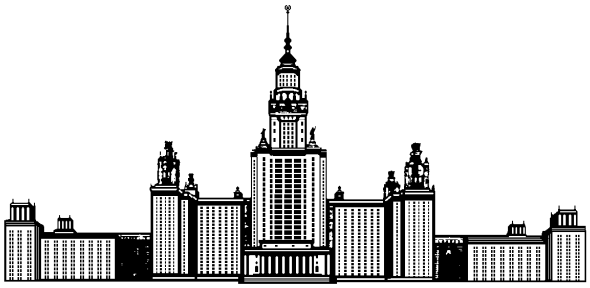


Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

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 являются главной причиной плохой интерпретации. Существует три наиболее известные категории *важности признаков* (feature importance): filter, embedded, wrapper.

Фильтр-методы (filter) опираются на знание о самих данных, например, коэффициенты корреляции, взаимная информация.

Встроенные методы (embedded) 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.

Оберточные методы (wrapper) — 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».

Смешанные методы (mixed) — 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. Выборка обозначается, как $(x^{(i)}, y^{(i)})_{i=1}^{n}$.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 в общем случае выглядит 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 $(X, Y, \mathbf{a})$.

Естественно интерес заключается в нахождении важности, которая:

- выделяет релевантные признаки
- быстро считается
- дает дополнительную информацию об устройстве модели
- устойчива к шуму в данных

Разработать универсальный метод, удовлетворяющий вышеперечисленным свойствам еще не удалось. Рассмотрим существующие кандидаты.

§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 от одной, стоит воспользоваться ранговым аналогом. Такие коэффициенты измеряют степень только 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 [2] 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.

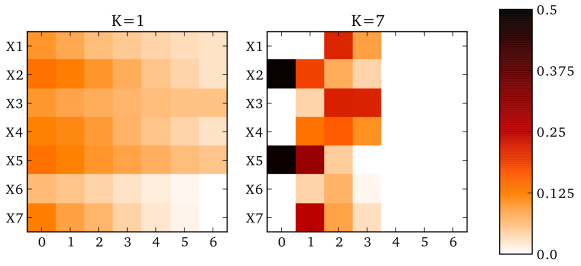


Рис. 1: Важность признаков в зависимости от мощности множества, на котором она обуславливается. Значение в клетке (i, j) — j-ое слагаемое из первой суммы в (1) для X_i. На картинке слева используется обычное дерево, справа — случайный лес [2]

Это также было замечено в [3], 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 [4]. Данный подход можно развить и на группу признаков, как это сделано в статье [5]:

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

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

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.

Частичная зависимость (partial dependence) может и сама быть инструментом для подсчета важности.

$$\mathrm{PD}^{\mathrm{IndSet}}(x) = \mathbb{E}\left[\mathbf{a}(z) \mid z_{\mathrm{IndSet}} = x_{\mathrm{IndSet}}\right], \quad \mathrm{IndSet} \subseteq \{1, 2, ..., p\}$$

В pyBreakDown [6] авторы хотят найти такие признаки, чтобы при их небольшом возмущении прогноз модели существенно изменился. Это они добиваются следующим образом. Начальное множество $\mathrm{IndSet} = \varnothing$. На каждой итерации алгоритма в/из IndSet добавляется/удаляется один признак. В первом варианте он максимизирует $\left|\mathrm{PD}^{\mathrm{IndSet}}(x) - \mathrm{PD}(x)\right|$

(*step-up*), во втором минимизирует (*step-down*). В итоге получается последовательность признаков с убывающей/увеличивающейся важностью, так как она определяется, как разность между соответствующими $\mathrm{PD}(x)$ на соседних IndSet.

Можно сделать предположение, что важность признака больше, если он сильнее взаимодействует с другими. Это можно формально выразить через метрику

$$H_j^2 = \sum_{i=1}^{n} \left[\mathbf{a}\left(x^{(i)}\right) - \mathrm{PD}^{\{j\}}\left(x^{(i)}\right) - \mathrm{PD}^{\{-j\}}\left(x^{(i)}\right) \right]^2 / \sum_{i=1}^{n} \mathbf{a}^2\left(x^{(i)}\right)$$

Она положительная и равна 0 тогда и только тогда, когда признак не взаимодействует с другими. Однако, как показано в разделе 4.1.2, это не очень подходящее определение значимости.

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) [7]: 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 [8] рассматривается другой подход задать такие значения, помогающий избежать экспоненциальной сложности:

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

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

где $\pi(p)$ — 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 [9] основан на разделении отрицательного и положительного вклада в целевую переменную. 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 $m_{\Delta x_m \Delta y}$, используя Shapley values, получим DeepShap [7].

Shapley values можно апроксимировать с помощью линейной регрессии, если взять MSE лосс с определенным ядром. Тогда мы получим так называемый Kernel SHAP [7]. Он сходится гораздо быстрее в отличии от простого сэмплирования Shapley values.

Другой подход связан с построением так называемой *объясняющей модели* (explanation model). В CXplain [10] 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

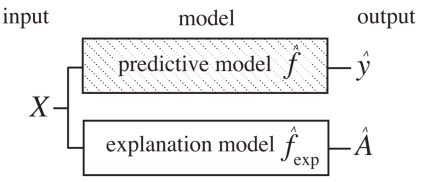


Рис. 2: Важность в CXPlain. Объясняющая модель $\hat{f}_{\text{exp}}$ обучается выдавать важность признаков $\hat{A}$ для исходной модели $\hat{f}$ [10]

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

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

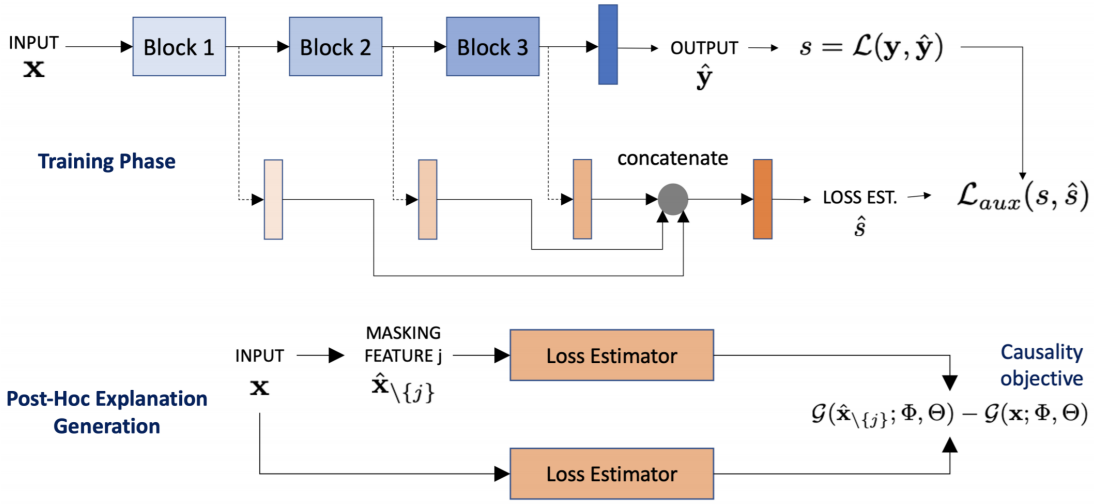


Рис. 3: Схема работы алгоритма получения важности в PRoFILE. Синим цветом отмечена исходная модель, а коричневым — объясняющая [11]

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.

