ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Департамент программной инженерии

УДК 004.8

СОГЛАСОВАНО

Научный руководитель,
старший преподаватель
департамента больших данных
и информационного поиска
Факультета компьютерных наук

_____ Е. А. Соколов
«_____» _____ 2022 г.

УТВЕРЖДАЮ

Академический руководитель
образовательной программы
«Программная инженерия»,
профессор департамента программной
инженерии, канд. техн. наук

_____ В. В. Шилов
«_____» _____ 2022 г.

Выпускная квалификационная работа
(академическая)

на тему: **Исследование нейросетевых архитектур для табличных данных**

по направлению подготовки 09.03.04 «Программная инженерия»

ВЫПОЛНИЛ

студент группы БПИ181
образовательной программы
09.03.04 «Программная инженерия»

_____ Е. Д. Плющ
«_____» _____ 2022 г.

Москва 2022

Реферат

В последние годы машинное обучение активно используется во всех сферах для различных целей, и зачастую входными данными являются табличные данные. Для табличных данных в индустрии принято использовать градиентные бустинги, решающие деревья и прочие "традиционные"модели, которые показывали куда более впечатляющие результаты, чем нейронные сети.

Подход с использованием нейросетей является относительно новым, и не отличался популярностью до появления модели TabNet, которая показала внушительные результаты, вызвав интерес у сообщества. После выхода статьи многие исследователи пытались опровергнуть или подтвердить эффективность этой модели, сравнивая ее по различным параметрам с другими на разных датасетах.

В данной работе исследуется подход к обучению с использованием аугментаций для табличных данных, созданных с помощью генеративно-состязательных сетей (GAN), а также self-supervised подход к обучению с целью найти способы улучшить результаты TabNet.

Данная работа состоит из 66 страниц, 3 глав, 41 рисунка, 6 таблиц. Использовано 58 источников.

Ключевые слова: Глубинное обучение; TabNet; табличные данные; обучение с учителем; unsupervised pretraining; самооообучение; генеративно-состязательные сети; генерация табличных данных.

Abstract

In recent years, machine learning has been actively used in many areas for various purposes, and often the input data is tabular data. For tabular data, it is common in the industry to use gradient boosting, decision trees and other "traditional"models, which showed much more impressive results than neural networks.

The neural network approach is relatively new, and did not gain popularity until the advent of the TabNet model, which showed impressive results, generating interest from the community. After the publication of the TabNet article, many researchers tried to refute or confirm the effectiveness of this model by comparing it in various parameters with others on different datasets.

In this work was explored an augmentation learning approach for tabular data generated with Generative Adversarial Networks (GANs), as well as a self-supervised learning approach to find ways to improve TabNet results.

The paper contains 66 pages, 3 chapters, 41 figures, 6 tables. 58 sources are used.

Keywords: Deep Learning; TabNet; tabular data; supervised learning; unsupervised pretraining; self-supervised learning; generative adversarial networks; tabular data generation.

Содержание

Используемые определения и термины

Аугментации (Augmentations) – увеличение выборки данных для обучения модели через модификацию существующих данных.

Батч (Batch) – пакет, малая выборка, на которые разбиваются данные.

Батч-нормализация (Batch Normalization, BN) – приведение распределения к стандартному нормальному в пакете.

Генеративно-состязательные сети (GAN) – нейронные сети, используемые для обучения без учителя, архитектура которых состоит из двух частей: генератора и дискриминатора; генератор генерирует объекты, а дискриминатор пытается отличить настоящие объекты от сгенерированных.

Глубинное обучение – обучение нейронных сетей.

Нейронные сети((Deep) Neural Networks, DNN) – модель машинного обучения, по структуре основанная на принципах взаимодействия нейронов в человеческом мозге.

Самообучение (Self-supervised learning) – обучение особого вида. Состоит из двух частей - unsupervised pretraining (предобучение без учителя) и supervised fine-tuning(дообучение с учителем).

Табличные данные – данные, организованные в таблицы, где каждая колонка представляет признак.

Введение

В последние годы машинное обучение активно используется во всех сферах для различных целей, начиная от медицины и заканчивая рекомендательными системами. В машинном обучении различают категории обучения: supervised (обучение с учителем), unsupervised(обучение без учителя), а также reinforcement learning (обучение с подкреплением), а также некоторый спектр задач (регрессия, классификация, кластеризация и прочие). Значительно различаются и данные, на которых обучаются модели - это могут быть изображения, звук, текст, но зачастую на вход подаются табличные данные. Как утверждают авторы статьи [1], судя по опросам 2021 года, табличные данные являются самой популярной формой представления данных, поэтому проблемы обучения на табличных данных все еще актуальны.

Нейронные сети уже давно применяются в разных задачах, хорошо справляясь с сегментацией, детекцией изображений [2], работой с временными рядами [3], аудио [4], но на табличных данных по-прежнему доминировали традиционные методы машинного обучения (например, ансамбли решающих деревьев), поэтому можно увидеть статьи, подобные [5], в которой автор утверждает, что нейросети никогда не заменят классические методы машинного обучения по многим причинам.

Так было до появления TabNet [6]. С появлением TabNet интерес к глубинному обучению на табличных данных значительно вырос [7], поскольку авторам удалось создать модель, которая использовала преимущества нейронных сетей, и в то же время сохранила простоту применения (end-to-end) и интерпретируемость. Некоторые авторы пытались доказать, что TabNet по-прежнему не превосходит традиционные модели ([9; 8]), кто-то адаптировал TabNet под свои потребности [10].

