mathbf{Y}, \mathbf{a})$ заменяется на $(X, Y, \mathbf{a})$.

§3. Методы

3.1. Filter методы

Методы данной группы работают с данными непосредственно. Широко известный пример — *линейный коэффициент корреляции Пирсона*:

$$\rho_{\mathbf{X}_i, \mathbf{Y}} = \frac{\mathbb{E}[\mathbf{X}_i \mathbf{Y}] - \mathbb{E}[\mathbf{X}_i]\mathbb{E}[\mathbf{Y}]}{\sqrt{\mathbb{E}\left[\mathbf{X}_i^2\right] - (\mathbb{E}[\mathbf{X}_i])^2}\sqrt{\mathbb{E}\left[\mathbf{Y}^2\right] - (\mathbb{E}[\mathbf{Y}])^2}} = \phi_i$$

Если две переменные сильно коррелируют с $\mathbf{Y}$, но $\mathbf{Y}$ в действительности зависит только одной, стоит воспользоваться ранговым аналогом. Такие коэффициенты только измеряют степень линейной (неранговые) или (ранговые) монотонной зависимости.

Другой подход, позволяющий уловить нелинейную связь — *взаимная информация*:

$$I(X; Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} P_{X,Y}(x, y) \log_2 \frac{P_X(x) P_Y(y)}{P_{X,Y}(x, y)}$$

$$I(X;Y \mid Z) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}}\sum_{z\in\mathcal{Z}} P_{X,Y,Z}(x,y,z)\log_2 \frac{P_{X|Z}(x \mid z)P_{Y|Z}(y \mid z)}{P_{X,Y|Z}(x,y \mid z)}$$

Она имеет несколько полезных свойств:

- $I(X;Y) \geq 0$
- $I(X;Y \mid Z) = I(Y;X \mid Z)$
- $I(\mathbf{X};\mathbf{Y}) = 0$ тогда и только тогда, когда $\mathbf{X}$ и $\mathbf{Y}$ независимые случайные величины
- $I(X,Z;Y \mid W) = I(X;Y \mid W) + I(Z;Y \mid W,X)$

Хотя это является огромными плюсами для метода, всё же он не всегда хорош. Например, когда ошибка MSE распределена по Стьюденту алгоритм выбирает нелучшие с точки зрения MSE подмножества признаков [1].

3.2. Embedded методы

L_1 *регуляризация* является достаточно простым способом выявления «хороших» признаков. Однако с увеличением уверенности, что какой-то признак важный, уменьшается сложность модели.

В статье [5] авторы не прибегают к подобным «трюкам», а используют *среднее увеличение количества информации* (Mean Decrease Impurity):

$$\phi_m = \frac{1}{N_T}\sum_{T}\sum_{t\in T: v(s_t)=X_m} p(t)\Delta i\left(s_t,t\right)$$

где T — дерево, $p(t)$ — доля объектов дошедших до узла t, $\Delta i\left(s_t,t\right)$ — изменеие количества информации в узле t с разбиением s_t. Основные результаты представлены для *полностью рандомизированных и до конца построенных* (totally randomized and fully developed) деревьев. В частности, при бесконечно большой выборке категориальных данных справедливо:

$$\phi_m = \sum_{k=0}^{p-1}\frac{1}{C_p^k}\frac{1}{p-k}\sum_{B\in\mathcal{P}_k(V^{-m})} I\left(X_m;Y \mid B\right) \tag{1}$$

$$\sum_{m=1}^{p}\phi_m = I\left(X_1,\ldots,X_p;Y\right) \tag{2}$$

Формула (1) даёт нам полноценное представление о зависимости признака и целевой переменной. Здесь присутствует разложение как по мощности множества взаимодействия с другими признаками (сумма по k), так и по её степени (сумма по подмножествам). Оказывается если ограничить глубину деревьев до $q \leq p$, важность будет равна первым q слагаемым из первой суммы в (1). Разбиение наглядно визуализируется на Рис. 1. Можно заметить, что некоторые признаки становятся неважными в присутствии других. В случае случайного леса признаки X_2, X_5 находились вверху дерева. Это привело к «скрытым эффектам»: часть признаков вносят свой вклад только при обуславливании с X_2, X_5.

