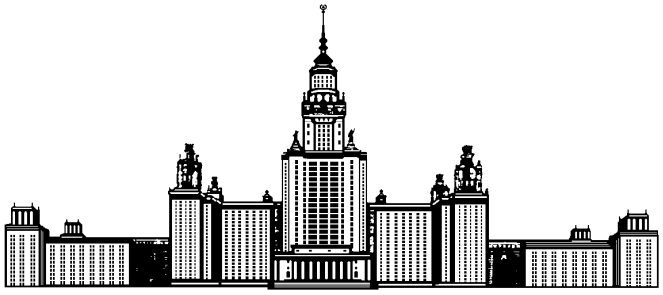

Московский государственный университет имени М. В. Ломоносова
Факультет Вычислительной Математики и Кибернетики
Кафедра Математических Методов Прогнозирования

Михеев Борис Михайлович

# Предобработка табличных данных для глубокого машинного обучения

# Preprocessing tabular data for deep machine learning

Выпускная квалификационная работа

**Научный руководитель:**
д. ф.-м. н., профессор
*Дьяконов Александр Геннадьевич*

Москва, 2023

# Содержание

**Аннотация**

В данной работе проводится подробный обзор существующих подходов к предобработке табличных данных для моделей глубокого машинного обучения. Проведен сравнительный анализ наиболее распространенных методов, рассмотрены их особенности, преимущества и недостатки, а также проведены вычислительные эксперименты с наиболее распространенными нейросетевыми архитектурами применительно к данным из различных задач и областей с целью более детального исследования.

# 1 Введение

В настоящее время модели глубокого обучения получили значительное развитие и успешно применяются во многих задачах. Различные полносвязные, сверточные и рекуррентные архитектуры, а также модели трансформеры способны хорошо выделять ключевые особенности и закономерности в данных, достигают выдающихся результатов в задачах обработки и генерации изображений, текста, звука, т. е. в применении к в определенной степени однородным, гомогенным данным, имеющим определенную структуру и подчиняющимся некоторым закономерностям. Однако при работе с данными в формате таблиц нейронные сети все еще испытывают трудности и иногда уступают классическим алгоритмам машинного обучения, например, градиентному бустингу над решающими деревьями[1, 2, 3, 4]. Табличные данные в основном весьма разнородны, состоят из разреженных дискретных категориальных и непрерывных числовых признаков в различном соотношении и с разным диапазоном значений, могут содержать пропуски, и в общем случае порядок столбцов признаков не определен, и найти и использовать взаимосвязи между ними сложнее, чем в случае, например, изображений и текстов, для которых определены понятия пространственной и семантической близости соответственно. Данные в виде таблиц являются наиболее распространенными в прикладных задачах, в финансовой, медицинской, управленческой, научной и многих других сферах. Глубокие нейронные сети являются значительно более мощными, гибкими и выразительными моделями в сравнении с классическими алгоритмами машинного обучения, и их применение к табличным данным перспективно и востребовано. Поэтому задача адаптации моделей глубокого обучения к данным в формате таблиц является актуальной.

Большинство существующих решений опирается на создание новой архитектуры, как правило с использованием уже зарекомендовавших себя идей (полносвязные архитектуры и модели трансформеры, использование механизмов внимания[5], прокидывания связей и т. д.), или является модификациями других моделей. Однако согласно исследованиям[6, 7, 2], для моделей глубокого обучения сами данные и их предобработка как правило имеют намного большее значение в случае таблиц, чем особенности устройства сети, и в предобработке ключевую роль играет способ пре-

образования исходных признаков, тогда как способы заполнения пропусков, обнаружения и исключения объектов-выбросов и т. п. довольно слабо влияют на результат работы модели. Нахождение оптимальных способов предобработки табличных данных для нейронных сетей с точки зрения итогового качества актуально во множестве прикладных и вспомогательных задач, задачах автоматического обучения, мультимодальных моделях и т. д.

В данной работе рассматриваются и сравниваются некоторые методы преобразования табличных данных для моделей глубокого машинного обучения, проводятся вычислительные эксперименты и исследования реализаций способов предобработки применительно к различным задачам, данным и нейросетевым архитектурам.

# 2 Постановка задачи

Рассмотрим объекты $x$ из множества $X$, задающиеся признаковым описанием $x_i = (x_{i1}, \ldots, x_{id})$, $x_i \in \mathbb{R}^d$. Будем рассматривать табличные данные, и в таком случае векторы признаковых описаний объектов будут являться строками таблиц. Среди признаков $x_{ij}$ могут присутствовать как непрерывные числовые признаки, которые могут иметь различные распределения и диапазоны значений, так и дискретные категориальные, которые могут иметь различный тип (число, символ, текст и т. д.) и число возможных значений. Пусть $y = (y_1, \ldots, y_n)^T$ – известные значения целевой переменной (метка класса или вещественное число) для объектов $x_1, \ldots, x_n$ из множества $X$. Пусть $D = \{x_i, y_i\}_{i=1}^n$ – обучающая выборка, $\mathcal{A}$ – алгоритм глубокого машинного обучения, обучаемый на выборке $D$. Пара вида $(D, \mathcal{A})$ будет задавать конкретную конфигурацию и конкретную итоговую модель в результате обучения. Пусть $M$ – множество всех таких пар $(D, \mathcal{A})$.

Цель задачи поиска оптимального способа предобработки табличных данных для нейросетевых моделей – найти отображение $f : M \to \mathbb{R}^{n \times d}$, наилучшее с точки зрения качества прогнозов, т. е. найти обучаемое преобразование входных данных для конкретной модели и набора данных, позволяющее достичь наилучшего значения некоторого показателя качества дальнейших предсказаний.

# 3 Основные проблемы и особенности решаемой задачи

В работах на тему применения нейронных сетей в задачах с табличными данными[1, 2, 3, 4, 6, 7] исследователи делают общий вывод о том, что во многих случаях модели глубокого обучения пока еще недостаточно хорошо справляются с данными задачами и часто уступают классическим моделям машинного обучения, в особенности ансамблям решающих деревьев и градиентному бустингу. Выделяют следующие основные причины этого и сопутствующие проблемы:

- Малые размеры наборов табличных данных по сравнению с наборами данных для других областей применения глубокого обучения (изображения, текст, звук и т. д.), нейросетевым моделям с большим количеством настраиваемых параметров недостаточно этого объема данных для полноценного обучения.
- Отсутствие больших структурированных и универсальных наборов табличных данных с разнообразными объектами (наподобие ImageNet для изображений, SQuAD и GLUE для текстов и т. д.) для обучения и оценивания моделей, и, соответственно, отсутствие предобученных архитектур на таблицах.
- Разнородность (гетерогенность) табличных данных, одновременное наличие дискретных категориальных признаков и непрерывных числовых. Нейронные сети чувствительны к масштабу и распределению входов, и если признаки имеют различные шкалы значений, то при обновлении весов в процессе обучения могут появиться доминирующие признаки или вовсе не обновляющиеся веса, нарушится процедура оптимизации, что приведет к затуханию или взрыву градиентов. При этом градиентный бустинг над решающими деревьями не имеет такой проблемы, он устойчив к масштабу, распределению и типу признаков, в нем эта проблема решается нахождением подходящих порогов разбиений в вершинах. Таким образом, необходима корректная нормализация числовых признаков перед их подачей на вход нейросети, и выбор конкретного способа как правило определяется эмпирически или экспериментально[4].

  Категориальные признаки также необходимо преобразовывать, т. к., вообще говоря, нейронные сети не способны обрабатывать дискретные значения, ведь по

сути нейронная сеть является непрерывной дифференцируемой функцией от входов, используемые в ней функции активации как правило непрерывны, веса вещественны, и в обучении используются непрерывные методы оптимизации, и т. д. Также категориальные признаки в табличных данных могут быть представлены в виде символов, текста, даты и т. п., и для дальнейшей их передачи в модель необходимо перевести их в числовой формат. При этом получаемые при перекодировании числовые значения не обязательно будут соответствовать близости или порядку между отдельными категориями. Потенциально перевод категориальных признаков в числовые в любом случае приведет к потере их первоначального смысла и снижению качества прогнозов. Можно использовать one-hot кодирование как один из самых простых и распространенных способов преобразования, однако это приводит к значительному увеличению размеров таблицы и возникновению сильной разреженности в случае большого числа возможных значений признаков, и такие данные будет сложно моделировать в целом. Это также приводит к значительному возрастанию размеров слоев модели в силу увеличения векторов признакового описания объектов, что усложняет дальнейшее обучение. Стоит отметить, что one-hot кодирование не дает использовать понятия близости и схожести категорий, т. к. представления всех значений признака будут равноудалены друг от друга. Другие способы кодирования категориальных признаков, вроде словарного (label) или порядкового (ordinal) кодирования, могут привести к появлению искусственного и не обязательно верного порядка между значениями категорий, появлению ложной интерпретации, а кодирование по целевой переменной (target encoding) и вовсе может привести к утечке информации и переобучению. Преобразования, основанные на проецировании категориального признака на вещественную ось, в целом приводят к сильному упрощению данных и потере смысла категориальности.

- Отсутствие в общем случае порядка в расположении столбцов признаков в таблицах, наличие среди них заведомо неинформативных (id, zipcode и т. п.), неупорядоченность значений категориальных переменных. Также во многих случаях большинство признаков оказывается достаточно слабо скоррелирован-

ными, практически независимыми. Объекты аналогично можно принять независимыми во многих табличных наборах данных. Нередко структура таблиц, описание, смысл и интерпретация признаков заранее неизвестны (анонимизированные или приватные данные, без информативных обозначений столбцов и т. д.). Это затрудняет выявление и использование потенциально полезной информации о признаках и возможных взаимосвязях между ними, не позволяет использовать схожесть между их значениями, преобразовывать конкретный признак с учетом информации об остальных. Также сложно определять и использовать взаимосвязи между непрерывными числовыми и дискретными категориальными переменными.

- Отсутствие понятий близости и локальности в общем случае для таблиц в силу их неоднородной структуры, что затрудняет выявление закономерностей известными методами, опирающимися на использование пространственной и семантической близости (применение сверток, введение понятия контекста и т. д.).
- Высокая гибкость и выразительность моделей глубокого обучения, слишком большая для табличных данных. Нейросетевые модели обладают очень большим числом настраиваемых параметров, и неоднородность и разнотипность признаков в таблицах может приводить к переподгонке сети к конкретным значениям отдельных переменных. Также в исследованиях[3, 4, 2] отмечается, что целевые зависимости на таблицах часто негладкие, не всегда дифференцируемые, более сложно устроенные, чем в других доменах, изменения значений различных признаков в значительно разной степени могут влиять на изменение целевой переменной. Все это во многом затрудняет процесс обучения нейросетей.
- Стоит отметить, что в табличных данных нередко наблюдается наличие выбросов и пропусков, присутствие шумовых или наоборот сильно скоррелированных с целевой переменной признаков, частый и значительный дисбаланс классов. Однако, согласно исследованиям[6, 7, 2, 4], все это слабо сказывается на качестве нейросетевых моделей на таблицах, и главной проблемой является именно природа самих табличных данных, их неоднородность и вытекающие

перечисленные выше особенности. Также в указанных работах отмечается, что оптимизация гиперпараметров для нейросетей как правило не дает значимого прироста качества в задачах с таблицами. На практике также было выяснено, что использование специализированных архитектур не так сильно влияет на качество в целом, как способ предобработки табличных данных[6].

Таким образом, можно сказать, что основной проблемой для нейронных сетей при работе с таблицами является задача их правильной предобработки и преобразования. Актуальной является задача нахождения способов преобразования признаков, которые в наибольшей степени решают указанные выше проблемы.

Что касается используемых на табличных данных нейросетевых архитектур, то на практике как правило применяются известные модели глубокого обучения с некоторыми модификациями, в основном полносвязные сети, вариации ResNet и DenseNet, а также модели трансформеры как наиболее подходящие и эффективные в данных задачах. Также данные архитектуры используются как базовые во многих современных табличных моделях[8, 9, 7, 10, 11, 12]. Полносвязные сети демонстрируют неплохое качество в большом числе задач, достаточно просто реализуются, модифицируются и встраиваются в более сложные архитектуры. В моделях наподобие ResNet[13] использование прокидывания связей между слоями позволяет комбинировать и учитывать большое число признаков разного уровня при последовательном построении внутренних представлений в сети и при формировании прогноза, что часто оказывается полезным в задачах на таблицах (комбинаторные признаки вроде «возрастная группа == A, специальность == B» в задачах кредитного скоринга или рекомендательных системах и т. п.). Модели трансформеры за счет применения механизмов внимания[5] позволяют извлекать и использовать попарные взаимодействия переменных, и это может существенно повысить качество предсказания (например, учет взаимосвязи признаков наподобие «пол», «возраст», «категория товара» и т. п.). Однако для использования трансформеров необходимо преобразовать исходные табличные признаки в векторные представления (т. н. «эмбеддинги», embeddings) одинакового размера. Сверточные сети практически не используются в задачах с таблицами в силу сложности применения операции свертки из-за разнородности данных и отсутствия понятия пространственной близости.