Наиболее полное исследование области моделей для табличных данных сделали авторы статьи [11], выводы и заключения из которой послужили основой постановки задач для данной работы.

Целью этой работы было провести исследования для улучшения результатов TabNet при применении для регрессии и классификации, для чего были поставлены следующие задачи:

1) Проверить, как повлияют на качество обучения аугментации табличных данных

2) Проверить, как работает self-supervised подход (unsupervised pretraining и затем supervised fine-tuning)

В Главе 1 приведен краткий обзор предыдущих работ по схожей тематике и смежным областям. В Главе 2 описаны схемы экспериментов и наборов данных. В Главе 3 расположены результаты проведенных экспериментов и выводы. В заключении приведены краткие выводы по проделанной работе.

Для возможности воспроизведения экспериментов код (а также дополнительные материалы) опубликован по ссылке: https://github.com/Zhekuson/TabnetResearch.

Глава 1. Обзор источников

В этом разделе приведён краткий обзор предыдущих работ по нескольким темам:

1) Обучение на табличных данных
   - Обучение на табличных данных "традиционными"методами
   - Модели глубокого обучения до TabNet
   - Краткий обзор статьи TabNet
   - Прочие модели
   - Анализ текущего состояния области
2) Аугментации табличных данных
   - Тривиальные методы
   - Генерация данных с помощью автокодировщиков
   - Генерация синтетических данных с помощью генеративно-состязательных сетей (GAN)

1.1. Обучение на табличных данных

Список рассмотренных моделей для табличных данных приведен в таблице 1

1.1.1. Обучение на табличных данных традиционными методами

Говоря о традиционных методах, первыми стоит упомянуть самые простые, а именно линейную [12] и логистическую [13] регрессии, изобретенные более сотни лет назад. Несмотря на свою простоту, при правильно применении они могут показывать хорошие результаты [14].

Началом новой эры в машинном обучении можно считать создание модели решающего дерева [15]. Особенность данной модели состоит в построении дерева предикатов, по которому определяется ответ на поступающие на вход данные, причем сама модель является очень сильной и легко переобучается, подстраиваясь под выборку.

Главный недостаток решающего дерева исправляет принцип, лежащий в основе более современных методов - принцип ансамблирования моделей.

1.1.1.1. Случайный лес, бэггинг, градиентный бустинг

Случайный лес является самым простым, но при этом весьма эффективным способом ансамблирования моделей. Каждое дерево обучается на случайном подмножестве признаков, а при предсказании модели голосуют.

Бэггинг [16] принципиально похож на лес: каждая модель независимо от других обучается на разных подмножествах обучающей выборки (в отличие от леса).

Градиентный бустинг [17; 18] использует другой метод ансамблирования - модели обучаются последовательно, корректируя ошибки предыдущих. Этот подход позволяет добиться более высокого качества, но повышает риски переобучения

1.1.1.2. CatBoost, LightGBM, XGBoost

В настоящий момент среди бустингов в индустрии преобладают три фреймворка, сравнение с которыми провели в оригинальной статье TabNet [6]:

1) XGBoost [19] (eXtreme Gradient Boosting) - модель градиентного бустинга, ставшая известной после победы в соревновании [20]. особенностями является использование метода Ньютона и распараллеливание процессов

2) CatBoost [21] (Categorical Boosting) - градиентный бустинг, особенностями которого являются работа с категориальными признаками и использование oblivious decision tree (ODT)[22] в качестве базовых моделей

3) LightGBM [23] (Light Gradient Boosting Machine) - особенностью данной модели является leaf-wise (1) подход к построению решающих деревьев в отличие от обычного level-wise (2)

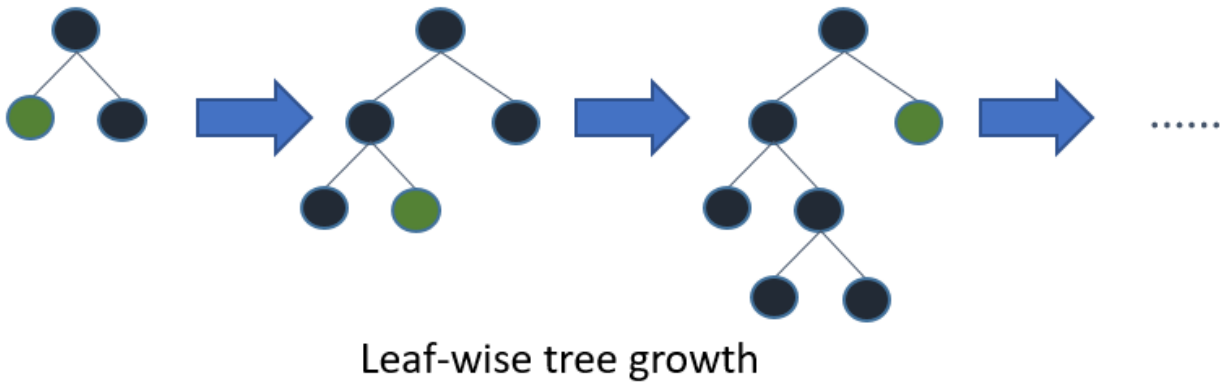


Рис. 1 — Leaf-wise рост [24]

Во многих статьях проводится сравнение этих трех моделей, например [26; 27; 25], и по-прежнему все три составляют достойную конкуренцию друг друг, незначительно опережая в конкретных случаях применений.