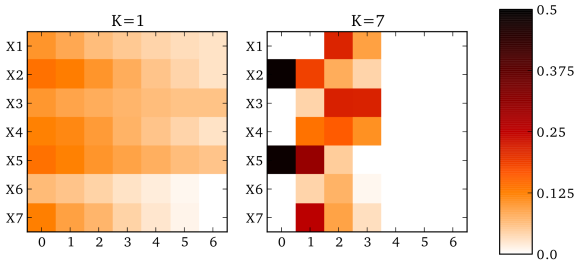

Рис. 1: Важность признаков в зависимости от мощности множества, на котором она обуславливается. Значение в клетке (i, j) — j-ое слагаемое из первой суммы в (1) для X_i. На картинке слева используется обычное дерево, справа — случайный лес [5]

Это также было замечено в [9], где авторы предложили использовать имплементированный ими метод построения *условных деревьев* (conditional trees [ctree]). Выбор переменной при расщеплении в узле осуществляется путем минимизации значения p-критерия независимости условного вывода, сравнимого, например, с тестом χ^2 со степенью свободы, равной числу категорий признака. Gini importance также сильна уязвима к сэмплированию с возвратом. Возможным решением может стать *перестановочная важность* (permutation importance). О ней пойдёт речь в секции 3.3.

3.3. Wrapper методы

Наиболее простым и эффективным методом является перестановочная важность. Для её вычисления необходимо сравнить выход модели на двух выборках: исходной и перемешенной по интересующему признаку. Такой подход сохраняет маргинальное распределение и прост в вычислении, однако при сильнокореллированных признаках он склонен занижать значимость, в частности, в случае аддитивной регрессионной модели [3]. Данный подход можно развить и на группу признаков, как это сделано в статье [2]:

$$\phi_J = \mathbb{E}\left[\left(\mathbf{Y} - f\left(\mathbf{X}_{(J)}\right)\right)^2\right] - \mathbb{E}\left[(\mathbf{Y} - f(\mathbf{X}))^2\right] = R(f, \mathbf{X}_{(J)}) - R(f, \mathbf{X})$$

$$\hat{\phi_J} = \frac{1}{N_T} \sum_T \left[\hat{R}\left(T, oob(\hat{\mathbf{X}}_{(J)})\right) - \hat{R}\left(T, oob(\hat{\mathbf{X}})\right)\right]$$

где $\hat{}$ обозначает эмпирический аналог, $\mathbf{X}_{(J)}$ — случайный вектор, полученный заменой в $\mathbf{X}$ случайных признаков $\mathbf{X}_J$ на их независимую от $\mathbf{Y}$ и оставшихся признаков копию. В случае аддитивной регрессионной модели, ϕ_J пропорционален дисперсии ответов на соответствующем подмножестве признаков. С помощью *графика частичной зависимости* (PDP) это можно наглядно увидеть.

Рассмотрим один из самых популярных методов построения важности. Пусть у нас есть некоторая характеристическая функция $v : 2^N \to \mathbb{R}$ такая, что $v(\varnothing) = 0$. Будем искать определение важности удовлетворяющее следующим свойствам:

- *сумма в конечный ответ* (efficiency): $\sum_{i \in N} \phi_i(v) = v(N)$
- *аддитивность* (additivity): $\forall v, w : \phi(v + w) = \phi(v) + \phi(w)$, где $(v + w)(S) = v(S) + w(S)$ $\forall S$
- *симметрия* (symmetry): Если $v(S \cup \{i\}) = v(S \cup \{j\})$ $\forall S$, где $S \subset N$ и $i, j \notin S$, тогда $\phi_i(v) = \phi_j(v)$
- *корректность* (dummy): Если $v(S \cup \{i\}) = v(S)$ $\forall S$, где $S \subset N$ и $i \notin S$, тогда $\phi_i(v) = 0$

Оказывается аддитивность и симметричность вместе эквивалентна *согласованности* (consistency) [6]: Если $\forall v, w; \forall S : i \notin S$ выполнено $v(S \cup \{i\}) - v(S) \geq w(S \cup \{i\}) - w(S)$, тогда $\phi_i(v) \geq \phi_i(w)$. Существует единственная важность, удовлетворяющая данным требования. Это *Shapley values*:

$$Sh_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{1}{n} \frac{1}{\binom{n-1}{|S|}} (v(S \cup \{i\}) - v(S)), \quad i = 1, \ldots, n$$