Рекуррентные архитектуры также почти не применяются к табличным данным, так как объекты и признаки в большинстве случаев не упорядочены в таблицах, и не всегда корректно будет рассматривать набор таких данных как последовательность.

# 4 Методы решения

Для решения перечисленных проблем обычно строят векторные представления исходных табличных признаков. Существует ряд способов получения таких представлений для категориальных переменных[14, 15], и их использование позволяет оперировать близостью в соответствующем векторном пространстве, получать более высокоуровневые признаки и использовать взаимосвязи между ними. Однако если использовать преобразованные в векторы категориальные переменные вместе с лишь нормализованными числовыми, то во входе нейросети (объединенных в один вектор исходных признаках и векторных представлениях категорий в случае, например, многослойного перцептрона или ResNet) категориальные переменные будут «представлены» сильнее, чем числовые, в силу больших размеров представлений, что также может привести к переобучению и нахождению неверных закономерностей в данных. Также для использования моделей трансформеров необходимо преобразовать все признаки в вектора одного размера. Поэтому для числовых признаков также разумно строить векторные представления, того же размера, что и для категориальных. Это в определенной степени соответствует проецированию переменных в векторные пространства одинаковой размерности, что может позволить использовать схожесть и взаимодействия между признаками одного и потенциально разных типов. Стоит отметить, что порядок векторных представлений во входе нейросетевых моделей (в наборе векторов для трансформера и в объединенном векторе для полносвязной сети и ResNet) не будет иметь значения в силу особенностей устройства данных архитектур.

Векторные представления признаков можно строить множеством способов. С целью повышения обобщающей способности и качества прогнозов модели разумнее всего рассматривать параметризованные, обучаемые трансформации. Наиболее

логичным и интуитивным будет преобразование числовых и категориальных переменных по отдельности, индивидуальными способами, т. к. данные типы признаков принципиально различны. Однако можно строить векторные представления для разнотипных признаков по одному принципу в предположении, что такое отображение в одно векторное пространство позволит лучше учесть возможные связи между признаками разного вида. При этом такое преобразование, очевидно, исказит изначальную суть числовых и категориальных переменных, и это может привести к ухудшению качества прогноза. Можно либо перевести все признаки в непрерывные, либо в дискретные, а затем применять тот или иной способ построения векторных представлений. Также, учитывая разнородность табличных данных, наиболее приемлемым видится индивидуальное независимое построение представлений для каждого признака, без использования информации об остальных. При преобразовании признака с учетом других исходная информация о нем может исказиться в силу разного типа, масштаба и наличия взаимосвязей между переменными, и они могут учесться неверно в таком случае. Для выявления и использования взаимозависимостей между признаками часто используют механизмы внимания уже применительно к построенным для них векторным представлениям[8, 9, 7, 10].

Среди методов построения самих векторных представлений признаков можно выделить и рассмотреть следующие современные и популярные способы.

## 4.1 Сопоставление отдельных представлений из набора каждому признаку

Данный способ является одним из простейших и наиболее распространенных. Пусть $x_i = (x_{i1}, \ldots, x_{id})$, $x_i \in \mathbb{R}^d$ – признаковое описание $i$-ого объекта. Среди признаков есть $n_{cat}$ категориальных и $n_{num}$ числовых признаков, $n_{cat} + n_{num} = d$. Пусть $x_{ij_1}, \ldots, x_{ij_{n_{cat}}}$ – категориальные признаки, и каждый такой признак $x_{ij_k}$ может принимать $d_k$ различных значений, $k = \overline{1, n_{cat}}$, т. е. $x_{ij_k} \in \{1, \ldots, d_k\}$. Для каждой категориальной переменной $x_{ij_k}$, $k = \overline{1, n_{cat}}$ создается матрица обучаемых параметров $E_{j_k} \in \mathbb{R}^{d_k \times d_{emb}}$, где $d_{emb}$ – размер векторных представлений числовых и категориальных переменных. Если для $i$-ого объекта $x_{ij_k} = t$, $t \in \{1, \ldots, d_k\}$, то

данному признаку сопоставляется $t$-ая строка матрицы $E_{j_k}$ в качестве векторного представления. По сути это эквивалентно умножению one-hot вектора признака на соответствующую матрицу представлений. Для числовых переменных $x_{il_1}, \ldots, x_{il_{n_{num}}}$ создается обучаемая матрица $E_{num} \in \mathbb{R}^{n_{num} \times d_{emb}}$, и преобразование признака $x_{il_k}$, $k = \overline{1, n_{num}}$ происходит путем умножения его скалярного значения $x_{il_k}$ поэлементно на $l_k$-ую строку матрицы $E_{num}$. Также для каждой числовой и категориальной переменной аналогичным образом можно сопоставлять обучаемый вектор смещения, прибавляющийся к итоговому представлению. Дальнейшая подача полученных представлений на вход модели зависит от конкретной архитектуры. Для модели трансформера дополнительных действий не требуется, можно подавать данные векторы без изменений, для многослойного перцептрона или архитектуры ResNet требуется сконкатенировать их в один вектор. Таким же образом формируются входы для данных моделей и при любом другом схожем способе преобразований признаков. В целом рассмотренный способ предобработки довольно прост и логичен, применяется во многих известных нейросетевых моделях для табличных данных, например, в FT-Transformer[7] и AutoInt[10]. На рис. 1 приведен наглядный пример такого преобразования для архитектуры FT-Transformer[7].

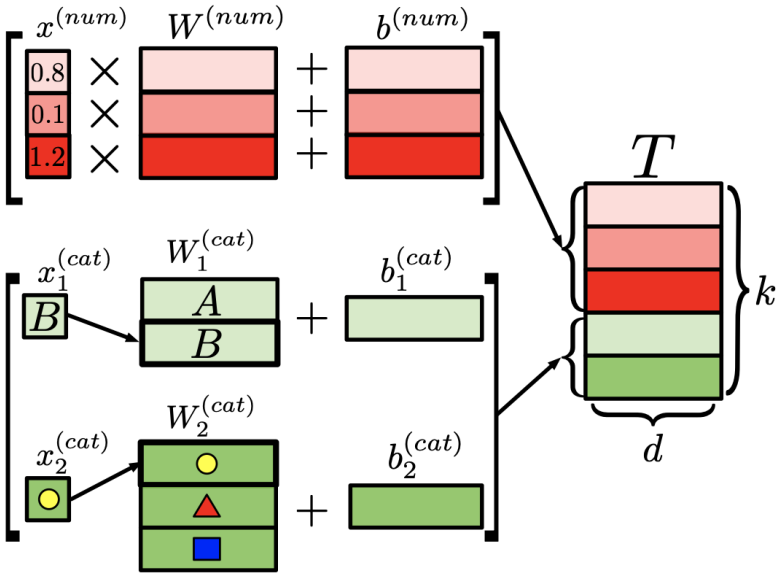


Рис. 1: Иллюстрация из статьи[7], демонстрирующая процесс построения векторных представлений для вещественных и категориальных переменных в модели FT-Transformer.

## 4.2 Использование линейных слоев с активациями

Этот способ предобработки часто и успешно используется во многих известных нейросетевых архитектурах для работы с таблицами[10, 16, 6] и в целом в глубоком обучении. Векторное представление строится для каждого признака в результате его прохода через последовательность линейных слоев и нелинейных функций активации. В общем случае для исходного вещественного признака $x_{ij} \in \mathbb{R}$ результат прохождения через первый слой и функцию активации будет формироваться как $v_1 = activation\left(x_{ij} \cdot W_1\right) \in \mathbb{R}^{d_1}$, $W_1 \in \mathbb{R}^{d_1}$, и финальное векторное представление после прохождения $k$ линейных слоев и нелинейностей будет иметь вид $v_k = activation\left(W_k v_{k-1}\right) \in \mathbb{R}^{d_k}$, $W_k \in \mathbb{R}^{d_k \times d_{k-1}}$, где $activation\left(\cdot\right)$ – некоторая нелинейная функция активации. Использование нескольких слоев и активаций позволяет получать более богатое признаковое описание и моделировать более сложные зависимости, что может положительно сказаться на качестве предсказания. Данный вид предобработки прост в реализации и может быть распространен и на категориальные переменные путем замены $v_1 = activation\left(W_1 e_1\right) \in \mathbb{R}^{d_1}$, $W_1 \in \mathbb{R}^{d_1 \times d_0}$, где $e_1 \in \mathbb{R}^{d_0}$— one-hot вектор признака, $d_0$ – число возможных значений признака $x_{ij}$. Также можно варьировать размеры и число слоев, вид используемых функций активаций и т. д. Часто на практике реализация данного преобразования в виде двух линейных слоев с функциями активации $LeakyReLU$ демонстрирует наилучшие результаты[6, 16], данный вариант будет рассматриваться в дальнейшем в данной работе.

## 4.3 Проекция признака на набор векторных представлений

В данном подходе в определенном смысле проводится «мягкая дискретизация» признаков, и их представления строятся как агрегация нескольких обучаемых векторов. Для каждой числовой переменной $x_{ij} \in \mathbb{R}$ поддерживается матрица обучаемых параметров $M_j \in \mathbb{R}^{H_j \times d_{emb}}$, где $H_j$ – число обучаемых представлений, $d_{emb}$ – их размерность. Строится вектор «вероятностей» распределения признака по $H_j$ векторам с помощью одного линейного слоя и функции $softmax$: $\widetilde{x_j} = x_{ij} \cdot W_j$, $W_j \in \mathbb{R}^{H_j}$, $\widehat{x_{jk}} = \frac{\exp\left(\frac{\widetilde{x_{jk}}}{\tau}\right)}{\sum\limits_{l=1}^{H_j} \exp\left(\frac{\widetilde{x_{jl}}}{\tau}\right)}$, $k = \overline{1, H_j}$, $\tau$ – параметр температуры, параметры

$W_j$ обучаемы. Вектор $\widehat{x_j} = \left(\widehat{x_{j1}}, \ldots, \widehat{x_{jH_j}}\right) \in \mathbb{R}^{H_j}$ – искомый вектор «вероятностей». Итоговое векторное представление признака $x_{ij}$ формируется из векторов матрицы $M_j$ как их взвешенная сумма с весами $\widehat{x_{jk}}$: $e_j = \widehat{x_j}^T M_j$. Адаптацию способа для категориальных переменных в целом можно свести к умножению ohe-hot векторов на таблицу представлений. В целом такое преобразование признака позволяет конструировать векторные представления более гибко и разнообразно, используя сразу несколько обучаемых векторов, и такие представления будут получаться потенциально более мощными и богатыми. Данный способ используется в различных табличных моделях, например, в библиотеке transformers4rec (SoftEmbeddings[17]).

## 4.4 AutoDis

Метод[16] является по своей сути объединением и развитием подходов с использованием линейных слоев и проекцией на набор векторов. Для каждого числового признака $x_{ij} \in \mathbb{R}$ аналогично используется обучаемая матрица $M_j \in \mathbb{R}^{H_j \times d_{emb}}$, но вектор весов ее строк строится уже с помощью двух линейных слоев с активациями, прокидывания связей и функции $softmax$: $h_j = LeakyReLU\left(x_{ij} \cdot w_j\right)$, $\widetilde{x_j} = W_j h_j + \alpha h_j$, где $w_j \in \mathbb{R}^{H_j}$, $W_j \in \mathbb{R}^{H_j \times H_j}$ – обучаемые параметры, $\alpha$ – гиперпараметр в прокидывании связей. Вектор $\widetilde{x_j} = \left(\widetilde{x_{j1}}, \cdots, \widetilde{x_{jH_j}}\right)^T$ интерпретируется как вектор проекций исходного признака на таблицу представлений $M_j$, далее к нему применяется функция $softmax$: $\widehat{x_{jk}} = \frac{\exp\left(\frac{\widetilde{x_{jk}}}{\tau}\right)}{\sum\limits_{l=1}^{H_j} \exp\left(\frac{\widetilde{x_{jl}}}{\tau}\right)}$, $\tau$ – гиперпараметр температуры. Конечное векторное представление аналогично получается из векторов $M_j$, взятых с весами $\widehat{x_{jk}}$, $k = \overline{1, H_j}$: $e_j = \widehat{x_j}^T M_j$. Данный подход объединяет в себе достоинства способов с линейными слоями и активациями и проекцией на множество векторов, за счет нескольких слоев, нелинейных активаций, прокидывания связей и агрегации нескольких обучаемых векторов итоговые представления признаков будут получаться более гибкими и выразительными. На рис. 2 изображен процесс получения векторного представления признака в этом методе.