1.1.2. Модели глубокого обучения до TabNet

До появления TabNet существовали модели нейросетей, которые использовали различные методы для улучшения качества. Значимым является фреймворк DeepTables [28], собравший воедино многие модели для обучения на табличных данных. Приведем некоторые из них:

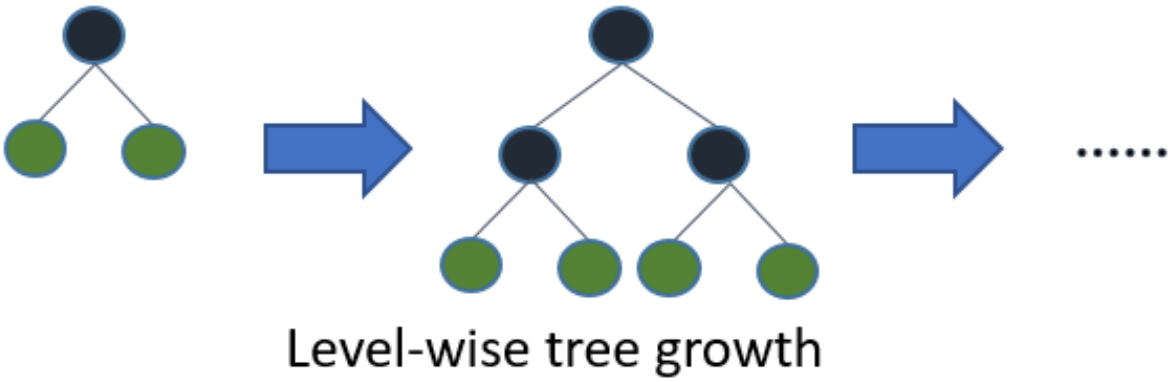


Рис. 2 — Level-wise[24]

- Wide&Deep [29]

  Данная модель использовала подход, совместивший в себе линейные модели (блок "wide") и нейронные сети (блок "deep"), обучаюмые совместно. Созданная авторами модель была использована в рекомендательной системе Google Play, показав статистически значимый прирост скачиваний приложений

- Deep&Cross Network (DCN) [30]

  Авторы этой модели пытались решить проблему feature engineering, возникшую

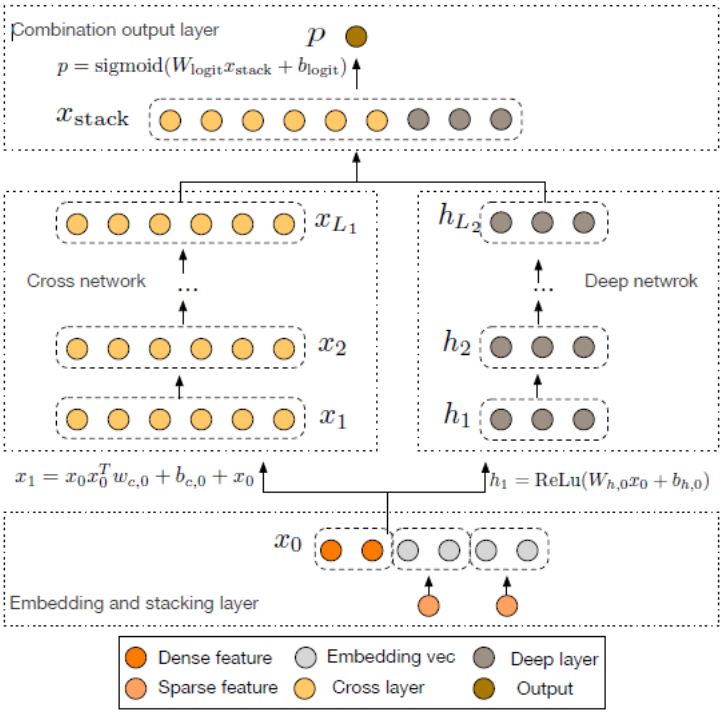


Figure 1: The Deep & Cross Network

Рис. 3 — Структура Deep&Cross [30]

из-за роста размерности в датасетах. Ключевая идея этой модели состоит в использовании пересечения признаков (feature crossing), которое усложняет модель, но позволяет избавиться от ручного создания признаков и поиска закономерностей.

- Attentional factorization machines (AFM) [31]
  Авторы этой модели решили применить механизм внимания (attention), чтобы

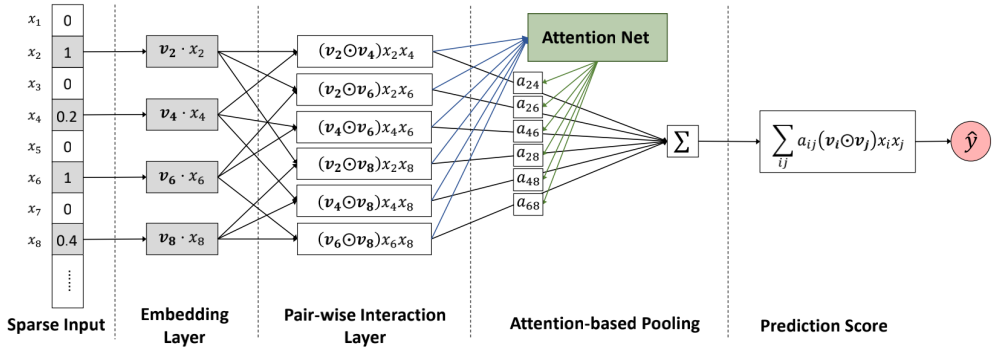


Figure 1: The neural network architecture of our proposed Attentional Factorization Machine model.