В данном случае учитывается вклад i-ого признака во всевозможные подмножества других. В [4] рассматривается другой подход задать такие значения, помогающий избежать экспоненциальной сложности:

$$\phi_i = \frac{1}{n! \cdot |\mathcal{X}|} \sum_{O \in \pi(N)} \sum_{y \in \mathcal{X}} \left[f\left(\tau\left(x, y, \mathrm{Pre}^i(O) \cup \{i\}\right)\right) - f\left(\tau\left(x, y, \mathrm{Pre}^i(O)\right)\right)\right]$$

$$\tau(x, y, S) = (z_1, z_2, \ldots, z_n), \quad z_i = \begin{cases} x_i & ; \quad i \in S \\ y_i & ; \quad i \notin S \end{cases}$$

где $\pi(N)$ — множество упорядоченных перестановок длины N, $\mathrm{Pre}^i(O)$ — множество индексов, которые стоят перед i в $O \in \pi(N)$. Авторы считают $65000 \cdot p$ итераций совместного сэмплирования перестановки и элемента выборки достаточным для аппроксимации с ошибкой 0.01 для 99% переменных.

Рассмотрим некоторые нейросетевые определения значимости признаков. Метод DeepLift [8] основан на разделении отрицательного и положительного вклада в «таргет». Таким образом возможно избежать некоторые проблемы, связанные с обнулением градиентов и отсутствием изменения ответов модели при перестановке входов. Пусть есть начальное значение x_{m0}, y_0. Положим $\Delta y = y - y_0, \Delta x_m = x_m - x_{m0}$, тогда:

$$\phi_m = m_{\Delta x_m \Delta y} \Delta x_m$$

где $m_{\Delta x_m \Delta y} = const$. Выполняются свойства:

- *Сумма в дельту* (summation to delta):

$$\sum_{i=1}^{p} \phi_i = \Delta y$$

- *Цепное правило* (chain rule):

$$m_{\Delta x_i \Delta y} = \sum_j m_{\Delta x_i \Delta z_j} m_{\Delta z_j \Delta y}$$

Внутренние состояния пересчитываются через цепное правило и специальное определение мультипликаторов, которое учитывает появление «знаковых Δ» в присутсвии или наличии Δ другого знака. Данный подход решил часть проблем, но некоторые всё же остаются, например, трасформация коэффициентов при проходе через max_pool слой.

Если посчитать для части модели коэффициенты, используя Shapley values, получим DeepShap [6].

Shapley values можно апроксимировать с помощью линейной регрессии, если взять MSE лосс с определенным ядром. Тогда мы получим так называемый Kernel SHAP [6]. Он сходится гораздо быстрее в отличии от простого сэмплирования Shapley values.

Другой подход связан с построением так называемой *объясняющей модели* (explanation model). В CXplain [7] используется Granger's определение причинности взаимосвязи между признаками и целевой переменной, в котором:

- все признаки релевантные
- признак временно предшествует метке, то есть для того, чтобы получить метку, нужна информация о признаке

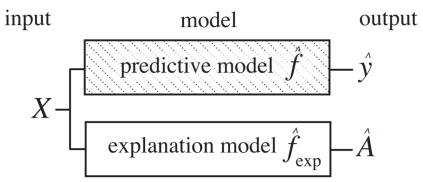

Рис. 2: Важность в CXPlain. Объясняющая модель $\hat{f}_{\text{exp}}$ обучается выдавать важность признаков $\hat{A}$ для исходной модели $\hat{f}$ [7]

Важность определяется как нормированная разница ошибок объясняемой модели на маскированных данных и исходных. У объясняющей модели:

- цель — предсказать важность признаков
- вход — элемент из выборки
- лосс — расстояние Кульбака — Лейблера между истинным и предсказанным распределениями важности признаков

Заметим, что данная модель нужна когда нет истинных меток объектов. Для устойчивости авторы обучают ансамбль обучающих моделей на сэмплированных выборках. Итоговая важность — медиана предсказаний ансамбля, а точность — интерквартильный размах. В таком подходе точность оценки важности коррелирует с ошибкой ранжирования важности признаков. Даже при небольшой мощности ансамбля хорошо оценивается точность. Сильной стороной данного метода является быстрота, что как мы помним оказалась краеугольным камнем в Shapley values.