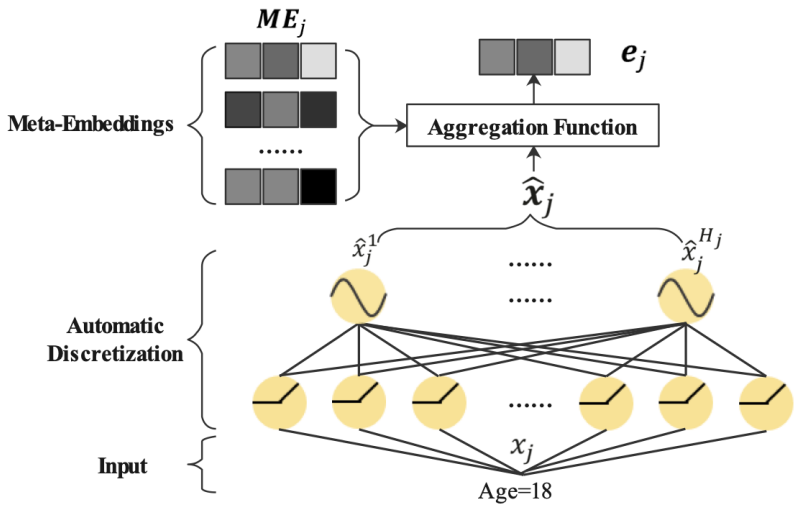


Рис. 2: Иллюстрация из статьи[16], демонстрирующая процесс построения векторных представлений для признаков в методе AutoDis.

## 4.5 Использование периодических функций

Данный вид построения векторных представлений первоначально использовался для вещественных признаков[18, 19, 6] и был успешно применен в ряде задач. Утверждается, что он может быть эффективным в задачах приближения сложных нелинейных и высокочастотных зависимостей, что может быть полезно в случае табличных данных в силу их неоднородности и соответствующих особенностей. Метод устроен следующим образом. Пусть $x_{ij} \in \mathbb{R}$ – вещественный признак (в более общем случае может быть и вектором). Для него конструируется вектор $v = (2\pi c_1 x_{ij}, \cdots, 2\pi c_{d_{emb}} x_{ij})^T$, $v \in \mathbb{R}^{d_{emb}}$, а затем строится преобразование вида $\hat{v} = concat\left(\sin(v), \cos(v)\right)^T$, т. е. функции синуса и косинуса применяются к вектору $v$ поэлементно, а результаты конкатенируются. Параметры $c_i$ являются обучаемыми, в оригинальной реализации[19] инициализируются как $c_i \sim \mathcal{N}(0, \sigma^2)$. Данный способ предобработки можно адаптировать и для категориальных переменных: пусть $e_k = (0, \cdots, 0, 1, 0, \cdots, 0)^T, e_k \in \mathbb{R}^n$ – one-hot вектор, соответствующий $k$-ому значению некоторого категориального признака с $n$ различными возможными значениями, единица в $e_k$ стоит на $k$-ой позиции. Тогда можно каждой $k$-ой категории сопоставить вектор обучаемых коэффициентов $v_k = (2\pi c_{k1}, \cdots, 2\pi c_{kd_{emb}})^T$ и объединить их все в матрицу $V \in \mathbb{R}^{n \times d_{emb}}$. Итоговое векторное представление соответствующего катего-

риального признака можно получить матричным умножением $V$ на его one-hot вектор и применением функций синуса и косинуса: $v = e_k^T V$, $\hat{v} = concat\,(\sin(v), \cos(v))$. Стоит отметить, что для данного способа предобработки $\sigma$ и $d_{emb}$ являются гиперпараметрами, играющими существенную роль[6].

## 4.6 Использование периодических функций с необучаемыми коэффициентами

Метод по сути аналогичен предыдущему, однако коэффициенты $c_i$ в нем необучаемы и сэмплируются из $\mathcal{N}(0, \sigma^2)$, где $\sigma$ – гиперпараметр. Данный способ, очевидно, проще предыдущего варианта, однако требует меньшего количества вычислений и потенциально может построить полезные нелинейные признаки[20]. В некоторых работах[21] такой метод преобразования продемонстрировал свою эффективность на малых табличных наборах данных.

## 4.7 ARM

Данный способ был предложен в архитектуре ARM-Net[22], и его основной особенностью является моделирование попарных нелинейных взаимодействий между признаками (т. н. кросс-признаки). Сначала для всех исходных табличных переменных строятся представления путем сопоставления обучаемого вектора из таблицы, т. е. описанным в разделе 4.1 способом: каждому из исходных признаков $x_{i1}, \ldots, x_{id}$ сопоставляются вектора $e_1, \ldots, e_d$ соответственно, $e_j \in \mathbb{R}^{d_{emb}}$, $j = \overline{1, d}$, $d_{emb}$ – размерность представлений. Далее используется следующая идея. Нейроны в моделях глубокого обучения агрегируют входные переменные с некоторыми весами, т. е. в них формируется выход вида $y = \sum_{j=1}^{d} w_j x_{ij}$, где $w = (w_1, \ldots, w_d) \in \mathbb{R}^d$ – вектор весов нейрона, $x_i = (x_{i1}, \ldots, x_{id}) \in \mathbb{R}^d$ – вход нейрона, и этот выход $y$ далее проходит через нелинейную функцию активации. По сути выход нейрона формируется из входных признаков линейным образом. Хоть нейронные сети и являются универсальными аппроксиматорами непрерывных функций[23], но все же за счет своего устройства они не так хорошо моделируют неаддитивные и нелинейные зависимо-

сти, содержащие взаимодействия между переменными в виде произведений, отношений, возведений в степень и т. д.[24]. Авторы метода предлагают использовать переход в экспоненциальное пространство для лучшего учета такого рода связей, используя идею, что при потенцировании аддитивные операции переходят в мультипликативные. Используются т. н. экспоненциальные нейроны, в которых выход формируется по принципу $y_i = \exp\left(\sum_{j=1}^{d} w_{ij} e_j\right) = \exp(e_1)^{w_{i1}} \circ \cdots \circ \exp(e_d)^{w_{id}}$, где $e_1, \ldots, e_d$ – векторные представления признаков с предыдущего шага, $\circ$ – операция покомпонентного произведения (произведение Адамара), и операции взятия экспоненты и возведения в степени $w_{i1}, \ldots, w_{id}$ также производятся покомпонентно. Веса $w_{i1}, \ldots, w_{id}$ для каждого экспоненциального нейрона определяются с помощью механизмов внимания с множеством голов. Это способствует тому, чтобы каждый экспоненциальный нейрон условно уделял внимание более информативным признакам и их комбинациям. Имеется $K$ голов внимания с $m$ экспоненциальными нейронами в каждой. Каждому такому $i$-ому нейрону сопоставляется обучаемый вектор весов $v_i \in \mathbb{R}^d$ (value), а также вектор $q_i \in \mathbb{R}^{d_{emb}}$ (query). Веса внимания строятся следующим образом: $\widetilde{z_{ij}} = \phi_{att}(q_i, e_j) = q_i^T W_{att} e_j$, $j = \overline{1,d}$, $\widetilde{z_i} = (\widetilde{z_{i1}}, \ldots, \widetilde{z_{id}}) \in \mathbb{R}^d$, где $W_{att} \in \mathbb{R}^{d_{emb} \times d_{emb}}$ – обучаемая матрица коэффициентов. Далее используется функция $\alpha$-$entmax$[25, 26](в некоторой степени разреженная версия функции $softmax$): $z_i = \alpha\text{-}entmax(\widetilde{z_i}) = ((\alpha - 1)\widetilde{z_i} - \tau \mathbf{1})^{\frac{1}{\alpha - 1}}$, где $\mathbf{1} \in \mathbb{R}^d$. $\alpha$ является гиперпараметром, и большие его значения соответствуют более разреженным выходным векторам распределений и, соответственно, более жесткому отбору признаков. Сами веса для экспоненциальных нейронов определяются как $w_i = z_i \circ v_i$, $w_i \in \mathbb{R}^d$. Вектор $z_i$ по сути отвечает за фильтрацию шумных и неинформативных признаков, что способствует более качественному моделированию взаимодействий между признаками[25, 26]. Таким образом, для каждой из $K$ голов имеется $m$ экспоненциальных нейронов, и для каждой $k$-ой головы внимания, $k = \overline{1,K}$, имеется $m$ векторов вида $v_j^k$, $q_j^k$: $V^k = \left(v_1^k, \ldots, v_m^k\right)$, $V^k \in \mathbb{R}^{d \times m}$, $Q^k = \left(q_1^k, \ldots, q_m^k\right)$, $Q^k \in \mathbb{R}^{d_{emb} \times m}$. В итоге строится $K \cdot m$ выходных векторов $y_i = \exp\left(\sum_{j=1}^{d} w_{ij} e_j\right)$, $y_i \in \mathbb{R}^{d_{emb}}$, $i = \overline{1, K \cdot m}$ – итоговые представления признаков. Стоит отметить, что их число $K \cdot m$ вообще говоря отличается от исходного числа переменных $d$, однако предполагается, что такие представления позволят эффективно моделировать и учитывать нелинейные взаимодействия между признаками вроде

произведений, возведений в степень и т. д., нередко встречающихся в табличных наборах данных[22]. На рис. 3 изображена описанная схема формирования векторных представлений признаков.

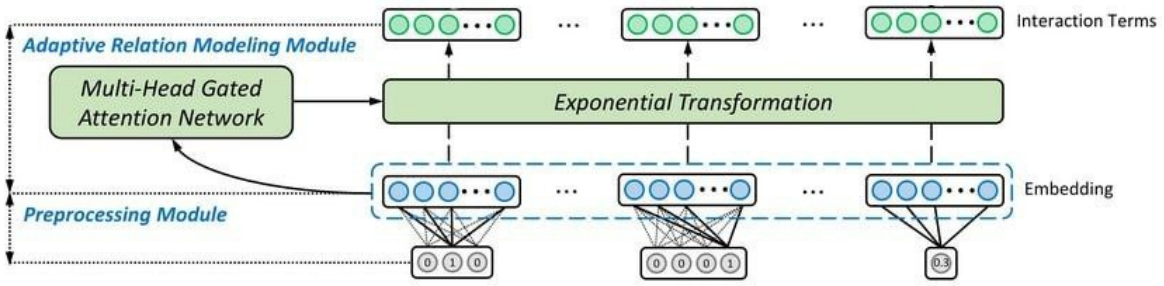


Рис. 3: Иллюстрация из статьи[22], демонстрирующая схему формирования векторных представлений признаков в методе ARM.