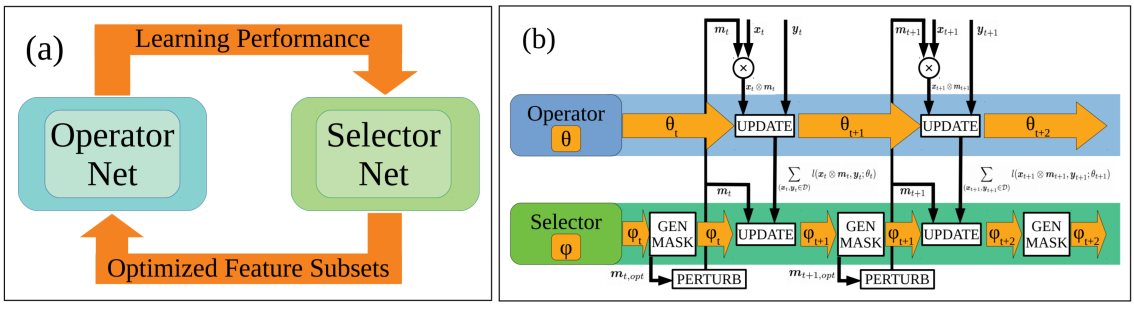


Рис. 4: Схема работы алгоритма получения важности в FIR [12]

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

4.1. Коррелированные признаки

4.1.1 Данные

Рассмотрим задачу классификации. Признаки состоят из пяти групп (gr_1, ..., gr_5) и одного шумового (rand). Каждая группа состоит из трёх признаков: gr_{ij}, j = 1, 2, 3. Группы независимы от друг друга и сэмплированы из нормального распределения $\mathcal{N}_3(0, C_i)$, где

$$C_i = \begin{pmatrix} 1 & \rho_i & \rho_i \\ \rho_i & 1 & \rho_i \\ \rho_i & \rho_i & 1 \end{pmatrix}, \quad \rho_i = \frac{i}{6}$$

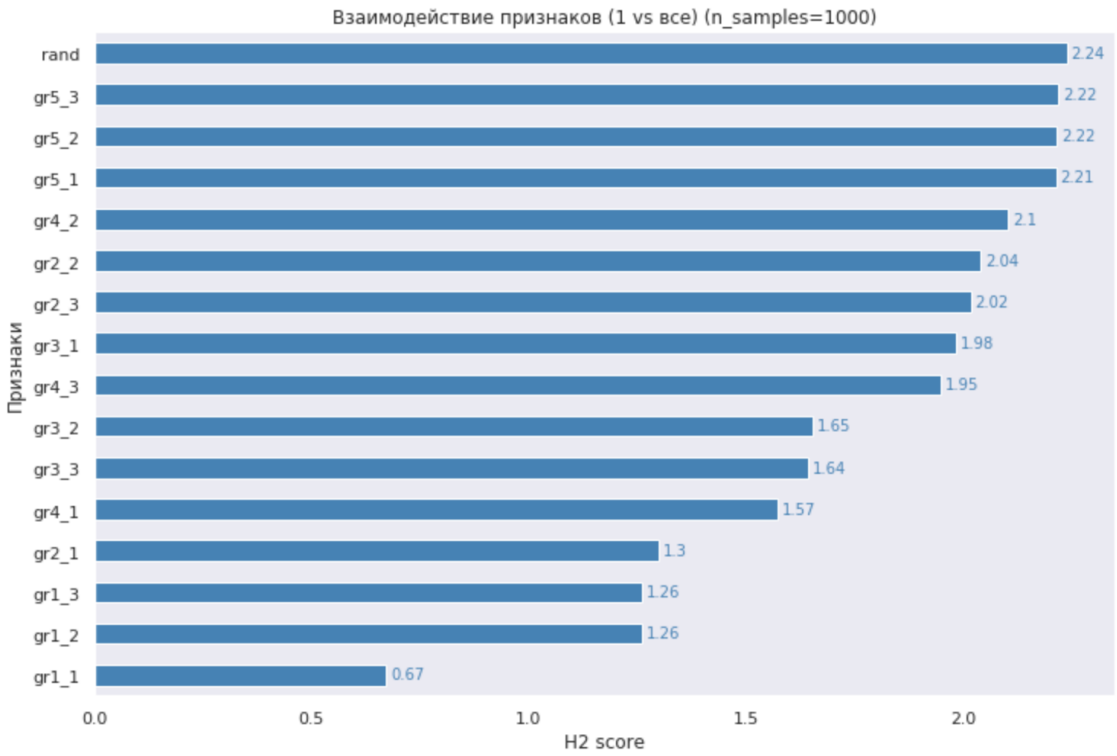
Шум был взят из распределения $\mathcal{N}_3(0, 1)$. Пусть $\boldsymbol{\xi}_i$ имеет биномиальное распределение $Bi(1, 0.5)$. Тогда целевая переменная y равна

$$\mathbb{I}[Z \geq \text{median}(Z)], \quad \text{где}$$
$$Z = \sigma(\xi_1 gr_{11} + ... + \xi_5 gr_{51})$$

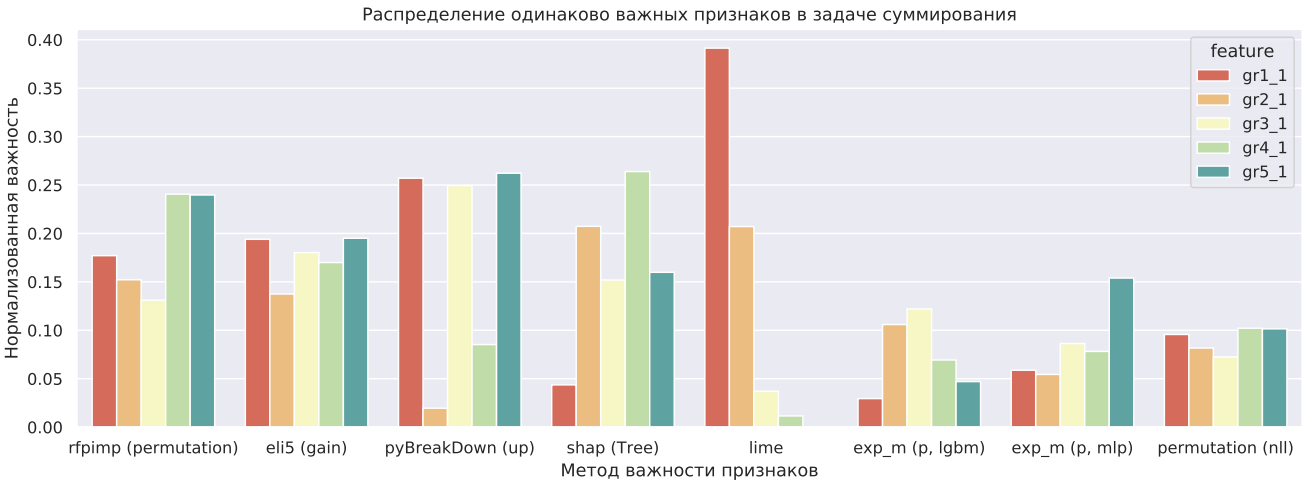
Количестов сэмплированных объектов равно 5000. В качестве моделей брались бустинг (LightGBM) и метод опорных векторов (SVM).

4.1.2 Важность

Посчитаем H_2 score для наших данных.



Видно, что рандомный признак оказался на первом месте. Чем больше групповая корреляции, тем больше взаимодействия получают признаки.



Наиболее равномерно распределены перестановочная, gain, Shapley, CXplain важности. Lime оставляет 2 наиболее важных признака, исходя из их независимости. Более подробную статистику можно посмотреть в A.1.

4.1.3 Удаление признаков

Рассмотрим задачу рекурсивного удаления признаков (RFE). На каждой итерации удаляется наименее важный признак и замеряется качество.