Рис. 4 — Структура AFM [31]

использовать важности попарного взаимодействия признаков. В статье показано применение AFM на двух датасетах, где она превзошла D&C и D&W

- AutoInt [32]
  Авторы использовали механизм самовнимания (self-attentive neural network) и остаточные связи (residual connections), для проверки своих гипотез взяли четыре больших (миллионы объектов) датасета, на которых эта модель показала себя лучше, чем остальные.

1.1.3. Краткий обзор статьи TabNet

TabNet[6] аккумулировала в себе опыт прошлых исследований, авторы решили взять лучшие по их мнению подходы.

- **Решающие деревья.** Решающие деревья строят решающие гиперплоскости (Рисунок), авторы решили использовать тот же метод, но используя обычные блоки DNN, а затем имитировать ансамблирование.

- **Обработка признаков и механизм внимания.** Двумя базовыми составляющими работы с признаками являются *feature transformer* и *attentive transformer* (рисунок 6). Первый принимает на вход признаки и состоит из FC(полносвязный) слой, BN (батч-нормализацию) and GLU(Gated Linear Unit)[33], второй принимает отделенные в ходе отбора признаки и пропускает их через Sparsemax[34].

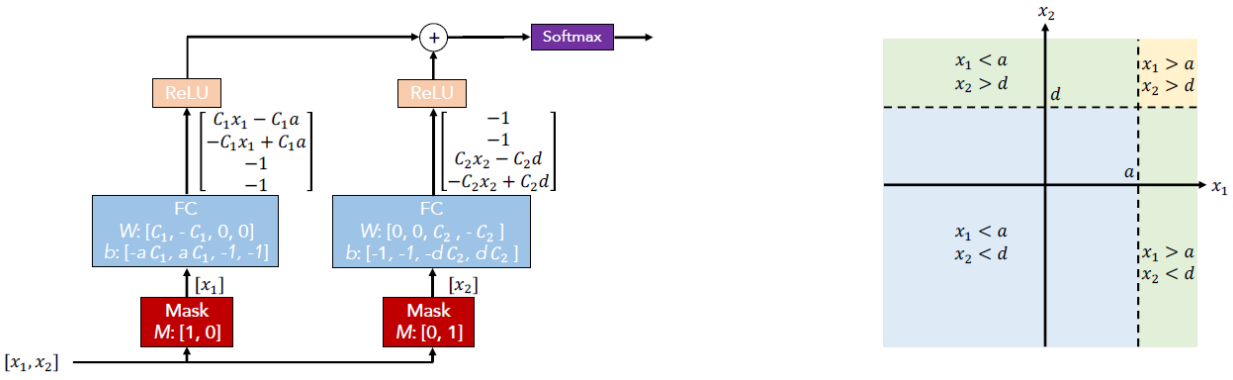


Рис. 5 — Особенности решащих деревьев [6]

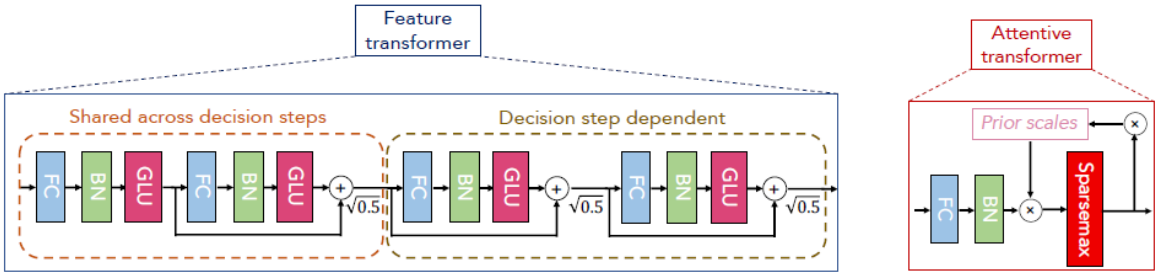
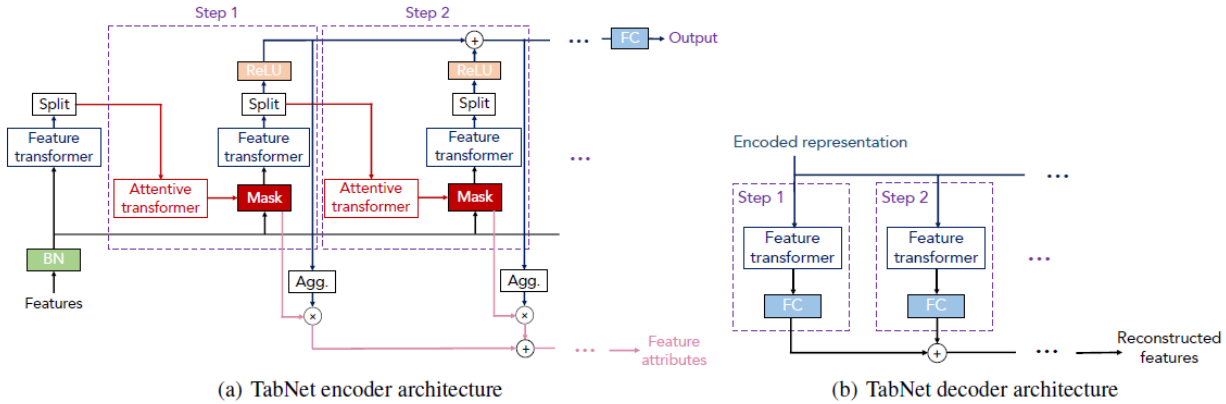


Рис. 6 — Feature transformer и attentive transformer [6]

- **Кодировщик и декодировщик.** Структуры более высокого уровня. Кодировщик используется для перехода из пространства признаков в пространство эмбеддингов, декодировщик - для обратного преобразования.