## 4.8 Использование механизмов внимания и моделей трансформеров

Подход с использованием механизмов внимания и архитектуры трансформер[5] для построения векторных представлений признаков используется в ряде современных табличных нейросетевых моделей, например, TabTransformer[9], AutoInt[10], FT-Transformer[7] и т. д., чаще всего применительно к категориальным переменным. Основная идея применения данных решений заключается в формировании «контекстуальных» векторных представлений признаков, т. е. учитывающих возможные взаимосвязи между ними. Каждому из признаков $x_{i1}, \ldots, x_{id}$ сопоставляются вектора $e_1, \ldots, e_d$ соответственно способом, описанным в разделе 4.1, $e_j \in \mathbb{R}^{d_{emb}}$, $j = \overline{1,d}$. В матричном виде это может быть представлено как $E = (e_1, \ldots, e_d)^T \in \mathbb{R}^{d \times d_{emb}}$. Далее они подаются в модель трансформер, в общем случае состоящую из последовательности блоков следующего вида. В начале используется механизм внимания с $n_h$ головами. Для каждой $h$-ой головы поддерживаются отдельные обучаемые матрицы параметров $W_h^K \in \mathbb{R}^{d_{emb} \times k}$ (key), $W_h^Q \in \mathbb{R}^{d_{emb} \times k}$ (query), $W_h^V \in \mathbb{R}^{d_{emb} \times v}$ (value). Строятся проекции входов на данные матрицы: $K = EW_h^K$, $Q = EW_h^Q$, $V = EW_h^V$. Далее выход одной головы формируется как $H_h = softmax\left(\frac{QK^T}{\sqrt{k}}\right)V \in \mathbb{R}^{d \times v}$. Матрица $A_h = softmax\left(\frac{QK^T}{\sqrt{k}}\right) \in \mathbb{R}^{d \times d}$ по сути

отражает, насколько исходные признаки взаимодействуют друг с другом. В итоге выходы каждой из голов внимания агрегируются и умножаются на обучаемую матрицу: $\widehat{E} = concat\,(H_1, \ldots, H_{n_h})\,W^O \in \mathbb{R}^{d \times d_{emb}}$, $W^O \in \mathbb{R}^{n_h v \times d_{emb}}$. Затем матрицы $E$ и $\widehat{E}$ складываются, и над ними проводится нормализация по слою[27]: $\widetilde{E} = \widehat{E} + E$, $\widetilde{E}_{LN} = \frac{\widetilde{E}-\mu}{\sqrt{\sigma^2+\varepsilon}}$, $\mu = \frac{1}{d \cdot d_{emb}} \sum_{i=1}^{d} \sum_{j=1}^{d_{emb}} \widetilde{E}_{ij}$, $\sigma^2 = \frac{1}{d \cdot d_{emb}} \sum_{i=1}^{d} \sum_{j=1}^{d_{emb}} \left(\widetilde{E}_{ij} - \mu\right)$. После матрица $\widetilde{E}_{LN}$ подается в полносвязный блок с выходным слоем той же размерности $d_{emb}$, его выход строится как $\widetilde{E}_{FFN} = FFN\left(\widetilde{E}_{LN}\right) \in R^{d \times d_{emb}}$. Во многих реализациях блок $FFN$ задается как 2-3 линейных слоя с активациями. В конечном счете вход $\widetilde{E}_{LN}$ и выход $\widetilde{E}_{FFN}$ аналогично суммируются и снова проходят нормализацию по слою, и в итоге строятся векторные представления $E_1$. Далее выход одного такого блока трансформера может проходить через последовательность аналогичных блоков, претерпевая все те же описанные преобразования. На рис. 4 изображена схема одного описанного блока в последовательности.

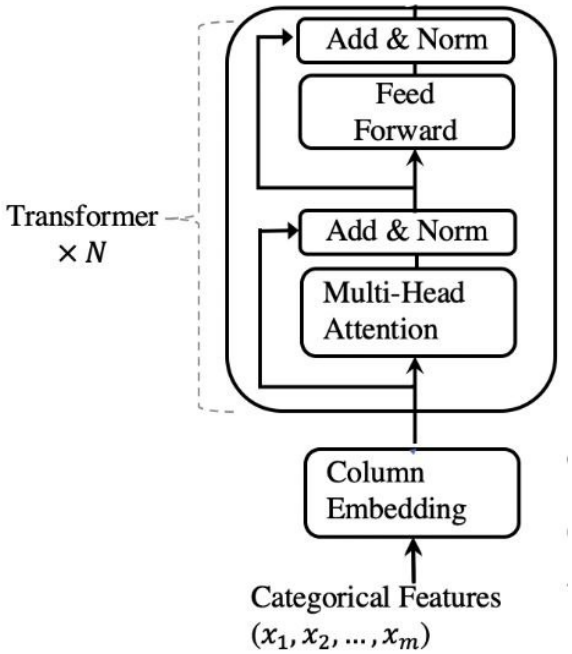


Рис. 4: Иллюстрация из статьи[9], демонстрирующая схему блока модели трансформер, используемой для построения векторных представлений признаков.

Получаемые таким образом векторные представления будут достаточно информативными и мощными за счет использования множества параметризованных преобразований и прокидывания связей, а также будут учитывать взаимодействия с

остальными признаками благодаря применению механизмов внимания. Утверждается также, что при таком способе преобразований представления скоррелированных или схожих признаков будут близки как векторы в соответствующем пространстве[9].

# 5 Эксперименты

Для анализа и сравнения рассмотренных методов предобработки табличных данных для нейросетевых моделей были проведены вычислительные эксперименты с использованием наиболее популярных архитектур глубокого обучения и методов преобразования признаков применительно к различным задачам и данным. Основным предметом исследования является качество предсказаний на тестовых данных, полученное при использовании конкретного способа предобработки и модели. Реализации методов преобразования признаков, нейросетевых архитектур и всех необходимых функций для проведения вычислительных экспериментов были выполнены с использованием языка программирования Python и библиотеки Pytorch[28].

## 5.1 Данные

Для проведения экспериментов были рассмотрены задачи бинарной классификации и задачи регрессии. Наборы данных были взяты из открытых источников, таких как платформы Kaggle[29] и OpenML[30]. Характерными особенностями рассмотренных данных можно назвать одновременное присутствие непрерывных числовых и дискретных категориальных переменных, разный масштаб признаков и целевых переменных в целом, произвольный порядок столбцов и строк. Также можно сказать, что данные достаточно разнообразны, содержат различное число признаков и объектов, содержат информацию из различных прикладных областей и отражают различные виды зависимостей. Список наборов данных и их характеристики представлены в табл. 1.

| **Набор данных** | **Задача** | **Обозначение** | **# строк** | **# числовых признаков** | **# категориальных признаков** |
|---|---|---|---|---|---|
| Give Me Some Credit | Бинарная классификация | GSC | 150000 | 4 | 6 |
| Churn Modelling | Бинарная классификация | CM | 10000 | 4 | 6 |
| Vehicle Loan Default | Бинарная классификация | VLD | 233154 | 33 | 8 |
| Adult Income | Бинарная классификация | AI | 48842 | 5 | 9 |
| HELOC | Бинарная классификация | HL | 10459 | 30 | 5 |
| Fraud Ecommerce | Бинарная классификация | FE | 151112 | 24 | 3 |
| Black Friday | Регрессия | BF | 166821 | 4 | 5 |
| Brazilian Houses | Регрессия | BH | 10685 | 9 | 3 |
| Beijing PM2.5 | Регрессия | BP | 43792 | 9 | 2 |
| Colleges | Регрессия | CL | 7063 | 39 | 5 |
| House Sales | Регрессия | HS | 21568 | 16 | 5 |
| NYC Taxi | Регрессия | NT | 581006 | 11 | 7 |

Таблица 1: Данные, использованные в ходе вычислительных экспериментов. Источник — платформы Kaggle[29] и OpenML[30].

## 5.2 Рассмотренные методы предобработки данных

Были рассмотрены следующие способы преобразования признаков табличных данных:

- **Сопоставление векторного представления из обучаемой таблицы**. Для каждого типа признаков представления строятся способом, описанным в разделе 4.1 с использованием обучаемого вектора смещения.
- **Использование линейных слоев с активациями**. Рассматривается реализация в виде двух линейных слоев с функциями активации $LeakyReLU$ как одна из наиболее простых и эффективных на практике конфигураций[6, 16].

- **Проекция признака на набор векторных представлений**. Размер таблицы векторных представлений в методе принимается равным 20 согласно рекомендациям в исходной статье[17, 16], параметр температуры по умолчанию берется равным 1.
- **AutoDis[16]**. Число векторов в таблице для проекций аналогично берется равным 20, используется равный 1 температурный коэффициент.
- **Использование периодических функций с обучаемыми коэффициентами**. Векторные представления признаков строятся в соответствии с описанным в разделе 4.5 способом. Используется значение гиперпараметра $\sigma = 0.1$ в соответствии с исходной статьей.
- **Использование периодических функций с необучаемыми случайными коэффициентами**. Способ аналогичен предыдущему, однако коэффициенты $c_i$ являются необучаемыми и сэмплируются из $\mathcal{N}(0, \sigma^2)$, где $\sigma = 0.1$ по аналогии с предыдущим методом.
- **ARM[22]**. Метод реализован согласно разделу 4.7, используется 8 голов внимания с 16 экспоненциальными нейронами каждая и коэффициентом $\alpha = 1.5$ на основании оригинальной статьи[22].
- **Использование механизмов внимания и архитектуры трансформер**. Для построения векторных представлений применяется архитектура трансформер с 5 основными блоками, 8 головами внимания в каждом, полносвязная часть блока реализуется в виде двух линейных слоев с активациями.

Для удобства дальнейшего описания результатов введем систему обозначений: **P** – использование периодических функций с обучаемыми коэффициентами, **F** – использование периодических функций с необучаемыми коэффициентами, **L** – применение линейных слоев с активациями, **AD** – AutoDis, **E** – использование обучаемой таблицы представлений, **SE** – проекции признака на набор представлений, **ARM** – ARM, **TT** – использование механизмов внимания и архитектуры трансформер. Стоит отметить, что среди перечисленных способов преобразования признаков напрямую без дополнительных действий к непрерывным переменным могут применяться все методы, к категориальным же переменным без вспомогательных преобразований, по сути, корректно применять только методы **E**, **ARM**, **TT**, т. к. в них есть стадия пер-

воначального сопоставления дискретным категориям вещественнозначных векторов. Для остальных методов предобработки, вообще говоря, необходимы вещественные значения на входе. Для них в ходе экспериментов применяется нормализация исходных числовых признаков для улучшения сходимости процесса обучения нейронной сети.

В ходе экспериментов рассматривались 3 схемы построения представлений для всех табличных признаков:

- **Раздельное преобразование числовых и категориальных признаков**. Данный способ является наиболее логичным и интуитивным в силу принципиальных различий этих видов переменных, и он в некоторой степени сокращает потери исходной информации о данных при преобразованиях. Для каждого типа переменных применяется один конкретный способ получения векторных представлений одинакового размера. Для числовых признаков используются все рассмотренные выше способы, для категориальных – методы **E**, **ARM**, **TT**.
- **Рассмотрение всех исходных признаков как непрерывных и последующие преобразования**. При таком способе, очевидно, искажается изначальная суть категориальных признаков и требуется их предварительное преобразование. Однако данный вид построения векторных представлений был рассмотрен в предположении, что это в некоторой степени будет способствовать проекции переменных в одно векторное пространство, и что это может позволить учесть взаимосвязи между ними и привести к повышению качества прогнозов. К категориальным переменным применялось one-hot кодирование как в наибольшей мере сохраняющее суть категориальности преобразование (каждая отдельная категория становится бинарным признаком), далее все признаки трактовались как непрерывные числовые, и использовался один из перечисленных способов получения векторных представлений.
- **Рассмотрение всех исходных признаков как категориальных и последующие преобразования**. Мотивация использования данной схемы и вытекающие недостатки аналогичны предыдущей идее. Числовые признаки дискретизуются по квантилям распределения их значений на заданное число групп, причем объекты таким образом распределятся по этим группам значений при-

знаков достаточно равномерно. Попадание объекта в ту или иную группу по величине признака описывается отдельным бинарным признаком. В итоге все переменные преобразуются в дискретные категориальные, и применяется один из способов формирования векторных представлений, **E**, **ARM** или **TT**.

Каждая из схем рассматривалась с каждым из возможных наборов методов построения векторных представлений признаков применительно к каждой из рассмотренных нейросетевых архитектур. Способы формирования представлений переменных по своей сути выполнены в виде отдельных слоев с настраиваемыми параметрами в составе общей архитектуры модели.

## 5.3 Рассмотренные модели глубокого обучения

В ходе вычислительных экспериментов были рассмотрены следующие нейросетевые архитектуры как наиболее используемые на практике и подходящие к применению на табличных данных:

- **Многослойный перцептрон**. Используется стандартная реализация в виде последовательности линейных слоев с батч-нормализацией[31] и dropout-регуляризацией[32] на входе и функцией активации $LeakyReLU$ на выходе.
- **Модель ResNet**. Рассматривается архитектура, составленная из последовательности т. н. ResNet-блоков, состоящих из слоев батч-нормализации, линейного слоя, активации, dropout-регуляризации, второго линейного слоя и еще одной dropout-регуляризации. Также для всех блоков последовательности используется прокидывание связей между входом предыдущего и входом следующего блока.
- **Модель трансформер**. Используется стандартная версия, состоящая из последовательности т. н. трансформер-блоков. Каждый блок состоит из модуля внимания с несколькими головами, полносвязной части в виде двух линейных слоев с активациями, а также с прокидыванием связей, dropout-регуляризацией и нормализацией по слою между ними. Стоит отметить, что данной архитектуре необходимо преобразование входных признаков в векторные представления

одной длины, тогда как многослойный перцептрон и ResNet в принципе могут использовать исходные признаки, объединенные в один вектор.

Для удобства представления результатов экспериментов аналогично введем обозначения: **M** – многослойный перцептрон, **R** – ResNet, **T** – трансформер. Для многослойного перцептрона и ResNet представления признаков конкатенируются в один вектор перед подачей на вход, для трансформера же построенные представления признаков подаются без каких-либо объединений и прочих преобразований.