3.4. Mix методы

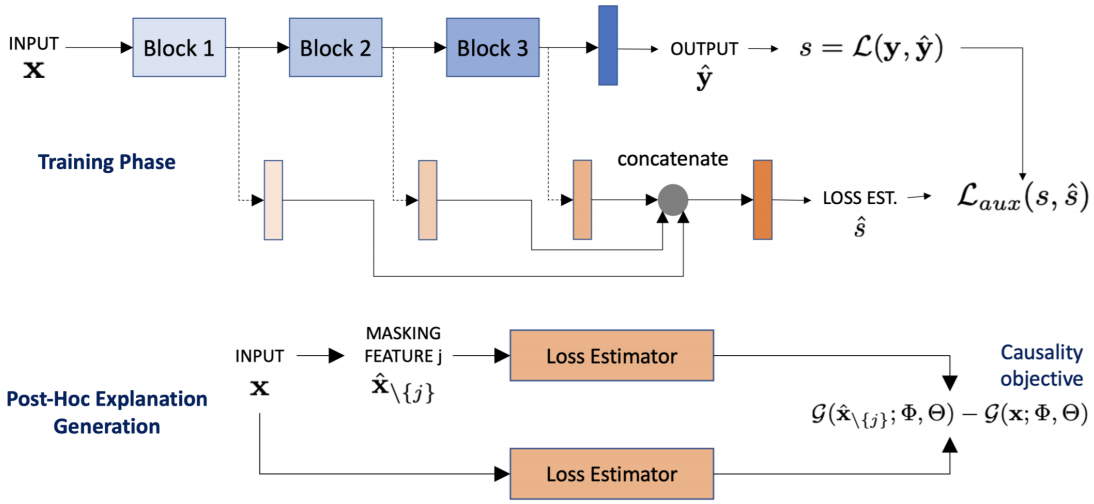

Рис. 3: Схема работы алгоритма получения важности в PRoFILE. Синим цветом отмечена исходная модель, а коричневым — объясняющая [10]

Использование не только выходов исходной модели, а также её внутреннее представление и знание о лоссе дают возможность построить «хорошую» объясняющую модель.

В PRoFILE (см. рис. 3) у нее:

- цель — научиться предсказывать лосс основной сети
- вход — латентные представления после некоторых слоёв основной сети, за каждым из которых следует линейный слой
- лосс — $\sum_{(i,j)} \max\left(0, -\mathbb{I}\left(s_i, s_j\right) \cdot \left(\hat{s}_i - \hat{s}_j\right) + \gamma\right)$,
  где $\mathbb{I}\left(s_i, s_j\right) = \begin{cases} 1 & \text{если } s_i > s_j \\ 0 & \text{иначе} \end{cases}$, $s = \mathcal{L}_{mainnet}(y, a), \hat{s} = \mathcal{L}_{expnet}(s, \hat{s})$

Стоит заметить, что градиент от ошибки объясняющей модели влияет на слои в основной. Таким образом, мы тренируем модели совместно. В отличии от Shap, CXplain и Lime данный подход устойчив к «помехам» в данных: на датасетах Cifar10-C, MNIST-USPS PRoFILE оказался лучше по метрике:

$$\Delta \text{ log-odds } = \text{ log-odds } \left(p_{\text{ref}}\right) - \text{ log-odds } \left(p_{\text{masked}}\right)$$

где $\text{log-odds}\ (\text{p}) = \log\left(\frac{\text{p}}{1-\text{p}}\right)$, p_{ref} — вероятность, полученная на оригинальных данных, а p_{masked} — на маскированных.

Рассмотрим другой подход. Допустим мы знаем заранее важность скольких признаков хотим найти, например, s штук. Тогда логичным способом получения таких ϕ_i может стать обучение нейронной сети, которая выдает нам набор переменных, входящих в интересующее множество.

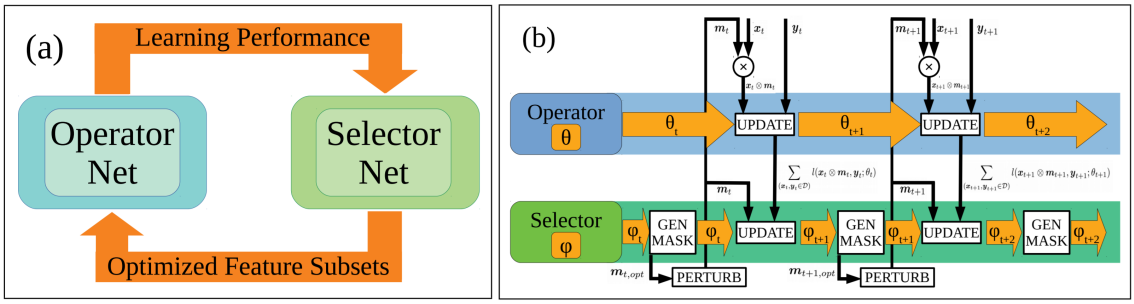

Рис. 4: Схема работы алгоритма получения важности в FIR [11]