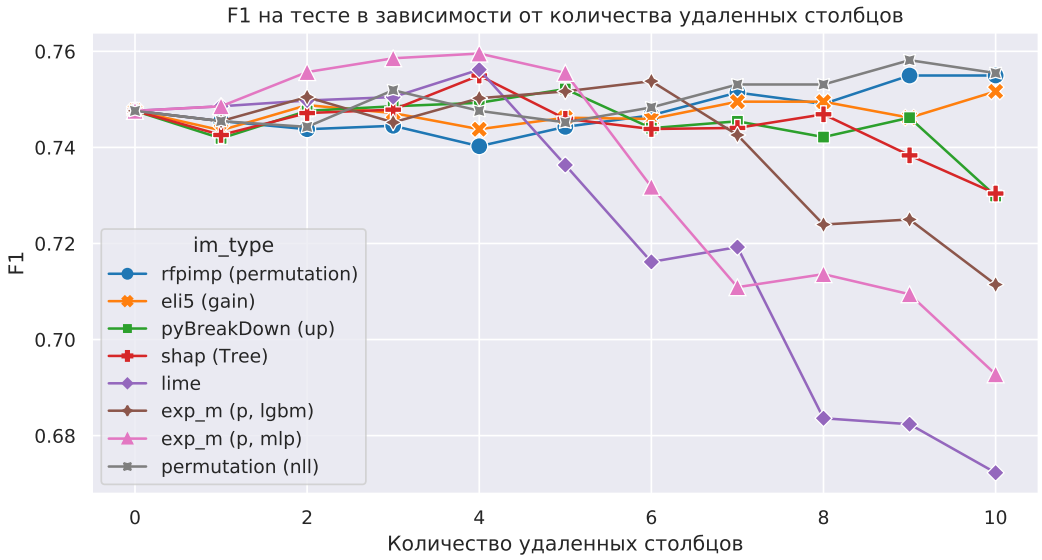
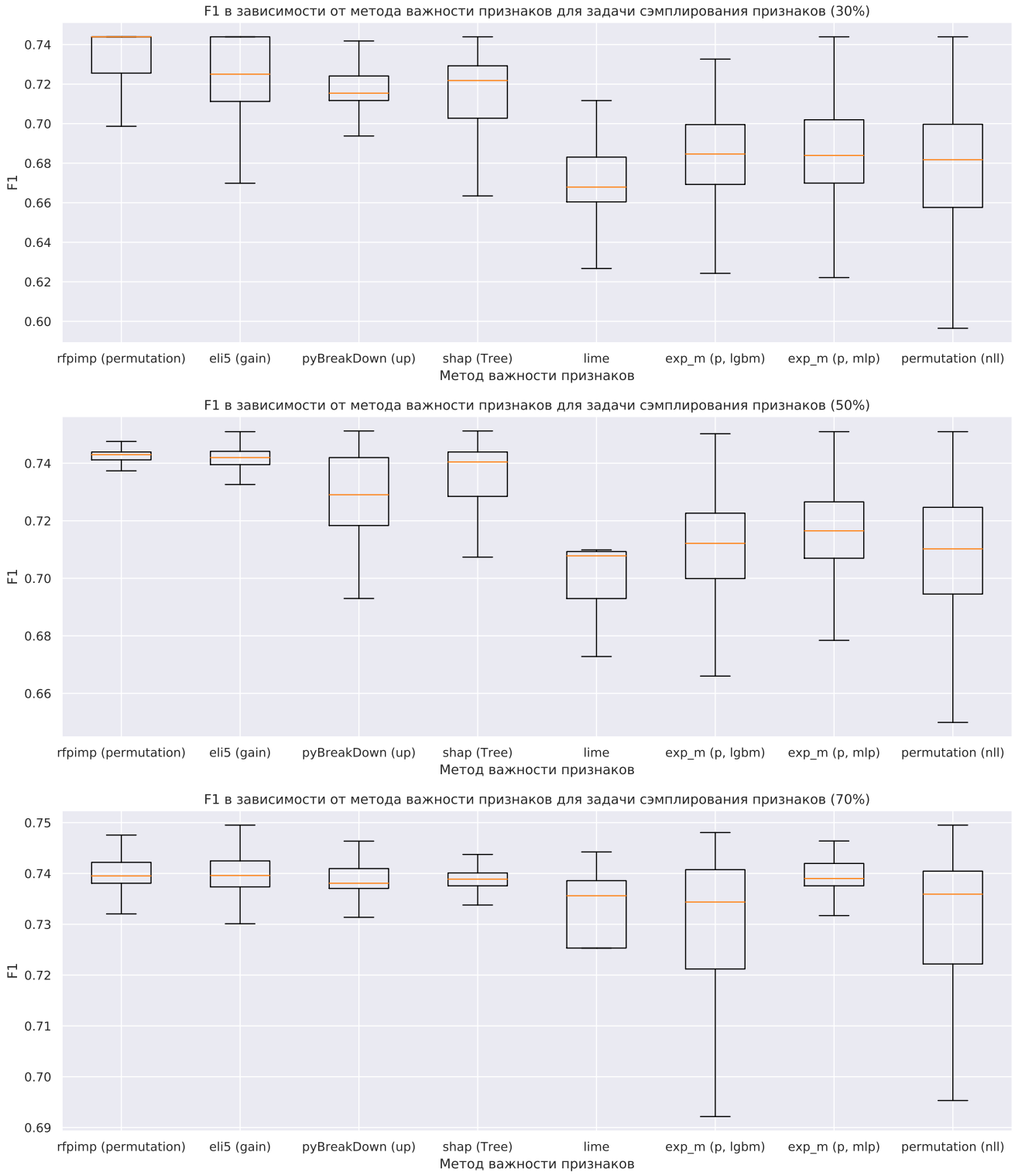


Рис. 5: Один запуск эксперимента RFE. Конечное количество признаков = 30% от исходного. Для вычисления shap сэмплировались 200 объектов из выборки.

Перестановочная и gain важность лучше всего обирают признаки «на будущее». Lime не учитывает зависимости второго и более порядков. Это может стать причиной выбрасывания относительно «хороших» признаков.

4.1.4 Сэмплирование признаков

Посмотрим, как важность хороша с точки зрения дальнешего обучения модели. Для каждого метода мы взяли нормализованный модуль важности и просэмплировали 1000 раз. После чего замерили качество (f1-метрика).



При большом количестве признаков перестановочная, gain, pyBreakDown, shap, exp_m(p, mlp) работают примерно одинаково. Однако при меньшем количестве признаков качество у pyBreakDown резко падает из–за итеративного алгоритма работы.

4.1.5 Копия признака

Выберем самый важный признак. Добавим его копию. На каждой итерации будем удалять наименее важный признак, за исключением выбранного и его копии. Обучать заново модель и повторять процедуру по достижению определенного количества оставшихся признаков