(a) TabNet encoder architecture (b) TabNet decoder architecture

Рис. 7 — Кодировщик и декодировщик [6]

- **Отбор признаков и интерпретируемость.** Для отбора признаков используется обучаемая маска (рисунок 8) $M[i] \in \Re^{B \times D}$ (B - размер батча, D - размерность датасета), которая проецирует из Евклидова пространства в вероятностное. Эта маска, как показали авторы, может быть использована для интерпретации важности признаков.

- **Self-supervised learning.** Декодировщик, описанный ранее, может быть использован для декодирования обратно в исходные признаки, и тем самым восстанав-

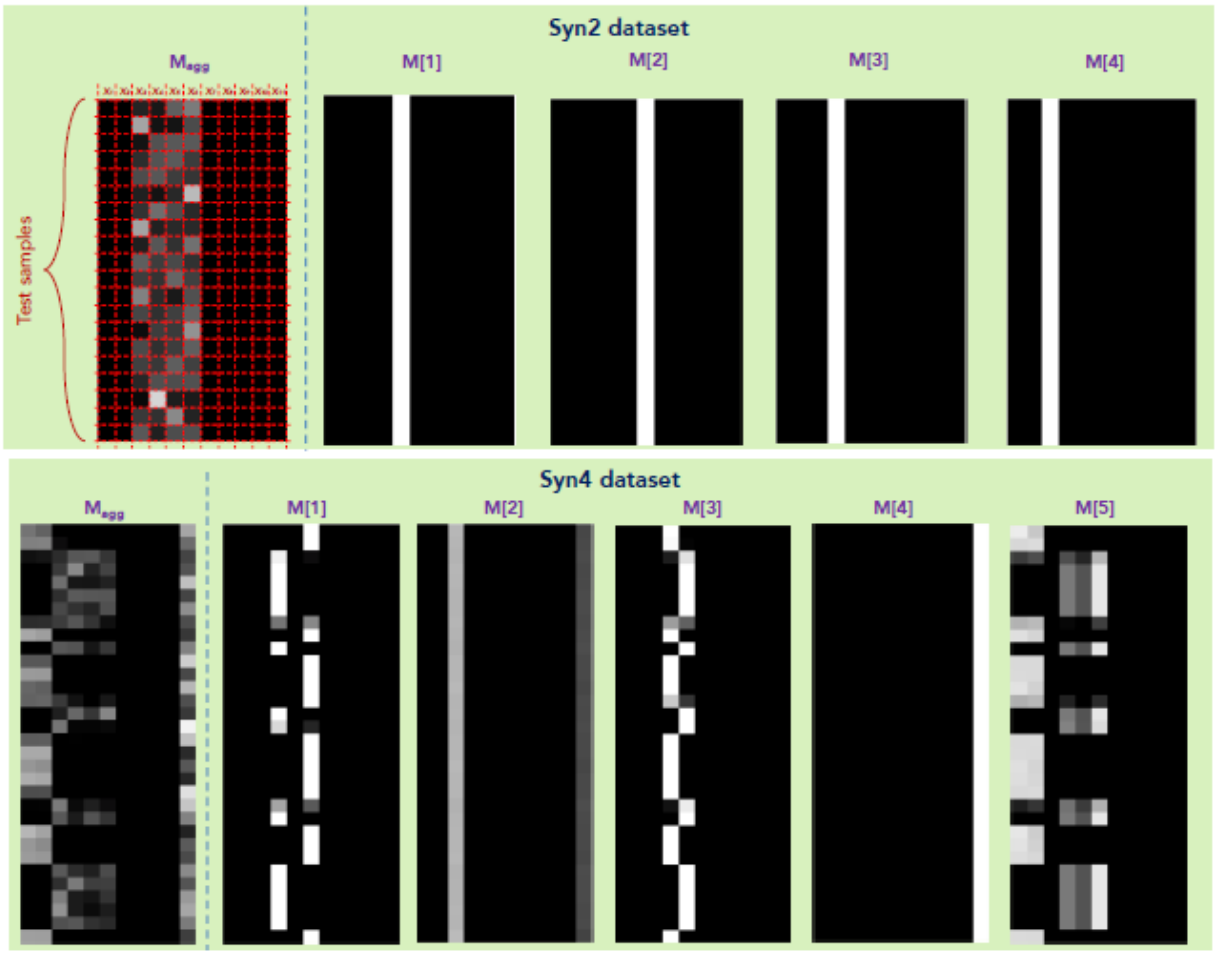


Рис. 8 — Пример визуализации масок из статьи[6]

ливать пропуски в данных и подстраиваться под данные.

1.1.4. Прочие модели

Помимо упомянутых выше, существуют модели, которые выходили до и после TabNet, но не вошли в предыдущие категории. В их основе лежат разные принципы и идеи с целью превзойти эффективные аналоги в виде фреймворков бустингов, но особой популярностью они не отличаются. Приведем наиболее значимые из них:

- Regularization Learning Networks (RLNs) [35] - модель, создатели которой решали проблему большого количества гиперпараметров путем сведения ее к подбору регуляризационных коэффициентов. Авторы показали значительный выигрыш по сравнению с нейросетями, но не смогли значительно обогнать градиентные бустинги (конкретно в их случае - XGBoost).