В одном из FIR методов (см. рис. 4) обучение происходит поочередно. Оно сочетает в себе два этапа, где маски для признаков — бинарные вектора (1 - берем признак, 0 - нет):

1. operator net генерирует обучающую выборку для selector net: пары масок и соответствующий лосс на них
2. selector net передаёт operator net'у следующие «хорошие» маски:
   (a) лучшая маска с предыдущей итерации
   (b) маска, получаемая результатом следующего алгоритма:
      i. стартуем с маски $\boldsymbol{m}_0 = \left(\frac{1}{2}, \cdots, \frac{1}{2}\right)$, выбираем топ s компонент градиента selector net'а. Валидируем полученную «оптимальную» маску:
         A. берем топ s компонент градиента только теперь уже в точке, равной полученной маске. Таким образом получаем две маски: $\boldsymbol{m}_{\text{opt}}$ — содержит s единиц, $\overline{\boldsymbol{m}}_{\text{opt}}$ — содержит $d - s$ единиц.
         B. заменяем компоненту маски $\boldsymbol{m}_{\text{opt}}$ с отрицательным градиентом на компоненту с наибольшим градиентов в маске $\overline{\boldsymbol{m}}_{\text{opt}}$
         C. проверяем условие $f_S\left(\boldsymbol{m}_{opt}\right) \leq f_S\left(\boldsymbol{m}'_{opt}\right)$, где $\boldsymbol{m}'_{\text{opt}}$ получена заменой компоненты маски $\boldsymbol{m}_{opt}$ с наименьшим градиентом на компоненту маски $\overline{\boldsymbol{m}}_{\text{opt}}$ с наибольшим. Если это условие не выполнено повторяем A.-B.

(c) полученная $\boldsymbol{m}_{\text{opt}}$ на самом деле может быть неоптимальной, поэтому добавляем небольшую случайность: случайно выберем $s_{rand} < s$ компонент в $\boldsymbol{m}_{\text{opt}}$ /$\overline{\boldsymbol{m}}_{\text{opt}}$, инвертируем их и поменяем значения местами с другой маской. Сделаем так несколько раз.

В итоге у operator net:

- цель — обучение с учителем конкретной задачи
- вход — x и маска признаков
- лосс — соответствующий задаче

А у selector net:

- цель — предсказать loss operator net
- вход — маска признаков
- лосс — MSE с лоссом, переданным от operator net

Важность признака — соответствующая компонента градиента лосса selector net'а в точке оптимального набора признаков. Процесс построения оптимального набора очень долгий, но как показали результаты экспериментов качество среди DFS, RF, RFE оказалось лучшим, как и качество выбранных признаков. Однако время работы несравнимо больше: x440 дольше RF и x2 дольше Lime.

§4. Эксперименты

§5. Заключение

§6. Список литературы

[1] Frenay B Doquire G, V. M. Is mutual information adequate for feature selection in regression / Verleysen M Frenay B, Doquire G. — 2012.

[2] Gregorutti B Michel B, S.-P. P. Grouped variable importance with random forests and application to multiple functional data analysis / Saint-Pierre P Gregorutti B, Michel B. — 2015.

[3] Gregorutti B Michel B, S.-P. P. Correlation and variable importance in random forests / Saint-Pierre P Gregorutti B, Michel B. — 2017.

[4] I, K. An efficient explanation of individual classifications using game theory / Kononenko I. — 2010.

[5] Louppe G Wehenkel L, S. A. Understanding variable importances in forests of randomized trees / Sutera A Louppe G, Wehenkel L, Geurts P. — 2013.

[6] Lundberg S, L. S. A unified approach to interpreting model predictions / Lee S Lundberg S. — 2017.

[7] Schwab P, K. W. Cxplain: Causal explanations for model interpretation under uncertainty / Karlen W Schwab P. — 2019.

[8] Shrikumar A Greenside P, K. A. Learning important features through propagating activation differences / Kundaje A Shrikumar A, Greenside P. — 2017.

[9] Strobl C Boulesteix A-L, Z. A. Bias in random forest variable importance measures: Illustrations, sources and a solution / Zeileis A Strobl C, Boulesteix A-L, Hothorn T. — 2007.

[10] Thiagarajan JJ Narayanaswamy V, A. R. B. P.-T. S. A. Accurate and robust feature importance estimation under distribution shifts / Anirudh R Bremer P-T Spanias A Thiagarajan JJ, Narayanaswamy V. — 2020.

[11] Wojtas M, C. K. Feature importance ranking for deep learning / Chen K Wojtas M. — 2020.