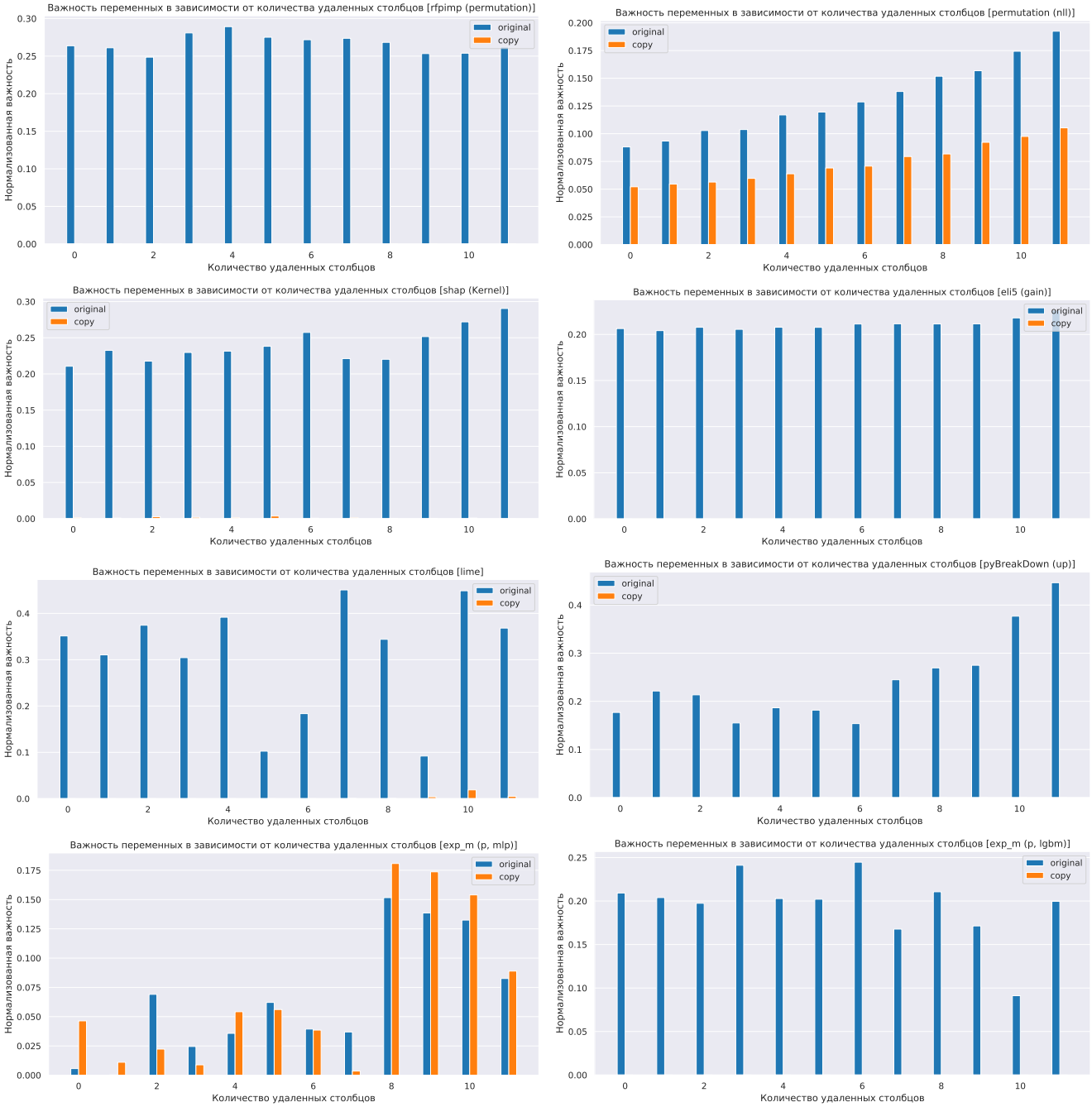


Рис. 6: Исходная модель LGBM. Конечное количество признаков= 30% от исходного. Для сэмплирования shap использовалось 100 объектов.

В большинстве случаев деревья не берут копию в качестве признака для разделения данных. Как следствие, значение копии равняется нулю. Однако с точки зрения (permutation (nll) вероятности положительного класса меняются.

Возьмём SVM в качестве модели.

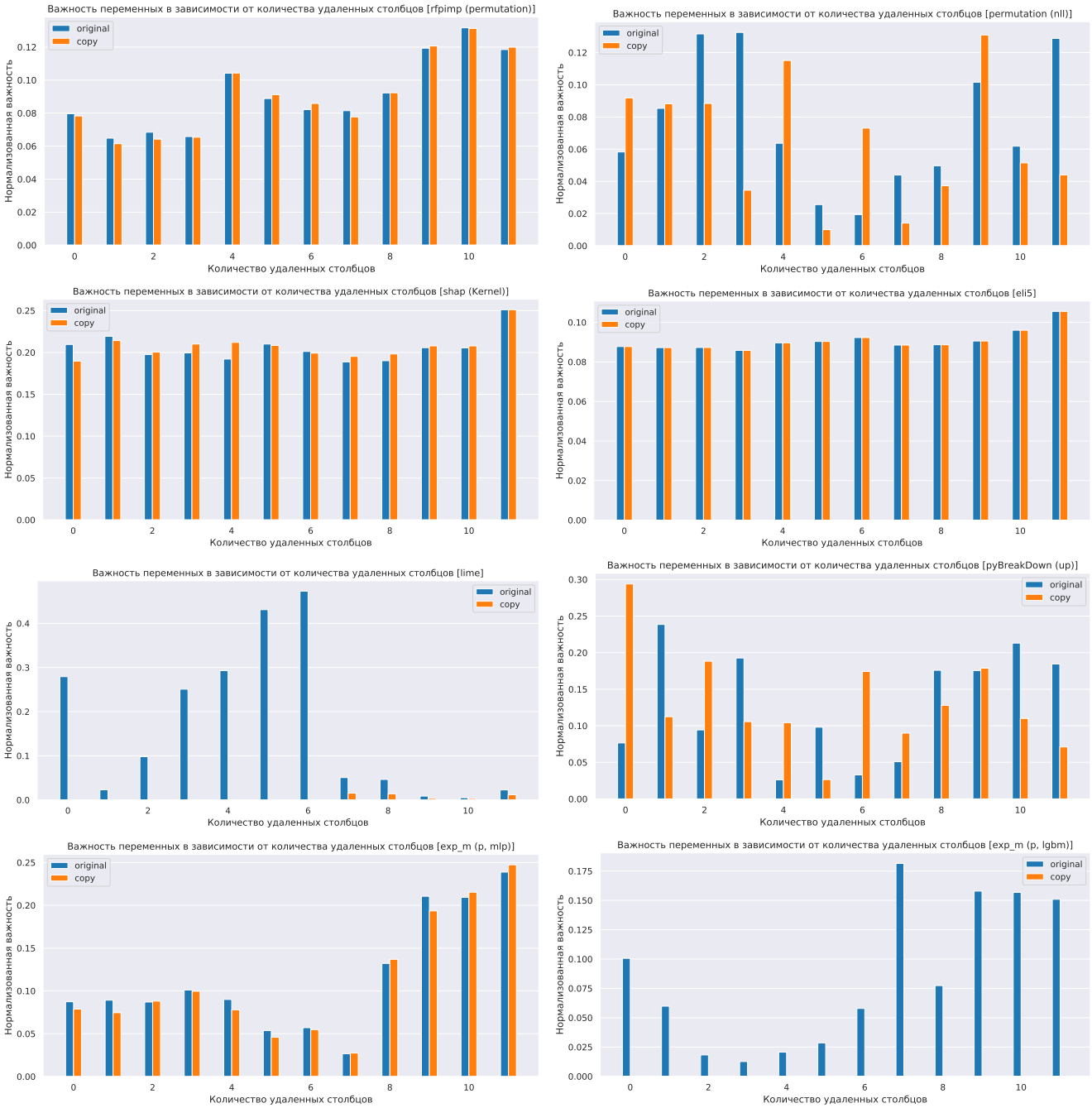


Рис. 7: Исходная модель SVM. Конечное количество признаков= 30% от исходного. Для сэмплирования shap использовалось 100 объектов.

Одинаковые важности дают rfpimp, eli5 (веса признаков), shap, exp_m (p, mlp). Использование линейного слоя нейронной сети в объясняющей модели помогло избавиться от асимметричного распределения важности.

4.2. Функции

Рассмотрим некоторые простые функции от многих переменных.

4.2.1 Данные

Признаки $X_1, ..., X_6, eps$ связаны с Y следующим образом:

$$Y_{reg} = X_1X_2X_3 + X_4 + X_5 + X_6 + eps, \quad \text{где } esp \sim N\left(0, 0.1^2\right), X_i \sim N(0,1)$$
$$Y_{clf} = \mathbb{I}[Z \geq \text{median}(Z)], \quad \text{где } Z = \sigma\left(Y_{reg} - \text{mean}\left(Y_{reg}\right)\right)$$

Количестов сэмплированных объектов равно 1000. В качестве модели использовался бустинг (LightGBM).