## 5.4 Детали обучения и экспериментов

Каждая из схем преобразования признаков рассматривалась с каждым из возможных наборов способов построения векторных представлений для категориальных и числовых признаков, применительно к каждой из выбранных нейросетевых архитектур на каждом из наборов данных. Данные предварительно разделялись на обучающую и тестовую выборки в соотношении 7/3, обучающий набор данных также делится на непосредственно обучающую и валидационную выборку в соотношении 8/2. Валидационная выборка используется для оптимизации гиперпараметров и контролирования процесса обучения нейросетей. Для каждой конфигурации данных, модели и способа преобразования признаков предварительно проводится оптимизация гиперпараметров на валидационной выборке с использованием пакета Optuna[33] с ограничением в 20 попыток. Также некоторые конфигурации параметров проверялись в отдельных запусках, опираясь на статьи, использующие соответствующие модели и преобразования признаков. Оптимизируются как гиперпараметры моделей, так и параметры обучения и размер векторных представлений переменных. Параметры методов формирования представлений при их наличии берутся фиксированными в соответствии с реализациями и результатами в исходных статьях. Рассмотренные значения гиперпараметров для различных архитектур и оптимизатора приведены в табл. 2, табл. 3, табл. 4, табл. 5, табл. 6. Обучение моделей производилось с использованием GPU Nvidia Tesla P100.

| **Гиперпараметр** | **Значения** |
|---|---|
| # слоев | $\{2, 3, 4, 5\}$ |
| Размерность слоев | $\{64, 128, 256, 512\}$ |
| Параметр dropout-регуляризации | $\{0, 0.1, 0.25, 0.5\}$ |

Таблица 2: Оптимизируемые гиперпараметры для многослойного перцептрона и рассмотренные их значения.

| **Гиперпараметр** | **Значения** |
|---|---|
| # ResNet-блоков | $\{2, 3, 4, 5\}$ |
| Множитель в скрытом слое (hidden factor) | $\{2, 3, 4, 5\}$ |
| Параметр dropout-регуляризации (для слоев и для прокидывания связей) | $\{0, 0.1, 0.25, 0.5\}$ |

Таблица 3: Оптимизируемые гиперпараметры для архитектуры ResNet и рассмотренные их значения.

| **Гиперпараметр** | **Значения** |
|---|---|
| # трансформер-блоков | $\{2, 3, 4, 5\}$ |
| Число голов внимания | $\{8, 16\}$ |
| Параметр dropout-регуляризации (для механизма внимания и для прокидывания связей) | $\{0, 0.1, 0.25, 0.5\}$ |

Таблица 4: Оптимизируемые гиперпараметры для архитектуры трансформер и рассмотренные их значения.

| **Гиперпараметр** | **Значения** |
|---|---|
| Число деревьев | $\{50, 100, 200, 500, 1000\}$ |
| Максимальная глубина деревьев | $[3, 10]$ |
| Темп обучения | $[10^{-3}, 1]$ |
| Минимальное уменьшение потерь для разбиения ($\gamma$) | $[10^{-2}, 1]$ |
| Коэффициент L1-регуляризации | $[10^{-2}, 1]$ |
| Коэффициент L2-регуляризации | $[10^{-2}, 1]$ |

Таблица 5: Оптимизируемые гиперпараметеры алгоритма XGBoost и рассмотренные их значения.

| **Гиперпараметр** | **Значения** |
|---|---|
| Темп обучения | $[10^{-6}, 5 \cdot 10^{-1}]$ |
| Коэффициент L2-регуляризации | $[10^{-6}, 10^{-1}]$ |
| Размер батча | $\{64, 128, 256, 512, 1024\}$ |
| Размерность векторного представления признаков | $\{5, 8, 10, 16, 32, 64, 128\}$ |

Таблица 6: Оптимизируемые общие параметеры обучения и рассмотренные их значения.

Далее с использованием наилучших гиперпараметров проводится 15 запусков обучения конкретной конфигурации модели и преобразования признаков на обучающей выборке и оценки на тестовой выборке, с использованием разных затравок генератора случайных чисел. Максимальное число эпох обучения – 100, используется ранний останов через 20 эпох в случае отсутствия улучшения качества на валидации. Итоговые показатели качества усредняются по всем запускам. В случае задач бинарной классификации в качестве оптимизируемой функции потерь использовалась бинарная кросс-энтропия, в случае задач регрессии – корень из среднеквадратичной ошибки ($RMSE$). Для оптимизации используется алгоритм $Adam$[34]. Показатель качества для бинарной классификации – $ROC\text{-}AUC$, для регрессии – $RMSE$. Также для сравнения проводились аналогичные запуски алгоритма градиентного бустинга в реализации XGBoost[35] как одного из самых эффективных алгоритмов машинного обучения на табличных данных на данный момент[1].

## 5.5 Результаты экспериментов и анализ

Результаты вычислительных экспериментов с рассмотренными моделями и способами преобразования признаков приведены в следующих таблицах. Условные обозначения наборов данных, моделей и методов предобработки соответствуют принятым и описанным выше.

| | **GSC** | **CM** | **VLD** | **AI** | **HL** | **FE** | | **GSC** | **CM** | **VLD** | **AI** | **HL** | **FE** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **M+E+P** | **0.849** | 0.854 | 0.660 | **0.915** | **0.846** | 0.847 | **M+ARM+P** | **0.846** | 0.850 | 0.657 | **0.914** | 0.841 | **0.847** |
| **M+E+F** | 0.847 | 0.854 | **0.662** | 0.914 | 0.843 | 0.837 | **M+ARM+F** | **0.846** | 0.851 | 0.656 | 0.911 | 0.839 | 0.842 |
| **M+E+L** | 0.835 | 0.858 | 0.659 | 0.912 | 0.833 | 0.844 | **M+ARM+L** | 0.837 | 0.856 | **0.660** | 0.912 | 0.839 | 0.843 |
| **M+E+AD** | 0.830 | **0.863** | 0.658 | 0.909 | **0.846** | **0.851** | **M+ARM+AD** | 0.825 | 0.859 | 0.656 | 0.910 | 0.840 | 0.845 |
| **M+E+E** | 0.833 | 0.860 | 0.653 | 0.913 | 0.837 | 0.843 | **M+ARM+E** | 0.831 | **0.862** | 0.651 | 0.909 | **0.842** | 0.839 |
| **M+E+SE** | 0.833 | 0.860 | 0.659 | 0.914 | 0.834 | 0.841 | **M+ARM+SE** | 0.826 | 0.857 | 0.653 | 0.906 | 0.835 | 0.833 |
| **M+E+ARM** | 0.831 | 0.857 | 0.660 | 0.906 | 0.843 | 0.842 | **M+ARM+ARM** | 0.830 | 0.857 | 0.647 | 0.902 | 0.840 | 0.834 |
| **M+E+TT** | 0.827 | 0.855 | 0.656 | 0.908 | 0.840 | 0.841 | **M+ARM+TT** | 0.822 | 0.853 | 0.643 | 0.903 | 0.838 | 0.837 |
| **R+E+P** | 0.845 | **0.871** | 0.659 | **0.916** | **0.841** | 0.844 | **R+ARM+P** | 0.844 | **0.872** | **0.651** | **0.920** | 0.828 | **0.846** |
| **R+E+F** | **0.847** | 0.867 | 0.656 | 0.914 | 0.832 | **0.846** | **R+ARM+F** | **0.845** | 0.870 | 0.649 | 0.913 | 0.824 | 0.842 |
| **R+E+L** | 0.826 | **0.871** | 0.658 | 0.906 | 0.827 | 0.841 | **R+ARM+L** | 0.837 | 0.866 | 0.646 | 0.909 | 0.830 | 0.838 |
| **R+E+AD** | 0.828 | 0.830 | 0.651 | 0.911 | 0.816 | 0.845 | **R+ARM+AD** | 0.831 | 0.846 | 0.649 | 0.914 | 0.821 | 0.842 |
| **R+E+E** | 0.830 | 0.855 | 0.663 | 0.907 | 0.832 | 0.844 | **R+ARM+E** | 0.831 | 0.857 | **0.651** | 0.909 | 0.830 | 0.842 |
| **R+E+SE** | 0.838 | 0.869 | **0.665** | **0.916** | 0.839 | 0.842 | **R+ARM+SE** | 0.830 | 0.859 | 0.650 | 0.911 | 0.835 | 0.836 |
| **R+E+ARM** | 0.841 | 0.865 | 0.654 | 0.915 | 0.833 | 0.837 | **R+ARM+ARM** | 0.834 | 0.864 | 0.643 | 0.917 | **0.837** | 0.838 |
| **R+E+TT** | 0.835 | 0.868 | 0.659 | 0.903 | 0.832 | 0.840 | **R+ARM+TT** | 0.837 | 0.862 | 0.638 | 0.914 | 0.833 | 0.834 |
| **T+E+P** | 0.836 | 0.855 | 0.654 | 0.916 | 0.831 | **0.845** | **T+ARM+P** | **0.835** | 0.870 | **0.649** | **0.926** | **0.826** | **0.844** |
| **T+E+F** | **0.838** | **0.861** | 0.658 | **0.926** | 0.835 | 0.836 | **T+ARM+F** | 0.833 | 0.863 | 0.644 | 0.921 | 0.824 | 0.841 |
| **T+E+L** | 0.830 | 0.847 | **0.661** | 0.918 | 0.833 | 0.843 | **T+ARM+L** | 0.832 | 0.867 | 0.647 | 0.910 | 0.824 | 0.838 |
| **T+E+AD** | 0.836 | 0.836 | 0.658 | 0.916 | 0.821 | 0.811 | **T+ARM+AD** | 0.831 | **0.872** | 0.648 | 0.919 | 0.820 | 0.835 |
| **T+E+E** | 0.829 | 0.844 | **0.661** | 0.914 | **0.840** | 0.844 | **T+ARM+E** | 0.832 | 0.867 | 0.646 | 0.918 | 0.823 | 0.837 |
| **T+E+SE** | 0.833 | 0.818 | 0.610 | 0.894 | 0.805 | 0.803 | **T+ARM+SE** | 0.830 | 0.820 | 0.643 | 0.899 | 0.816 | 0.827 |
| **T+E+ARM** | 0.830 | 0.837 | 0.641 | 0.911 | 0.837 | 0.838 | **T+ARM+ARM** | 0.827 | 0.837 | 0.641 | 0.905 | **0.826** | 0.834 |
| **T+E+TT** | 0.825 | 0.839 | 0.633 | 0.916 | 0.839 | 0.842 | **T+ARM+TT** | 0.827 | 0.831 | 0.627 | 0.901 | 0.821 | 0.827 |

Таблица 7: Результаты для задач бинарной классификации. Схема преобразования признаков – раздельное построение представлений для числовых и категориальных переменных. Левый столбец – модель + способ обработки категориальных признаков + способ преобразования числовых признаков. Для категориальных переменных рассмотрены методы **E** и **ARM**. Показатель качества – *ROC-AUC*.

| | **GSC** | **CM** | **VLD** | **AI** | **HL** | **FE** |
|---|---|---|---|---|---|---|
| **M+TT+P** | **0.841** | 0.846 | **0.660** | 0.911 | 0.836 | **0.844** |
| **M+TT+F** | 0.837 | 0.848 | 0.653 | **0.913** | **0.840** | 0.841 |
| **M+TT+L** | 0.836 | 0.844 | 0.658 | **0.913** | 0.838 | 0.840 |
| **M+TT+AD** | 0.832 | 0.851 | 0.652 | 0.908 | **0.840** | 0.842 |
| **M+TT+E** | 0.835 | **0.854** | 0.656 | 0.909 | 0.836 | 0.837 |
| **M+TT+SE** | 0.829 | 0.849 | 0.648 | 0.901 | 0.827 | 0.833 |
| **M+TT+ARM** | 0.831 | 0.840 | 0.636 | 0.901 | 0.835 | 0.836 |
| **M+TT+TT** | 0.828 | 0.843 | 0.639 | 0.903 | 0.833 | 0.831 |
| **R+TT+P** | **0.840** | **0.865** | 0.646 | **0.915** | **0.831** | **0.842** |
| **R+TT+F** | 0.836 | 0.858 | 0.643 | 0.910 | 0.822 | 0.837 |
| **R+TT+L** | 0.833 | 0.861 | **0.649** | 0.904 | 0.827 | 0.833 |
| **R+TT+AD** | 0.827 | 0.844 | 0.647 | 0.911 | 0.823 | 0.835 |
| **R+TT+E** | 0.829 | 0.851 | 0.642 | 0.905 | 0.830 | 0.838 |
| **R+TT+SE** | 0.824 | 0.848 | 0.639 | 0.908 | 0.825 | 0.833 |
| **R+TT+ARM** | 0.835 | 0.857 | 0.636 | 0.903 | 0.829 | 0.836 |
| **R+TT+TT** | 0.833 | 0.849 | 0.631 | 0.905 | 0.824 | 0.830 |
| **T+TT+P** | 0.830 | 0.866 | **0.645** | **0.925** | **0.820** | 0.839 |
| **T+TT+F** | **0.831** | 0.866 | 0.636 | 0.919 | 0.818 | **0.840** |
| **T+TT+L** | 0.830 | 0.868 | 0.640 | 0.921 | **0.820** | 0.836 |
| **T+TT+AD** | 0.826 | 0.865 | 0.637 | 0.911 | 0.816 | 0.832 |
| **T+TT+E** | 0.828 | **0.869** | 0.637 | 0.917 | 0.813 | 0.839 |
| **T+TT+SE** | 0.826 | 0.813 | 0.622 | 0.903 | 0.808 | 0.825 |
| **T+TT+ARM** | 0.824 | 0.840 | 0.633 | 0.906 | 0.817 | 0.827 |
| **T+TT+TT** | 0.821 | 0.819 | 0.614 | 0.904 | 0.818 | 0.815 |