- Neural Decision Trees (NDTs) [36] - в этой модели были в узлы решающего дерева были встроены нейросети. В этой же статье авторы создали Hashing Neural Network (HNN), улучшенную версии NDT, в которой последний слой играл роль хеш-функции. На датасетах из реальной жизни авторы модель не тестировали, но

на готовых искусственных они показывали многообещающие результаты.

- Deep Neural Decision Trees (DNDTs) [37] - модель, подход авторов которой отличается от NDT. В основу легли все те же решающие деревья, но идея состояла в ансамблировании сетей, сделанных под каждый признак. Несмотря на успехи в конкретных случаях, при общем сравнении на 14 датасетах DNDT потерпели поражение по количеству побед.

- Disjunctive normal form networks (Net-DNF) [9] - модель с довольно выделяющимся подходом на фоне остальных - строго определенные 4 слоя: входной, выходной, а между ними два слоя, имитирующие дизъюнктивную нормальную форму. На некоторых датасетах Net-DNF смогла победить XGBoost, но в целом не показала впечатляющих результатов.

- SubTab [38] - сравнительно новая работа, которая использует технику self-supervised learning и разбиения признаков на подмножества. Показала многообещающие результаты, победив в сравнении с XGBoost на всех предложенных авторами датасетах.

Таблица 1 — Модели для табличных данных

Название модели	**Подход (тип)**	**Особенности, лучшие применения**
Линейная регрессия	Регрессионная модель	Интерпретация коэффициентов в виде весов, линейность
Логистическая регрессия	Регрессионная модель	Использование сигмоиды. Высокие показатели в случае низкоразмерных датасетов с дисбалансом классов (заболевания) [14], превзошедшие градиентный бустинг и случайный лес
Решающее дерево	Решающее дерево	Использование предикатов в узлах, минимизация энтропии. Базовая модель для построения более сложных.
Бэггинг	Ансамбль (решающих деревьев)	Использование подвыборок (bootstrap aggregating). Для хорошего качества требуется, чтобы базовые модели независимо друг от друга давали хороший результат и были неустойчивыми, иначе "плохие модели станут еще хуже"[17].
Случайный лес	Ансамбль (решающих деревьев)	Обучение на подпространствах признаков. Успешно применяется во многих областях, например, при оценке качества статей Википедии (популярной онлайн-энциклопедии) [39]

Таблица 1 — Модели для табличных данных

Название модели	**Подход (тип)**	**Особенности, лучшие применения**
Градиентный бустинг	Ансамбль (решающих деревьев)	Последовательное обучение моделей для минимизации ошибки. На данный момент разновидности бустинга успешно побеждают во многих соревнованиях на kaggle
Catboost	Градиентный бустинг на деревьях	Использование Oblivious decision trees. Одна из трех сильнейших модификаций градиентного бустинга
LightGBM	Градиентный бустинг на деревьях	Leaf-wise построение деревьев. Одна из трех сильнейших модификаций градиентного бустинга
XGBoost	Градиентный бустинг на деревьях	Использование метода Ньютона. Одна из трех сильнейших модификаций градиентного бустинга
Wide&Deep	Нейросеть	Блоки линейных моделей вместе с полносвязными слоями.
DCN	Нейросеть	Feature crossing. В статье показано преобладание над логистической регрессией на датасете Criteo Display Ads.
AFM	Нейросеть	Механизм внимания. Превзошла DCN и W&D на датасетах Frappe и MovieLens как по качеству (RMSE), так и по количеству настраиваемых параметров
AutoInt	Нейросеть	Residual connections и механизм внимания. На датасетах Criteo, Avazu, KDD12 и MovieLens превзошла другие модели (W&D, D&C, AFM)
RLNs	Нейросеть	Упор на регуляризацию. Не смогла обойти градиентные бустинги
NDTs	Решающие деревья с нейросетями	Встроенные нейросети в узлы решающего дерева. Сравнения с другими моделями не проводились
DNDTs	Решающие деревья с нейросетями	Ансамблирование нейросетей для каждого признака. В статье сравнивали качество на 14 датасетах с нейронными сетями и обычными решающими деревьями, в среднем DNDTs превзойти конкурентов не смогли

Таблица 1 — Модели для табличных данных

Название модели	**Подход (тип)**	**Особенности, лучшие применения**
Net-DNF	Нейросеть	Четырехслойная сеть с имитацией ДНФ. Смогла превзойти XGBoost только на одном из 6 датасетов в статье
SubTab	Нейросеть	Разбиение на подпространства, пересечение эмбеддингов, self-supervised learning.
TabNet	Нейросеть	Имитация решающих деревьев, механизм внимания, эмбеддинги, отбор признаков, self-supervised learning. В статье превзошла все градиентные бустинги практически на всех сравнениях (кроме датасета KDD)

1.1.5. Анализ текущего состояния области

Существуют несколько обзорных статей по текущему состоянию исследований в сфере табличных данных. В первую очередь, это статья [8], авторы которой утверждают, что градиентный бустинг все еще не превзойден, а также поднимают другие вопросы, например, насколько вычислительно сложна подгонка гиперпараметров. Эксперименты показывают, что XGBoost в этом отношении значительно выигрывает у моделй глубокого обучения.