4.2.2 Стандартная классификация/регрессия

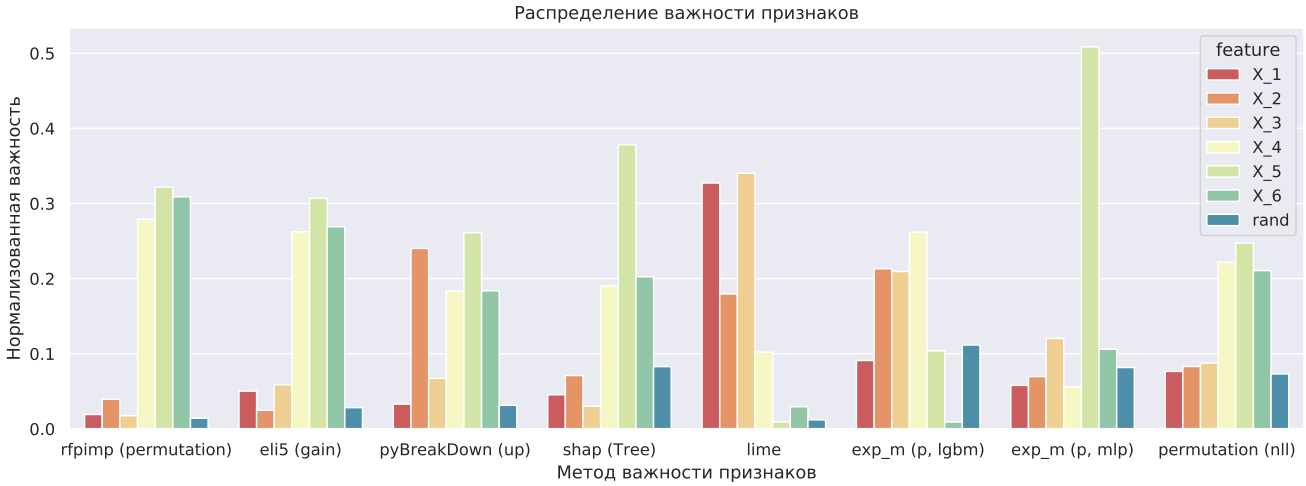


Рис. 8: $y = y_{clf}$

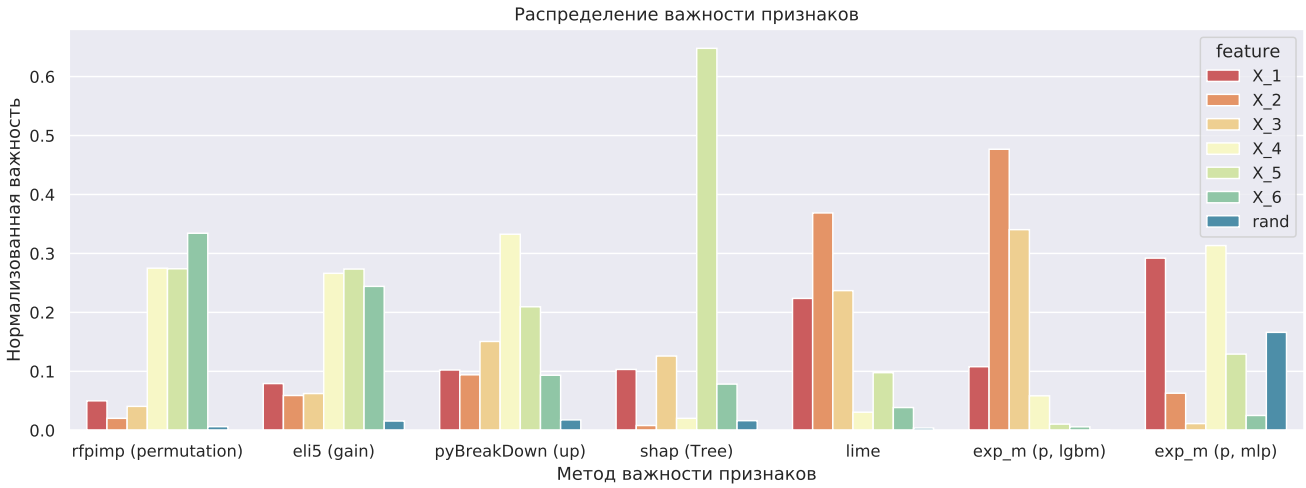


Рис. 9: $y = y_{reg}$

Наименьшую важность рандомному признаку дает rfpimp, lime.

4.2.3 Квадратная функция

Рис. 10: $y = y_{clf}\left(y^2_{\text{reg}}\right)$

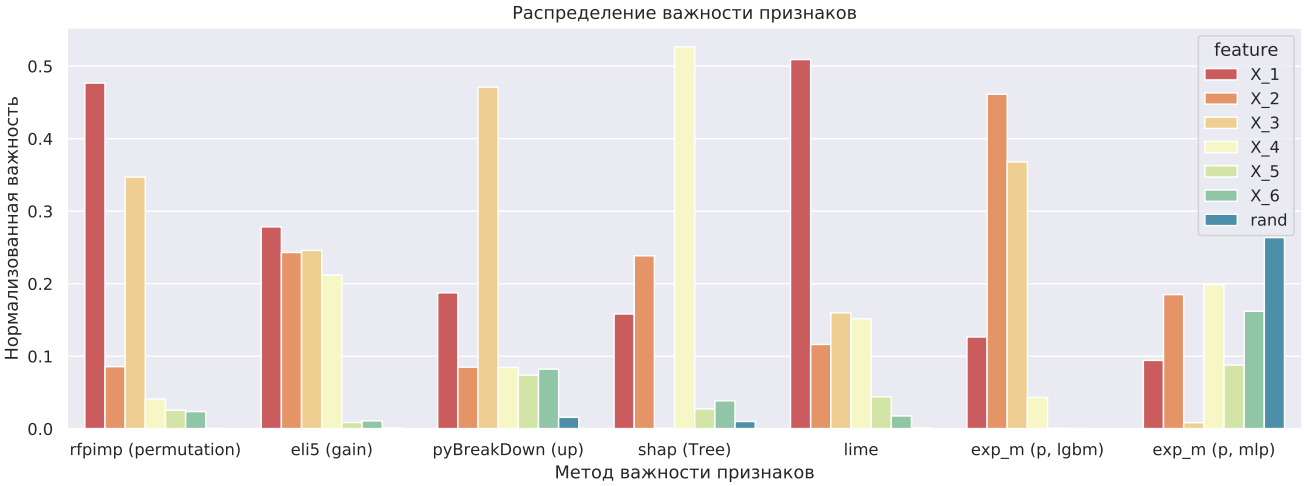


Рис. 11: $y = y^2_{reg}$

Признаки X_1, X_2, X_3 более важны для задачи регрессии так как имеют мультипликативный характер.