Таблица 8: Результаты для задач бинарной классификации. Схема преобразования признаков – раздельное построение представлений для числовых и категориальных переменных. Для категориальных переменных рассмотрен метод **TT**. Левый столбец – модель + способ обработки категориальных признаков + способ преобразования числовых признаков. Показатель качества – *ROC-AUC*.

| | BF | BH | BP | CL | HS | NT | | BF | BH | BP | CL | HS | NT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **M+E+P** | **3493.315** | **161.490** | **28.474** | **0.152** | **103696.648** | 1.212 | **M+ARM+P** | **3470.916** | 191.011 | **30.842** | 0.153 | **107594.007** | **1.239** |
| **M+E+F** | 3509.304 | 163.841 | 32.186 | 0.154 | 118521.179 | **1.211** | **M+ARM+F** | 3526.128 | 205.362 | 33.882 | 0.149 | 111992.492 | 1.247 |
| **M+E+L** | 3523.345 | 168.196 | 33.070 | 0.155 | 111275.718 | 1.254 | **M+ARM+L** | 3549.019 | 190.726 | 35.050 | 0.151 | 124202.234 | 1.261 |
| **M+E+AD** | 3537.696 | 178.859 | 32.367 | 0.160 | 107769.546 | 1.230 | **M+ARM+AD** | 3630.828 | 197.681 | 33.302 | **0.144** | 111069.359 | 1.256 |
| **M+E+E** | 3619.380 | 172.832 | 38.913 | 0.156 | 114695.625 | 1.231 | **M+ARM+E** | 3647.994 | 204.327 | 37.634 | 0.150 | 107999.406 | 1.252 |
| **M+E+SE** | 3499.366 | 170.916 | 30.843 | 0.153 | 110035.023 | 1.232 | **M+ARM+SE** | 3536.332 | 195.563 | 33.047 | 0.146 | 110888.671 | 1.260 |
| **M+E+ARM** | 3501.736 | 164.054 | 30.807 | 0.155 | 110183.651 | 1.239 | **M+ARM+ARM** | 3627.965 | **189.108** | 33.641 | 0.151 | 115274.288 | 1.257 |
| **M+E+TT** | 3604.567 | 170.901 | 33.155 | 0.160 | 111604.987 | 1.286 | **M+ARM+TT** | 3640.616 | 193.404 | 33.979 | 0.159 | 117376.018 | 1.283 |
| **R+E+P** | **3488.928** | 168.781 | **32.460** | 0.158 | 107012.937 | 1.239 | **R+ARM+P** | **3490.062** | 169.003 | **31.974** | 0.158 | **110273.745** | 1.251 |
| **R+E+F** | 3527.007 | 173.408 | 33.739 | 0.156 | 111822.898 | 1.231 | **R+ARM+F** | 3569.210 | 174.998 | 34.172 | 0.157 | 114068.130 | **1.246** |
| **R+E+L** | 3508.464 | 173.198 | 34.949 | 0.148 | 110362.132 | 1.247 | **R+ARM+L** | 3549.559 | 171.348 | 36.301 | 0.158 | 111988.096 | 1.253 |
| **R+E+AD** | 3543.546 | 162.755 | 35.697 | 0.151 | 112896.429 | 1.235 | **R+ARM+AD** | 3548.092 | **167.319** | 35.997 | **0.154** | 114103.738 | 1.248 |
| **R+E+E** | 3527.765 | 175.601 | 33.197 | 0.151 | **106084.085** | 1.233 | **R+ARM+E** | 3570.877 | 179.014 | 33.527 | 0.156 | 112440.102 | 1.249 |
| **R+E+SE** | 3530.290 | 166.651 | 34.950 | **0.149** | 107613.945 | **1.227** | **R+ARM+SE** | 3573.114 | 175.851 | 35.226 | 0.157 | 113914.041 | 1.252 |
| **R+E+ARM** | 3530.263 | **161.750** | 33.856 | 0.154 | 112302.484 | 1.229 | **R+ARM+ARM** | 3570.452 | 177.749 | 36.719 | 0.163 | 114833.758 | 1.254 |
| **R+E+TT** | 3600.735 | 173.314 | 34.948 | 0.157 | 113477.552 | 1.237 | **R+ARM+TT** | 3640.399 | 179.614 | 36.185 | 0.159 | 116003.992 | 1.252 |
| **T+E+P** | **3523.649** | 203.135 | **39.032** | 0.152 | 229517.421 | **1.271** | **T+ARM+P** | 3593.316 | **207.142** | **41.103** | 0.159 | **235093.144** | **1.274** |
| **T+E+F** | 3598.416 | **188.301** | 42.667 | 0.153 | 233169.062 | 1.273 | **T+ARM+F** | 4045.649 | 211.469 | 43.983 | 0.163 | 241803.334 | 1.276 |
| **T+E+L** | 3551.789 | 191.017 | 39.900 | 0.153 | **225858.004** | 1.300 | **T+ARM+L** | 4031.621 | 209.461 | 43.731 | 0.158 | 237528.402 | 1.275 |
| **T+E+AD** | 3583.109 | 198.582 | 41.046 | 0.155 | 234362.781 | 1.350 | **T+ARM+AD** | **3587.422** | 210.004 | 46.754 | **0.154** | 240397.749 | 1.297 |
| **T+E+E** | 3542.064 | 195.884 | 40.941 | **0.149** | 227154.406 | 1.348 | **T+ARM+E** | 3648.697 | 207.689 | 44.051 | 0.157 | 237436.307 | 1.294 |
| **T+E+SE** | 3572.242 | 198.977 | 47.139 | 0.164 | 237729.406 | 1.419 | **T+ARM+SE** | 3894.958 | 217.426 | 48.715 | 0.160 | 245903.774 | 1.376 |
| **T+E+ARM** | 3628.511 | 215.042 | 45.649 | 0.156 | 242694.092 | 1.288 | **T+ARM+ARM** | 4206.715 | 239.470 | 49.003 | 0.162 | 260791.948 | 1.368 |
| **T+E+TT** | 3679.052 | 213.968 | 47.436 | 0.161 | 252307.732 | 1.409 | **T+ARM+TT** | 4401.113 | 273.509 | 54.075 | 0.166 | 261037.474 | 1.371 |

Таблица 9: Результаты для задач регрессии. Схема преобразования признаков – раздельное построение представлений для числовых и категориальных переменных. Левый столбец – модель + способ обработки категориальных признаков + способ преобразования числовых признаков. Для категориальных переменных рассмотрены методы **E** и **ARM**. Показатель качества – $RMSE$.

| | **BF** | **BH** | **BP** | **CL** | **HS** | **NT** |
|---|---|---|---|---|---|---|
| **M+TT+P** | **3527.195** | **199.428** | **32.007** | **0.151** | 112561.143 | **1.240** |
| **M+TT+F** | 3538.741 | 203.649 | 32.659 | 0.153 | 114903.525 | 1.245 |
| **M+TT+L** | 3637.004 | 200.171 | 34.162 | 0.155 | 119748.036 | 1.263 |
| **M+TT+AD** | 3692.388 | 209.684 | 36.486 | **0.151** | 113872.404 | 1.261 |
| **M+TT+E** | 3665.846 | 202.440 | 36.609 | 0.153 | **108332.131** | 1.260 |
| **M+TT+SE** | 3714.741 | 204.918 | 35.891 | 0.156 | 112642.280 | 1.265 |
| **M+TT+ARM** | 3688.910 | 203.701 | 36.744 | 0.158 | 117140.463 | 1.266 |
| **M+TT+TT** | 3800.513 | 216.043 | 38.938 | 0.164 | 121007.536 | 1.289 |
| **R+TT+P** | **3549.785** | **184.720** | **32.142** | **0.156** | 115493.802 | 1.257 |
| **R+TT+F** | 3619.412 | 190.173 | 36.781 | 0.159 | 119046.376 | 1.262 |
| **R+TT+L** | 3631.937 | 192.447 | 36.601 | 0.161 | **114906.728** | **1.254** |
| **R+TT+AD** | 3643.432 | 188.709 | 38.484 | 0.158 | 116379.104 | 1.259 |
| **R+TT+E** | 3695.739 | 206.025 | 35.553 | 0.158 | 118002.131 | 1.261 |
| **R+TT+SE** | 3671.090 | 200.533 | 37.039 | 0.162 | 120748.346 | 1.265 |
| **R+TT+ARM** | 3682.774 | 197.499 | 39.112 | 0.166 | 120966.227 | 1.264 |
| **R+TT+TT** | 3703.496 | 208.541 | 39.056 | 0.164 | 123501.188 | 1.252 |
| **T+TT+P** | 4067.490 | 2365.813 | 85.556 | 0.157 | 288255.625 | **1.245** |
| **T+TT+F** | 4287.995 | 2917.840 | 85.494 | 0.154 | 290846.031 | **1.245** |
| **T+TT+L** | 3969.390 | 3181.036 | **84.739** | 0.161 | 364637.500 | 1.325 |
| **T+TT+AD** | 4282.730 | 3379.895 | 85.520 | 0.167 | 317759.750 | 1.350 |
| **T+TT+E** | **3860.260** | **1859.714** | 86.417 | **0.153** | 259393.875 | 1.347 |
| **T+TT+SE** | 4203.914 | 3824.551 | 85.211 | 0.170 | 339582.062 | 1.362 |
| **T+TT+ARM** | 4611.753 | 3707.824 | 86.397 | 0.156 | 386536.812 | 1.353 |
| **T+TT+TT** | 4872.236 | 3722.973 | 88.047 | 0.174 | **247741.402** | 1.359 |

Таблица 10: Результаты для задач регрессии. Схема преобразования признаков – раздельное построение представлений для числовых и категориальных переменных. Для категориальных переменных рассмотрен метод **TT**. Левый столбец – модель + способ обработки категориальных признаков + способ преобразования числовых признаков. Показатель качества – $RMSE$.

| | **GSC** | **CM** | **VLD** | **AI** | **HL** | **FE** |
|---|---|---|---|---|---|---|
| **M** | 0.823 | 0.834 | 0.616 | 0.861 | 0.821 | 0.833 |
| **M+P** | **0.835** | **0.860** | 0.658 | **0.914** | **0.844** | 0.843 |
| **M+F** | 0.828 | 0.859 | 0.658 | 0.911 | 0.839 | 0.844 |
| **M+L** | 0.830 | 0.857 | **0.659** | 0.911 | 0.834 | 0.843 |
| **M+AD** | 0.830 | 0.857 | 0.655 | 0.912 | 0.839 | 0.844 |
| **M+E** | 0.828 | 0.854 | 0.656 | 0.910 | 0.835 | 0.841 |
| **M+SE** | 0.828 | 0.857 | 0.652 | 0.906 | 0.831 | **0.845** |
| **M+ARM** | 0.826 | 0.851 | 0.633 | 0.901 | 0.835 | 0.838 |
| **M+TT** | 0.819 | 0.847 | 0.632 | 0.896 | 0.830 | 0.834 |
| **R** | 0.822 | 0.827 | 0.619 | 0.877 | 0.813 | 0.831 |
| **R+P** | 0.831 | 0.852 | **0.659** | **0.917** | 0.827 | 0.843 |
| **R+F** | 0.824 | 0.852 | 0.658 | 0.912 | **0.834** | 0.841 |
| **R+L** | **0.832** | 0.853 | 0.655 | 0.915 | 0.825 | 0.840 |
| **R+AD** | 0.831 | 0.852 | 0.646 | 0.912 | 0.819 | 0.841 |
| **R+E** | 0.829 | 0.849 | **0.659** | 0.913 | 0.833 | **0.844** |
| **R+SE** | 0.830 | **0.854** | 0.652 | 0.916 | 0.831 | **0.844** |
| **R+ARM** | 0.829 | 0.844 | 0.640 | 0.916 | 0.825 | 0.839 |
| **R+TT** | 0.825 | 0.831 | 0.638 | 0.914 | 0.823 | 0.833 |
| **T+P** | 0.829 | **0.864** | 0.655 | **0.920** | 0.840 | **0.844** |
| **T+F** | 0.829 | 0.863 | 0.654 | 0.918 | 0.841 | 0.841 |
| **T+L** | **0.859** | 0.857 | 0.658 | 0.914 | 0.837 | 0.842 |
| **T+AD** | 0.824 | 0.854 | 0.659 | 0.918 | 0.842 | 0.817 |
| **T+E** | 0.834 | 0.863 | **0.662** | 0.916 | **0.843** | 0.842 |
| **T+SE** | 0.817 | 0.837 | 0.639 | 0.898 | 0.818 | 0.825 |
| **T+ARM** | 0.816 | 0.841 | 0.636 | 0.912 | 0.814 | 0.822 |
| **T+TT** | 0.804 | 0.837 | 0.622 | 0.905 | 0.818 | 0.817 |
| **GBDT** | **0.867** | 0.861 | **0.662** | 0.875 | **0.844** | 0.837 |