Также стоит упоминания статья [40], авторы которой сравнивают CatBoost с другими моделями и делают важный вывод: исследования моделей глубокого обучения, направленные на превосхождение результатов фреймворков градиентного бустинга, должны обязательно содержать эксперименты на различных разнородных датасетах, помимо обычных однородных, которые должны служить лишь проверкой.

Наиболее полным исследованием на данный момент является статья [11], в которой выделены несколько проблемных моментов:

- **Ансамбли решающих деревьев все еще лучше**. XGBoost, LightGBM, и CatBoost, в среднем по-прежнему побеждают модели глубинного обучения в сравнениях
- **Единый бенчмаркинг**. Для того, чтобы исследования двигались дальше, ученым нужен единый общепринятый бенчмаркинг моделей, а его на данный момент нет.
- **Специальные архитектуры нейронных сетей**. Последние годы показали, что успеха достигают модели с архитектурой, специально приспособленной к табличным данным.
- **Transfer Learning**. В других сферах машинного обучения с применением глубинных моделей практика transfer learning общепринята, в то время как в сфере

табличных данных нет подобных применений.

- **Аугментации для табличных данных**. В компьютерном зрении аугментации показали свою высокую эффективность [41], расширяя обучающую выборку. Применение простых аугментаций (SMOTE-NC)[42] не дает улучшений в качестве, но потенциально это область для исследований.

- **Self-supervised подход**. На больших датасетах потенциально это может дать значительный прирост в качестве.

Последние два пункта легли в основу данного исследования.

1.2. Аугментации табличных данных

В этом разделе будут приведены краткие описания способов аугментации табличных данных. Список рассмотренных вариантов приведен в таблице 2

На данный момент наиболее полным и современным исследованием в области аугментаций табличных данных является статья [43]. В ней рассмотрена подавляющая часть существующих методов аугментаций.

1.2.1. Тривиальные методы

Самый простой способ создать аугментации для табличных данных - воспользоваться техникой oversampling (SMOTE, SMOTE-NC)[42]. Еще один способ - изменения с помощью синтетических методов интерполяции, например, Poly [44]. Однако, данные методы редко работают хорошо, поскольку простые модели не отражают структуру зависимостей в данных [11].

1.2.2. Генерация с помощью автокодировщиков

Автокодировщик учится генерировать данные, похожие на настоящие. Двумя составляющими являются энкодер и декодер. Положительной стороной данного подхода является обучение под конкретные данные, что позволяет создавать похожие на настоящие образцы.

1.2.3. Генерация синтетических данных с помощью генеративно-состязательных сетей (GAN)

Наиболее многообещающий подход по мнению авторов [11]. Главная идея данного подхода состоит в обучении сетей генератора и дискриминатора так, чтобы генератор создавал синтетические данные, а дискриминатор учился отличать их от настоящих. Среди важных моделей: TGAN [45], последовавшая за ней CTGAN[46], а также вероятностная CopulaGAN, реализованная в фреймворке SDV [47]

Таблица 2 — Методы аугментации табличных данных

Название метода	**Тип**	**Особенности**
SMOTE (SMOTE-NC)	Over-sampling	Простейшие аугментации, увеличивают число объектов путем похожих из исходных
Poly	Интерполяция	Метод создания синтетических объектов с помощью многочленов (Curve Fitting Methods to find the coefficients of a polynomial p(x))
TVAE	Автокодировщик	Tabular VAE, вариационный автокодировщик, создает закодированные представления объектов и декодирует их обратно в синтетические
TGAN	GAN	Tabular GAN,
CTGAN	GAN	Conditional TGAN, усиленный вектором условий (condition vector) на признаки
CopulaGAN	GAN	Вероятностная модель, использующая гауссовские копулы

Выводы по главе

В данной главе был проведен краткий обзор последних работ по смежным тематикам. Были показаны основные направления исследований и идеи для дальнейшей работы, а также мотивация проведения данного исследования.

Глава 2. Описание экспериментов

В данной главе будут описаны схемы экспериментов, порядок их проведения и обоснование выбора технологий и наборов данных

Для данного исследования были поставлены две основные цели: исследовать изменение качества при применении аугментаций для табличных данных и self-supervised подход при обучении TabNet. Были выделены следующие задачи:

1) Выбрать инструментарий для проведения работы
2) Выбрать модели и методику для сравнения
3) Выбрать способы табличных аугментаций для исследования
4) Обучить аугментирующие модели
5) Провести эксперименты с аугментирующими моделями
6) Провести эксперименты с self-supervised предобучением
7) Собрать результаты экспериментов и сделать выводы

2.1. Выбор инструментов и методов

Поскольку основой для проведения экспериментов служит статья TabNet, к которой приложен репозиторий [48] от создателей, для простоты был выбран язык Python версии 3.9, а конкретно реализация TabNet на фреймворке Pytorch []. Для сбора данных был выбран фреймворк Wandb [49]. В качестве вспомогательных инструментов - Numpy[50], Pandas[51], Sklearn[52]

2.2. Модели и методика сравнения

2.2.1. Набор моделей для сравнения

Помимо проведения экспериментов над TabNet, в качестве конкурирующих моделей было принято решение выбрать три фреймворка градиентного бустинга - CatBoost, XGBoost и LightGBM. Для каждого датасета набор гиперпараметров у каждой модели зафиксирован и не меняется независимо от пропорций аугментаций

2.2.2. Выбор датасетов

Для данного исследования были подобраны следующие датасеты:

- Синтетические датасеты: **Syn1**, **Syn2**, **Syn3**, **Syn4**, **Syn5**, **Syn6**
  Датасеты из статьи [53], использованные в статье TabNet. Цель - бинарная классификация, датасеты из 11 признаков генерируются по определенным правилам.