4.2.4 Степенная функция

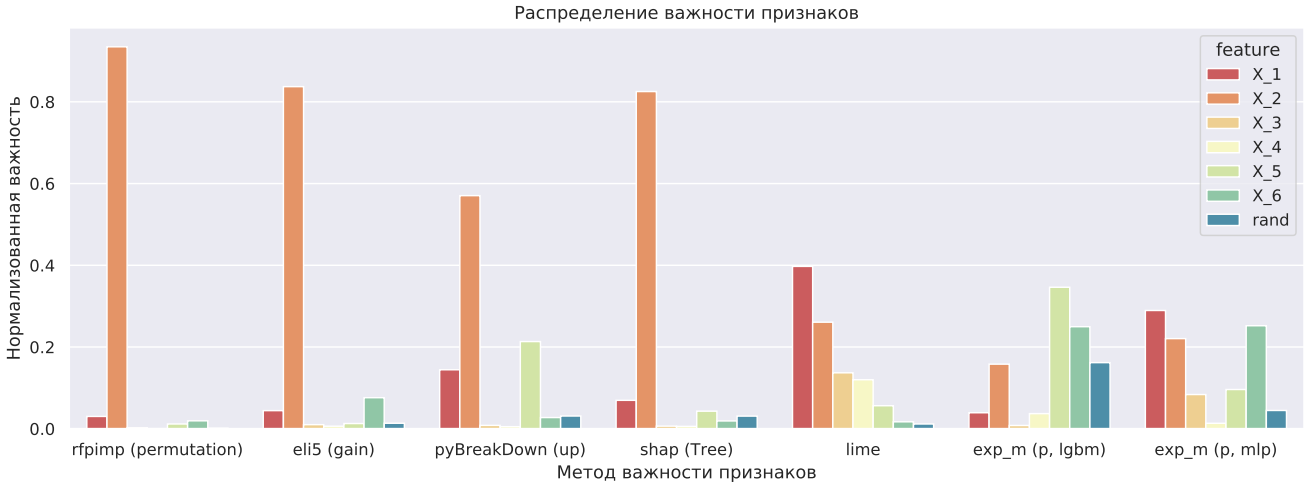


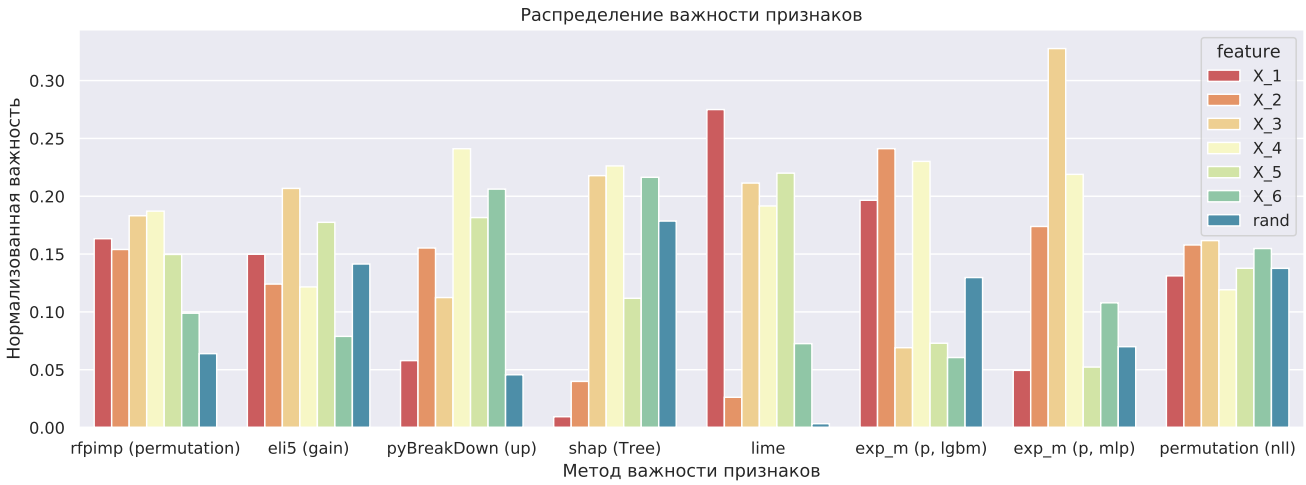
Рис. 12: $y = |X_1 + 10|^{(X_2+10)}$

Lime плохо обобщает из–за предположения независимости признаков. В большинстве случаев признак при степени более важен. Важность с использованием объясняющей модели показала плохие результаты. Большую роль играет сложность объясняющей модели. В данном примере ее оказалось недостаточно.

4.2.5 Сумма по модулю 2

Здесь

$$X_1, \ldots, X_6, X_7 \sim Bi(1, 0.5)$$
$$y = \text{Xor}\,(X_1, \ldots, X_6)$$



Shap дает большую важность рандомному признаку, так как в большом количестве подмножеств признаков является важным. Это может свидетельствовать о переобучении модели. Аналогичные рассуждения применимы и для permutation (nll). Заметим, что здесь использовалась валидационная выборка для оценивания качества.

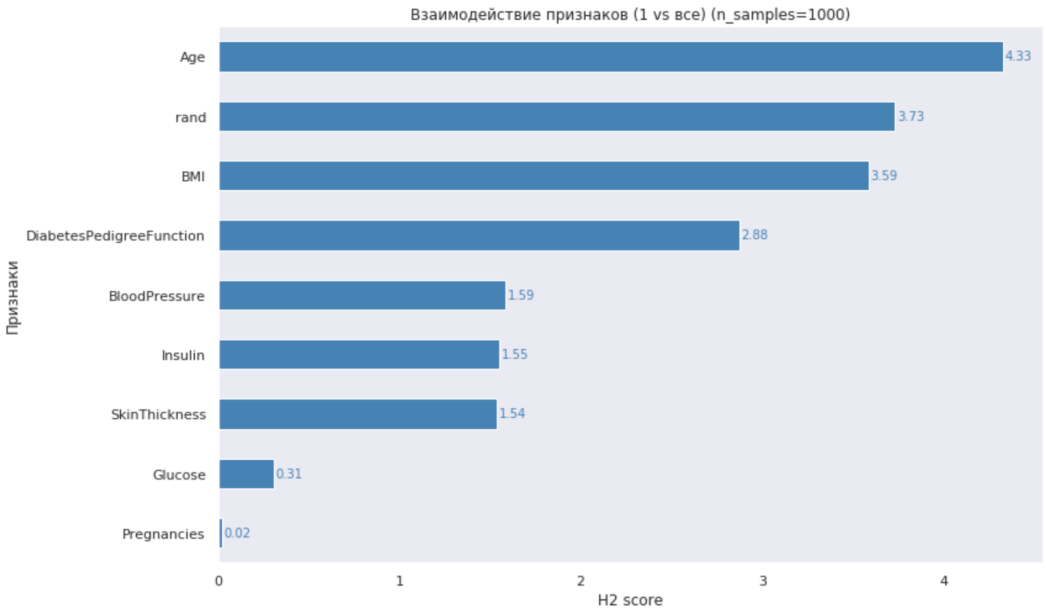
4.3. Прогноз диабета

4.3.1 Данные

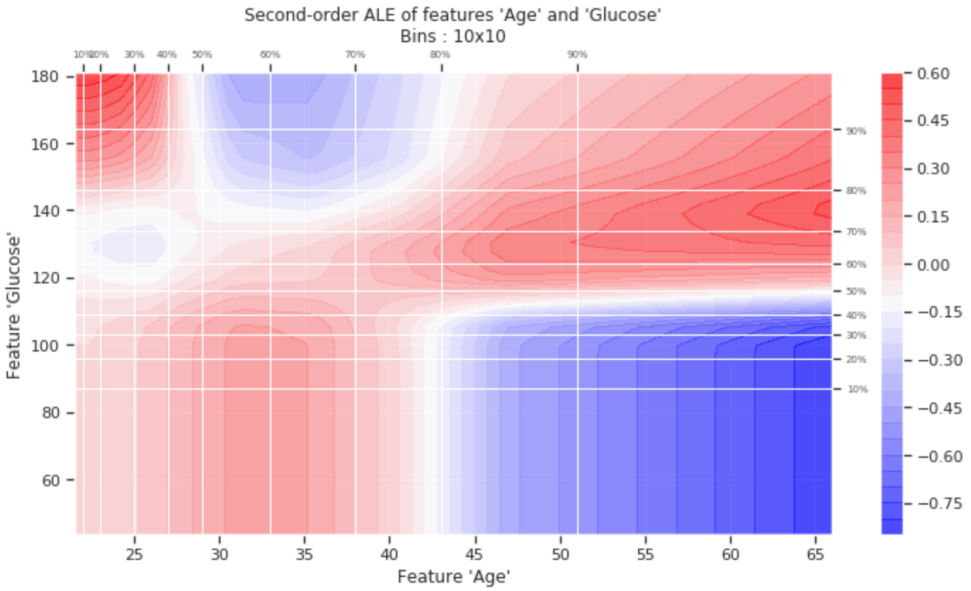
Датасет был взят с сайта kaggle. Признаки состоят из наиболее связанных с целевой меткой параметров: уровень глюкозы (Glucose), уровень инсулина (Insulin), функция родословной диабета (DiabetesPedigreeFunction) и так далее. Всего 9 штук. В качетсве модели использовались брались бустинг (LightGBM) и метод опорных векторов (SVM).