Таблица 11: Результаты для задач бинарной классификации. Схема преобразования признаков – one-hot кодирование для категориальных, нормализация для числовых, рассмотрение всех признаков как непрерывных числовых и построение векторных представлений одним из способов. Левый столбец – модель + способ преобразования переменных. Также приведены результаты для градиентного бустинга XGBoost на исходных данных. Показатель качества – $ROC\text{-}AUC$.

| | **BF** | **BH** | **BP** | **CL** | **HS** | **NT** |
|---|---|---|---|---|---|---|
| **M** | 3961.807 | 301.745 | 47.086 | 0.167 | 133462.307 | 1.415 |
| **M+P** | **3627.419** | **180.404** | **33.740** | **0.158** | 121498.483 | 1.371 |
| **M+F** | 3650.254 | 184.007 | 33.997 | 0.160 | 130905.040 | 1.382 |
| **M+L** | 3687.114 | 192.030 | 34.026 | **0.158** | 128700.091 | 1.399 |
| **M+AD** | 3703.503 | 200.631 | 33.908 | 0.165 | 124558.099 | 1.386 |
| **M+E** | 3749.900 | 196.036 | 37.750 | 0.168 | **119471.142** | 1.347 |
| **M+SE** | 3674.747 | 193.168 | 36.604 | 0.161 | 122835.572 | **1.344** |
| **M+ARM** | 3690.753 | 187.413 | 35.994 | 0.160 | 124037.717 | 1.364 |
| **M+TT** | 3784.740 | 203.588 | 38.708 | 0.164 | 128595.449 | 1.357 |
| **R** | 3913.094 | 289.103 | 42.622 | 0.159 | 132782.191 | 1.395 |
| **R+P** | **3501.659** | 189.285 | **32.117** | 0.154 | **111714.195** | 1.208 |
| **R+F** | 3523.423 | 175.311 | 34.247 | **0.146** | 114955.414 | **1.202** |
| **R+L** | 3547.485 | **142.581** | 36.700 | 0.149 | 117986.023 | 1.218 |
| **R+AD** | 3531.634 | 178.030 | 34.008 | 0.150 | 122512.789 | 1.240 |
| **R+E** | 3603.856 | 143.944 | 38.509 | 0.149 | 124784.882 | 1.277 |
| **R+SE** | 3532.988 | 170.441 | 35.448 | 0.153 | 115471.070 | 1.234 |
| **R+ARM** | 3580.760 | 183.049 | 36.004 | 0.157 | 119022.418 | 1.323 |
| **R+TT** | 3613.436 | 179.674 | 36.412 | 0.162 | 122487.043 | 1.307 |
| **T+P** | 3778.848 | 243.511 | 42.595 | **0.157** | **258092.883** | 1.347 |
| **T+F** | 3807.135 | 260.114 | 47.082 | 0.163 | 274303.071 | **1.342** |
| **T+L** | 3794.658 | **212.497** | **40.784** | 0.160 | 263669.947 | 1.374 |
| **T+AD** | 3800.694 | 258.115 | 44.719 | 0.162 | 269610.004 | 1.392 |
| **T+E** | **3763.341** | 215.500 | 45.507 | 0.159 | 272003.898 | 1.383 |
| **T+SE** | 3785.462 | 226.440 | 46.662 | 0.169 | 290017.663 | 1.369 |
| **T+ARM** | 3868.442 | 277.501 | 46.424 | 0.164 | 295048.284 | 1.378 |
| **T+TT** | 3940.757 | 278.094 | 48.003 | 0.169 | 297742.285 | 1.347 |
| **GBDT** | **3436.936** | 236.527 | **31.137** | **0.139** | **97989.352** | **1.135** |

Таблица 12: Результаты для задач регрессии. Схема преобразования признаков – one-hot кодирование для категориальных, нормализация для числовых, рассмотрение всех признаков как непрерывных числовых и построение векторных представлений одним из способов. Левый столбец – модель + способ преобразования переменных. Также приведены результаты для градиентного бустинга XGBoost на исходных данных. Показатель качества – $RMSE$.

| | GSC | CM | VLD | AI | HL | FE |
|---|---|---|---|---|---|---|
| **M+E** | **0.853** | **0.853** | 0.619 | 0.877 | **0.813** | **0.831** |
| **M+ARM** | 0.849 | 0.850 | **0.622** | **0.879** | 0.808 | 0.828 |
| **M+TT** | 0.847 | 0.849 | 0.613 | 0.871 | 0.805 | 0.829 |
| **R+E** | **0.862** | 0.847 | **0.654** | **0.891** | 0.828 | **0.839** |
| **R+ARM** | 0.860 | **0.849** | 0.607 | 0.887 | **0.830** | 0.836 |
| **R+TT** | 0.855 | 0.844 | 0.612 | 0.884 | 0.828 | 0.828 |
| **T+E** | **0.862** | 0.860 | **0.648** | 0.891 | 0.835 | 0.836 |
| **T+ARM** | 0.858 | 0.859 | 0.642 | 0.892 | 0.839 | **0.837** |
| **T+TT** | **0.862** | **0.862** | 0.637 | **0.894** | **0.843** | 0.831 |

Таблица 13: Результаты для задач бинарной классификации. Схема преобразования признаков – дискретизация числовых переменных по квантилям, рассмотрение всех признаков как дискретных категориальных и построение векторных представлений одним из способов (**E**, **ARM**, **TT**). Левый столбец – модель + способ преобразования признаков. Показатель качества – $ROC\text{-}AUC$.

| | BF | BH | BP | CL | HS | NT |
|---|---|---|---|---|---|---|
| **M+E** | **3863.029** | **931.066** | 50.115 | **0.158** | 133904.590 | **1.440** |
| **M+ARM** | 3902.487 | 978.482 | **48.917** | 0.161 | **130077.147** | 1.449 |
| **M+TT** | 3946.793 | 1101.659 | 49.601 | 0.163 | 131783.513 | 1.513 |
| **R+E** | **3861.725** | **914.779** | 48.436 | 0.161 | 131053.312 | **1.441** |
| **R+ARM** | 3902.750 | 953.038 | **44.242** | **0.154** | 122326.723 | 1.454 |
| **R+TT** | 3861.317 | 1007.080 | 49.988 | 0.158 | **109631.121** | 1.466 |
| **T+E** | **3863.002** | 4610.128 | 52.558 | **0.151** | **365614.028** | 1.457 |
| **T+ARM** | 3893.639 | 4535.657 | 52.293 | 0.152 | 389763.279 | **1.452** |
| **T+TT** | 4072.670 | **4460.844** | **50.889** | 0.157 | 412501.801 | 1.470 |

Таблица 14: Результаты для задач регрессии. Схема преобразования признаков – дискретизация числовых переменных по квантилям, рассмотрение всех признаков как дискретных категориальных и построение векторных представлений одним из способов (**E**, **ARM**, **TT**). Левый столбец – модель + способ преобразования признаков. Показатель качества – $RMSE$.

Стоит отметить, что стандартные отклонения результатов принимали значения примерно на 3 – 4 порядка меньше усредненных значений показателей качества (т. е. отличались от представленных в таблицах результатов значений примерно в $10^{-4}$ – $10^{-3}$ раз). В целом из проведенных исследований и их результатов можно сделать следующие выводы и отметить следующие наблюдения:

- В полученных экспериментально результатах в целом сложно выделить явные и значительные различия или закономерности в рассмотренных методах преобразований признаков, показатели качества в общем схожи и близки, в том числе и на разных данных. В целом это согласуется с результатами, полученными в исследованиях по этой теме[4, 6]. Однако использование предобработки признаков в большинстве случаев оказывается полезным и повышает качество предсказаний (табл. 11, табл. 12).
- Также сложно выделить наилучшую архитектуру нейронной сети по итогу проведенных экспериментов, соотношения между результатами несколько различаются в зависимости от задачи, набора данных, схемы и способов предобработки признаков. Схожие наблюдения присутствуют в ряде работ[4, 7, 6], можно сказать, что, вероятно, в рассмотренных задачах преимущества архитектуры играют небольшую роль. Причиной этому могли стать также и особенности данных, их принципиальная разнородность.

  Для многослойного перцептрона результаты достаточно стабильны, не наблюдается радикальных различий в качестве прогнозов, и в целом оно достаточно высоко. Также качество предсказаний для него слабее всего меняется в зависимости от способа преобразования признаков. Это может быть связано с относительной простотой данной архитектуры по сравнению с остальными рассмотренными и, соответственно, более простой процедурой её обучения и меньшей ее склонности к переобучению. Также многослойный перцептрон имеет заметно меньшее время работы и обучения, нежели ResNet и трансформер.

  Качество прогнозов ResNet не так стабильно относительно конфигурации данных и способа предобработки, результаты больше колеблются, однако в некоторых ситуациях для нее достигалось наилучшее среди всех моделей значение исследуемых показателей (табл. 7, табл. 9, табл. 10, табл. 12, табл. 13,

табл. 14). Вероятно, это может быть связано с тем, что используемые в ней прокидывания связей действительно позволяют лучше моделировать и использовать комбинации признаков различного уровня, часто оказывающиеся ключевыми во многих табличных задачах. В некоторых случаях ResNet имеет досточно большое время обучения и предсказания.

Модель трансформер же имеет наименее стабильное поведение на различных наборах данных и при использовании разнообразных методов их предобработки. Данная архитектура является наиболее сложной и перепараметризованной относительно остальных, и, возможно, в силу этого и с учетом сложного устройства табличных данных трансформер достаточно плохо обучается на них. Однако в некоторых ситуациях удается получить весьма неплохие результаты среди всех моделей, порой и наилучшие с определенным отрывом (табл. 7, табл. 8, табл. 11, табл. 13, табл. 14). При этом в задачах бинарной классификации показатели качества для трансформеров гораздо лучше, нежели для задачи регрессии, где часто наблюдается совершенно неудовлетворительно большое значение среднеквадратичной ошибки (табл. 9, табл. 10, табл. 12, табл. 14). На некоторых отдельных наборах данных качество прогнозов трансформера на порядки хуже результатов остальных моделей. Также модель трансформер наименее стабильна и чувствительна к данным и способам предобработки признаков, а также имеет значительное время обучения.

- Среди схем преобразования признаков наилучшей оказалось раздельное построение векторных представлений для числовых и категориальных переменных, что вполне логично и ожидаемо (табл. 7, табл. 8, табл. 9, табл. 10). Данный способ меньше всего искажает первоначальное значение признаков и сохраняет их принципиальное различие, и это положительно сказывается на результате. Рассмотрение одного типа признаков как принципиально другого приводит к ухудшению качества и не позволяет достичь улучшений за счет возможного учета взаимосвязей между переменными. Особенно это проявляется в задачах регрессии (табл. 12, табл. 14).