Повторим эксперименты секции 4.1.

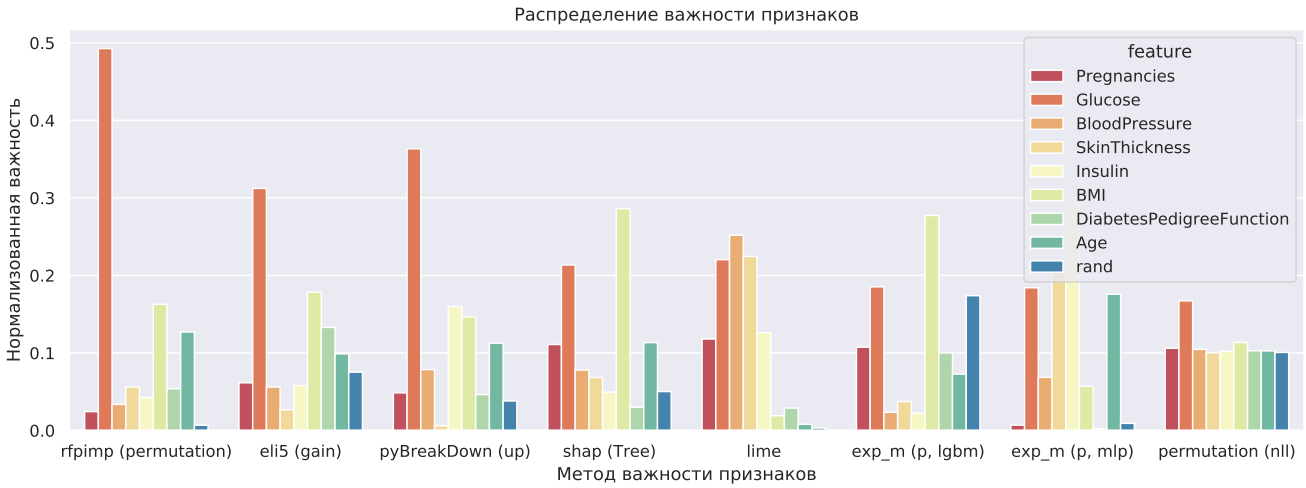
4.3.2 Важность



Логично увидеть возраст на первом месте.

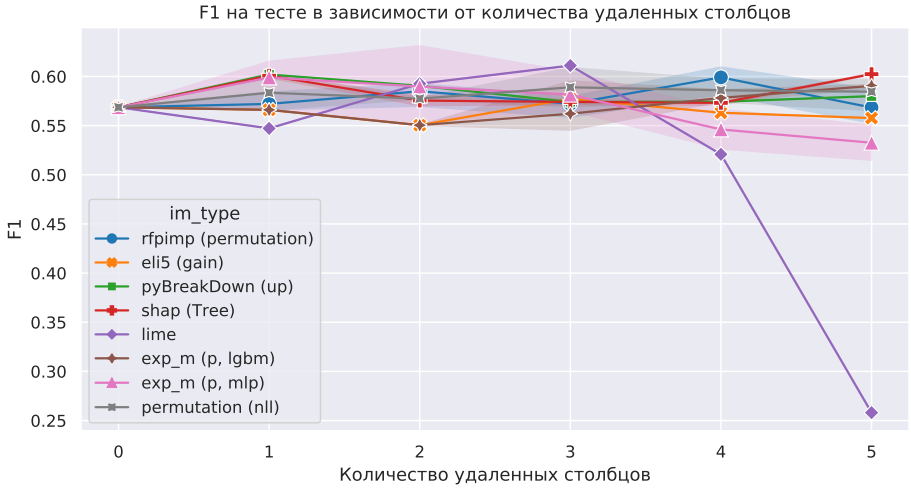


После 44 лет есть четкое пороговое значение по которому можно судить о диабете. При старении становится более важным соблюдать диету, так как небольшие отклонения приводят к резкому повышению вероятности того или иного случая.



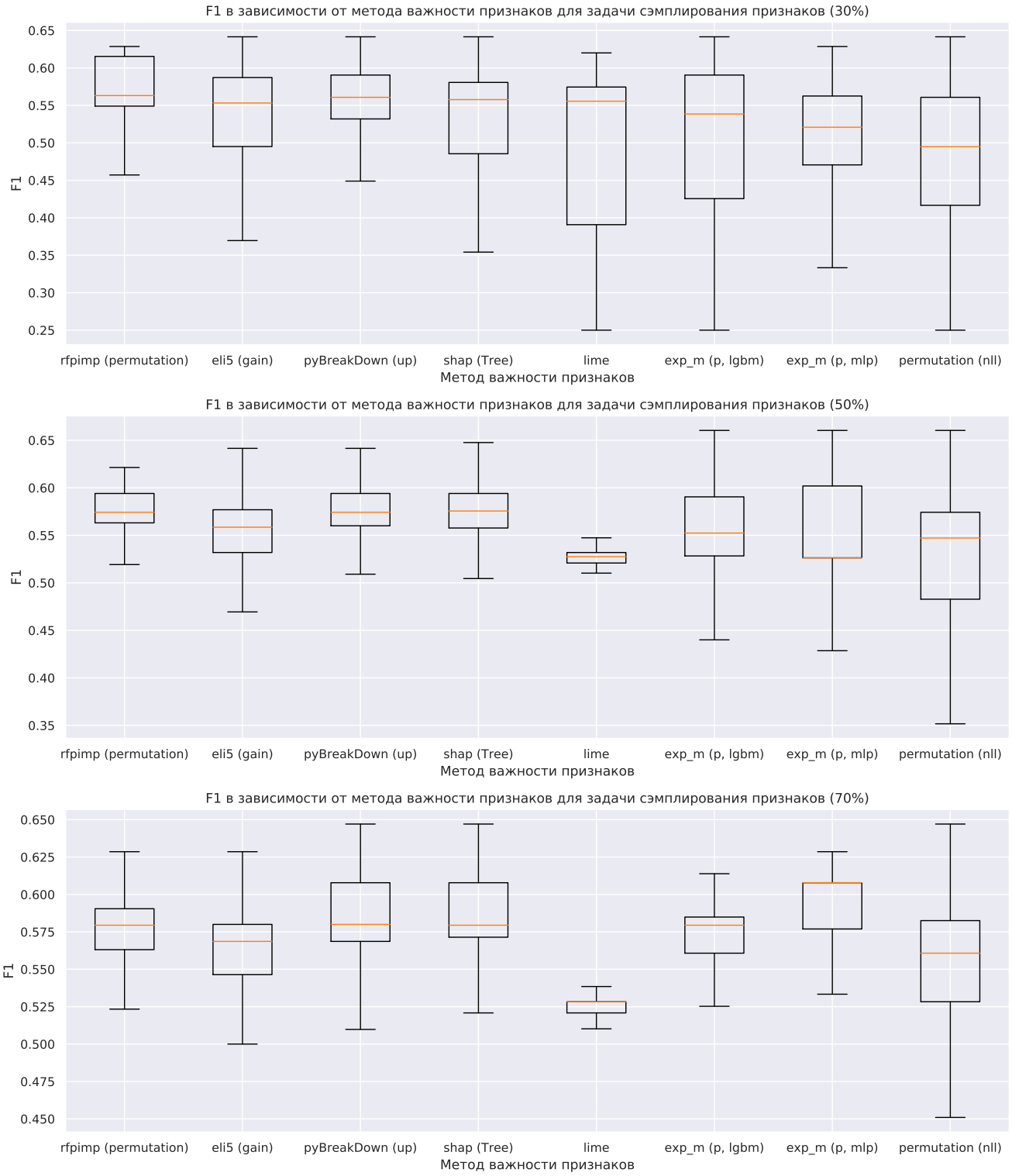
В большинстве случаем глюкоза и индекс массы тела влияют на наличие диабета. Gain показало высокую важность для рандомного признака. Большое число уникальных значений приводит к искусственному увеличению значимости. Permutation (nll) имеет невалидные значения.

4.3.3 Удаление признаков



На реальных данных все методы, за исключением lime, работают приблизительно одинаково.

4.3.4 Сэмплирование признаков



Перестановочная важность лучше всех выделяет признаки, которые в общем нужны для решения задачи. При большом количестве признаков объясняющая модель на основе многослойного перцептрона (MLP) дала лучшие результаты. За исключением lime, признаки выбранные алгоритмами дают в среднем одинаковое качество.

§5. Заключение

В данной работе рассмотрены различные методы определения важности. Проведены эксперименты, показывающие положительные и отрицательные стороны. Самая простая — перестановочная важность. Она является хорошей заменой более новым определениям. Одно из них включает дополнительную объясняющую модель. В эксперименте с сэмплированием признаков такой подход дал лучшие результаты. Однако для релевантных интерпретаций поиск хорошей архитектуры занимает большую часть времени. Для быстроты работы необходима gpu.

Локальные методы, например lime, плохо обобщают информацию на весь датасет.

Разделение на локальные и глобальные методы не производилось.

§6. Список литературы

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

[2] S. A. Louppe G, Wehenkel L and G. P, "Understanding variable importances in forests of randomized trees," 2013.

[3] Z. A. Strobl C, Boulesteix A-L and H. T, "Bias in random forest variable importance measures: Illustrations, sources and a solution," 2007.

[4] S.-P. P. Gregorutti B, Michel B, "Correlation and variable importance in random forests," 2017.

[5] M. B. Gregorutti B, "Grouped variable importance with random forests and application to multiple functional data analysis," 2015.

[6] B. P. Staniak M, "Explanations of model predictions with live and breakdown packages," 2018.

[7] L. S. Lundberg S, "A unified approach to interpreting model predictions," 2017.

[8] K. I, "An efficient explanation of individual classifications using game theory," 2010.

[9] K. A. Shrikumar A, Greenside P, "Learning important features through propagating activation differences," 2017.

[10] K. W. Schwab P, "Cxplain: Causal explanations for model interpretation under uncertainty," 2019.

[11] A. R. B. P.-T. S. A. Thiagarajan JJ, Narayanaswamy V, "Accurate and robust feature importance estimation under distribution shifts," 2020.

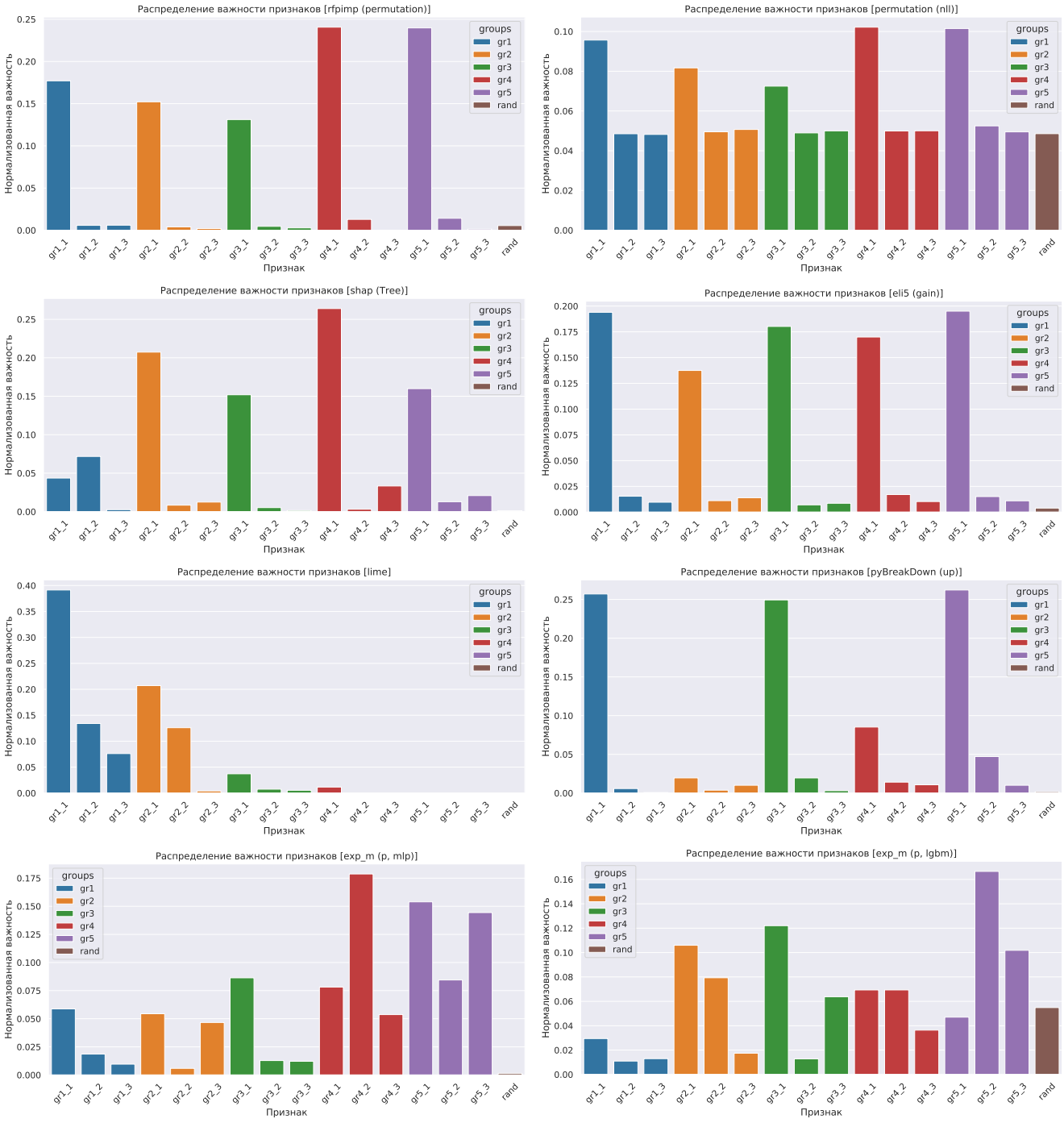
[12] C. K. Wojtas M, "Feature importance ranking for deep learning," 2020.

§A. Важность признаков

A.1. Дополнение к 4.1.2

В данном разделе:

$$y = \mathbb{I}[Z \geq \text{median}(Z)], \quad \text{где}$$
$$Z = \sigma(\xi_1 gr_{11} + ... + \xi_5 gr_{51})$$



Выбор оценочной функции для перестановочной функции важен. В данном случае f1 метрика оказалась более подходящей.

A.2. Дополнение к 4.3.2