- Что касается наилучшего способа построения самих векторных представлений признаков, то во многих экспериментах самое высокое качество достигалось при использовании периодических функций синуса и косинуса, причем как с обучаемыми, так и необучаемыми коэффициентами. Данный способ оказался наиболее эффективным для различных архитектур, данных и задач, почти во всех случаях. Это может подтверждать предположение о том, что в табличных данных в целом, в силу неоднородности признаков, целевые зависимости довольно сложные, во многом негладкие и высокочастотные, и применение данного вида преобразования признаков, вероятно, позволяет учесть эти факторы, что позволяет добиться хороших результатов. Также данный метод предобработки довольно прост относительно остальных рассмотренных, не так сильно усложняет модель в целом. Вероятно, целесообразно исследовать данный способ преобразования признаков детальнее с целью еще большего повышения качества результатов нейросетевых моделей на табличных данных.
- Использование механизмов внимания и модели трансформера для построения представлений признаков в основном приводит к падению качества предсказаний. По сути, в таком способе используется еще одна полноценная модель помимо основной архитектуры, общая модель заметно усложняется, и ее становится сложнее полноценно обучить. Особенно это заметно при использовании этого способа предобработки с основной архитектурой трансформером (табл. 8, табл. 10, табл. 11, табл. 12).
- Метод преобразования ARM также не достигает высокого качества на фоне остальных способов, его применение также, судя по всему, усложняет модель. Использование экспоненциальных нейронов не оказывает положительного влияния на качество прогнозов. Вероятно, моделируемые в данном методе нелинейные и мультипликативные взаимодействия между признаками оказываются не столь важными и сильными, и их использование не вносит улучшений на фоне ухудшения качества из-за излишнего усложнения моделей. То же можно сказать и про построение векторных представлений с помощью отдельного трансформера. Возможно, моделировать нелинейные взаимодействия между признаками окажется эффективнее с использованием иных подходов.

- Преобразование категориальных признаков с помощью обучаемой таблицы представлений оказалось наиболее эффективным при раздельной обработке числовых и категориальных переменных (табл. 7, табл. 9). Данный способ в целом намного интуитивнее и проще, чем ARM или использование отдельного трансформера. Он никак не моделирует взаимодействия между категориальными переменными, но отчасти его хорошее качество можно объяснить тем, что на практике признаки в таблицах не имеют порядка и редко оказываются хоть как-либо связаны, т. е. в некоторой степени их можно считать независимыми в большинстве случаев. Также использование таблицы векторов намного проще и не так сильно увеличивает время обучения и работы модели.
- В целом можно сказать, что для каждого из способов построения векторных представлений хотя бы раз достигалось наилучшее качество для конкретных данных, модели и схемы предобработки. Таким образом, их существование и использование оправдано, и их эффективность во многом зависит от данных, задачи, используемой архитектуры и т. д.
- В ходе оптимизации гиперпараметров наилучшие их конфигурации для различных моделей часто оказывались схожими (размер батча, темп обучения, размер векторного представления признаков, коэффициенты регуляризации и т. д.), и в целом качество на разных гиперпараметрах отличалось не так сильно. Также использование расписаний темпа обучения и варьирование числа итераций раннего останова почти не влияли на результат.
- Использование позиционного кодирования признаков (как в работе[5] и т. п.) почти никак не влияло на качество прогнозов, что подтверждает предположение о неупорядоченности и слабых взаимодействиях между переменными в табличных данных.
- Использование проекций признаков на набор представлений и метода AutoDis применительно к трансформеру в большинстве случаев приводило к заметно низким показателям качества, особенно в задачах бинарной классификации (табл. 7, табл. 8, табл. 11), и это очень слабо решалось варьированием коэффициента температуры, размера таблицы представлений и т. д. Причины такого эффекта в целом однозначно определить затруднительно, вероятно, построение

входных векторов признаков с помощью взвешенной суммы нескольких обучаемых приводит к большей потере информации о переменных для последующей их обработки трансформером, или же данный способ излишне усложняет модель.

- Использование специальных методов преобразования табличных признаков позволило в многих случаях превзойти качество градиентного бустинга на тех же данных (табл. 11, табл. 12), что оправдывает использование и разработку методов предобработки табличных данных для моделей глубокого обучения. Однако алгоритм градиентного бустинга по-прежнему порой превосходит нейросетевые модели на таблицах, что особенно заметно в результатах экспериментов для задач регрессии (табл. 12).
- Преобразования признаков в табличных данных способствуют повышению качества предсказаний моделей глубокого обучения, однако величина улучшений часто не столь велика. Согласно ряду работ, существуют и используются другие подходы к адаптации нейросетей к таблицам, например, построение композиций и гибридных архитектур с классическими алгоритмами машинного обучения (чаще всего линейной регрессией и решающими деревьями), как в моделях TabNet[8], NODE[36] и т. п., или использование обучаемых регуляризаций сети[37, 38, 39, 40], и в ряде случаев они оказываются более эффективными с точки зрения итогового качества моделей.

Данные и код реализации всех проведенных экспериментов доступны в репозитории [41].

# 6 Заключение

В данной работе были исследованы методы предобработки табличных данных для наиболее популярных архитектур глубокого обучения. Были проведены вычислительные эксперименты на различных наборах данных применительно к задачам бинарной классификации и регрессии для сравнения и анализа рассмотренных способов преобразования признаков. Использование специальной предобработки таблич-

ных данных позволило повысить качество предсказаний нейросетевых архитектур, а в некоторых ситуациях и превзойти результаты алгоритма градиентного бустинга, являющегося наиболее эффективным алгоритмом в задачах с таблицами в настоящий момент. Таким образом, предобработка данных в формате таблиц для моделей глубокого обучения оправдана и демонстрирует свою эффективность, приводит к улучшению качества прогнозов. Наиболее эффективным подходом к преобразованию признаков в результате проведенных исследований оказалось раздельное построение представлений для числовых и категориальных переменных, наиболее эффективными способами формирования самих представлений оказались метод с использованием периодических функций синуса и косинуса для числовых признаков и использование обучаемой таблицы представлений для категориальных. В целом актуальным видится дальнейшее исследование данных методов с целью получения более высоких показателей качества.

На защиту выносится:

- Реализация рассмотренных методов предобработки табличных данных для различных моделей глубокого обучения.
- Проведение вычислительных экспериментов для сравнения методов преобразования признаков применительно к множеству задач, нейросетевых архитектур, наборов данных из различных областей.
- Получены конфигурации преобразования признаков и моделей глубокого обучения, качество которых превосходит качество градиентного бустинга, являющегося наиболее эффективным алгоритмом на табличных данных на данный момент.

# Список литературы


1. *Grinsztajn L.*, *Oyallon E.*, *Varoquaux G.* Why do tree-based models still outperform deep learning on typical tabular data? // Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track. — 2022.
2. *Shwartz-Ziv R.*, *Armon A.* Tabular Data: Deep Learning is Not All You Need // 8th ICML Workshop on Automated Machine Learning (AutoML). — 2021.
3. *Borisov V.* [et al.]. Deep Neural Networks and Tabular Data: A Survey // ArXiv. — 2021. — Vol. abs/2110.01889.
4. *Marais J.* Deep learning for tabular data : an exploratory study. — 2019.
5. *Vaswani A.* [et al.]. Attention is All you Need // ArXiv. — 2017. — Vol. abs/1706.03762.
6. *Gorishniy Y.*, *Rubachev I.*, *Babenko A.* On Embeddings for Numerical Features in Tabular Deep Learning // Advances in Neural Information Processing Systems / ed. by A. H. Oh, A. Agarwal, D. Belgrave, K. Cho. — 2022.
7. *Gorishniy Y.*, *Rubachev I.*, *Khrulkov V.*, *Babenko A.* Revisiting Deep Learning Models for Tabular Data // Advances in Neural Information Processing Systems / ed. by A. Beygelzimer, Y. Dauphin, P. Liang, J. W. Vaughan. — 2021.
8. *Arik S. O.*, *Pfister T.* TabNet: Attentive Interpretable Tabular Learning. — 2020.
9. *Huang X.*, *Khetan A.*, *Cvitkovic M.*, *Karnin Z.* TabTransformer: Tabular Data Modeling Using Contextual Embeddings. — 2020. — Dec.
10. *Song W.* [et al.]. AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks. — 2019. — Nov.
11. *Somepalli G.* [et al.]. SAINT: Improved Neural Networks for Tabular Data via Row Attention and Contrastive Pre-Training. — 2022.
12. *Klambauer G.*, *Unterthiner T.*, *Mayr A.*, *Hochreiter S.* Self-Normalizing Neural Networks. — 2017. — June.

13. *He K.*, *Zhang X.*, *Ren S.*, *Sun J.* Deep Residual Learning for Image Recognition // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). — 2015. — P. 770–778.

14. *Hancock J.*, *Khoshgoftaar T.* Survey on categorical data for neural networks // Journal of Big Data. — 2020. — Apr. — Vol. 7.

15. *Guo C.*, *Berkhahn F.* Entity Embeddings of Categorical Variables // ArXiv. — 2016. — Vol. abs/1604.06737.

16. *Guo H.* [et al.]. An Embedding Learning Framework for Numerical Features in CTR Prediction. — 2021. — Aug.

17. *Li Y.* Time-Dependent Representation for Neural Event Sequence Prediction. — 2017. — July.

18. *Li Y.* [et al.]. Learnable Fourier Features for Multi-dimensional Spatial Positional Encoding // Advances in Neural Information Processing Systems / ed. by A. Beygelzimer, Y. Dauphin, P. Liang, J. W. Vaughan. — 2021.

19. *Tancik M.* [et al.]. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. — 2020. — June.

20. *Rahimi A.*, *Recht B.* Random Features for Large-Scale Kernel Machines. — 2007.

21. *Xie J.*, *Liu F.*, *Wang K.*, *Huang X.* Deep Kernel Learning via Random Fourier Features // ArXiv. — 2019. — Vol. abs/1910.02660.

22. *Cai S.* [et al.]. ARM-Net: Adaptive Relation Modeling Network for Structured Data // Proceedings of the 2021 International Conference on Management of Data. — 2021.

23. *Cybenko G. V.* Approximation by superpositions of a sigmoidal function // Mathematics of Control, Signals and Systems. — 1989. — Vol. 2. — P. 303–314.

24. *Hines J.* A Logarithmic Neural Network Architecture for Unbounded Non-Linear Function Approximation. — 1970. — Feb. — DOI: 10.1109/ICNN.1996.549076.

25. *Peters B.*, *Niculae V.*, *Martins A. F. T.* Sparse Sequence-to-Sequence Models // ArXiv. — 2019. — Vol. abs/1905.05702.

26. *Tezekbayev M.*, *Nikoulina V.*, *Gallé M.*, *Assylbekov Z.* Speeding Up Entmax // ArXiv. — 2021. — Vol. abs/2111.06832.

27. *Ba J.*, *Kiros J. R.*, *Hinton G. E.* Layer Normalization // ArXiv. — 2016. — Vol. abs/1607.06450.

28. *Paszke A.* [et al.]. PyTorch: An Imperative Style, High-Performance Deep Learning Library // ArXiv. — 2019. — Vol. abs/1912.01703.

29. Kaggle, an online community of data scientists and machine learning practitioners. — `https://www.kaggle.com`.

30. OpenML, an open platform for sharing datasets, algorithms, and experiments. — `https://www.kaggle.com`.

31. *Ioffe S.*, *Szegedy C.* Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift // International Conference on Machine Learning. — 2015.

32. *Srivastava N.* [et al.]. Dropout: a simple way to prevent neural networks from overfitting // J. Mach. Learn. Res. — 2014. — Vol. 15. — P. 1929–1958.

33. *Akiba T.* [et al.]. Optuna: A Next-generation Hyperparameter Optimization Framework // Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. — 2019.

34. *Kingma D. P.*, *Ba J.* Adam: A Method for Stochastic Optimization // CoRR. — 2014. — Vol. abs/1412.6980.

35. *Chen T.*, *Guestrin C.* XGBoost: A Scalable Tree Boosting System // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. — 2016.

36. *Popov S.*, *Morozov S.*, *Babenko A.* Neural Oblivious Decision Ensembles for Deep Learning on Tabular Data // ArXiv. — 2019. — Vol. abs/1909.06312.

37. *Lounici K.*, *Meziani K.*, *Riu B.* Muddling Label Regularization: Deep Learning for Tabular Datasets // ArXiv. — 2021. — Vol. abs/2106.04462.

38. *Valdes G.*, *Arbelo-González W.*, *Interian Y.*, *Friedman J.* Lockout: Sparse Regularization of Neural Networks. — 2021. — July.

39. *Shavitt I.*, *Segal E.* Regularization Learning Networks: Deep Learning for Tabular Datasets // Neural Information Processing Systems. — 2018.

40. *Kadra A.*, *Lindauer M. T.*, *Hutter F.*, *Grabocka J.* Well-tuned Simple Nets Excel on Tabular Datasets // Neural Information Processing Systems. — 2021.

41. *Miheev B.* Preprocessing tabular data for deep machine learning, work repository. — URL: `https://github.com/ezzbreezn/research_seminar_4course`.