- **Sarcos** [54]
  Из исходной статьи TabNet
- **Pokerhand**[55]
  Из исходной статьи TabNet
- **Forest cover type** [55]
  Из исходной статьи TabNet
- **Airfoil** [55]
  Датасет от NASA, регрессия шума аэродинамического профиля
- **Stars** [56]
  Датасет от NASA, классификация звезд

Сводный список датасетов расположен в таблице 3

Таблица 3 — Список использованных датасетов

Название	**Тип**	**Метрика качества**	**Размеры**
Syn1-6	Бинарная классификация	AUC	11 признаков, 10000 объектов
Forest cover	Многоклассовая классификация	Accuracy	55 признаков, 551k объектов
Stars	Многоклассовая классификация	Accuracy	5 признаков, 241 объект
Pokerhand	Многоклассовая классификация	Accuracy	9 признаков, 525k объектов
Sarcos	Регрессия	MSE	21 признак, 44k объектов
Airfoil	Регрессия	MSE	6 признаков, 1.5k объектов

2.2.3. Схемы экспериментов

Для обеспечения относительной правдоподобности результатов, было принято решение использовать k-fold кросс-валидацию, с числом $k = 5$. В каждом эксперименте датасет разбивается на 5 равных частей, модель обучается на 4 частях, а пятая часть служит для валидации. Результатом эксперимента служат среднее арифметическое и стандартное отклонение, то есть $[\overline{x}, \sigma]$,

$$\overline{x} = \frac{\sum_{i=1}^{k} metric[i]}{k}, \sigma = \sqrt{\frac{\sum_{i=1}^{k} (metric[i] - \overline{x})^2}{k}}$$

где $metric[i]$ - лучший результат (конкретной метрики) на валидационной части за все время обучения модели.

Общую схему эксперимента описывает алгоритм 1

Алгоритм 1 Алгоритм эксперимента

Initialize $dataset, model, augmentations, percentage, results$
$k \leftarrow 5$
$splits \leftarrow$ **split**$(dataset, k)$
for i = 0 to k **do**
  $X_{eval}, y_{eval} \leftarrow splits[i].X, splits[i].y$
  Initialize X_{train}, y_{train}
  for $j = 0$ to $k, j \neq i$ **do**
    Append $splits[j].X, splits[j].y$ to X_{train}, y_{train}
  for all $(augmentation, percentage)$ **do**
    Apply $augmentation$ to X_{train}, y_{train} in $percentage$
  Initialize $best\_metric$
  for iteration in model.iterations **do**
    train$(model, X_{train}, y_{train})$
    $current\_metric \leftarrow$ **evaluate**$(model, X_{eval}, y_{eval})$
    if $current\_metric$ is better than $best\_metric$ **then**
      $best\_metric \leftarrow current\_metric$
  Append $best\_metric$ to $results$
return $results$

2.3. Выбор аугментаций для сравнения

Поскольку существует множество различных способов для аугментаций, было принято решение взять по одному метод среди разных категорий. В итоге для простоты реализации были выбраны TVAE и CTGAN, реализованные в фреймворке SDV [47], а также одну из вариаций SMOTE, реализованную создателями TabNet в их репозитории [48]

Также остро стоит проблема измерения качества сгенерированных данных. Существуют различные способы оценить меру схожести данных, например, дивергенция Кульбака-Лейблера [57]. В итоге принято решение применять аггрегацию из 29 метрик библиотеки SDV.

2.4. Обучение аугментирующих моделей

В отличие от простейших аугментаций SMOTE, модели TVAE и CTGAN требуют обучения, причем сильно зависят от гиперпараметров. Для того чтобы подобрать оптимальные параметры для обучения, был использован фреймворк Optuna [58], с перебором по сетке гиперпараметров. Еще одной проблемой является обучение аугментирующих моделей на больших датасетах, требующие значительных вычислительных

ресурсов и времени, по этой причине было принято решение обучения на малых подвыборках больших датасетов, как и проведение бенчмаркинга.

2.4.1. Бенчмаркинг

Для проверки качества генерируемых синтетических данных была осуществлена проверка на случайной подвыборке датасета (в зависимости от размера) и в качестве результата взято среднее значение метрики качества от 10 запусков, описанной ранее.

2.5. Эксперименты с аугментирующими моделями

Каждый эксперимент включает в себя запуски с различными пропорциями аугментаций (пропорция от размера исходного датасета) с шагом 0.1 начиная от 0 до 0.5. Этого достаточно, чтобы проследить тенденцию и сделать верные выводы.

2.6. Self-supervised обучение для TabNet

Для проведения self-supervised обучения выполняется предобучение TabNetPretrainer с фиксированными гиперпараметрами для каждого датасета, затем происходит обучение по алгоритму 1 без аугментаций

Выводы по главе

В данной главе были рассмотрены методы и инструментарий для проведения экспериментов, а также обоснование их выбора

Глава 3. Результаты экспериментов

В этой главе будут приведены результаты экспериментов, а также некоторые выводы к ним.

3.1. Обучение аугментирующих моделей

Результаты обучения аугментирующих моделей для различных представлены в таблице 4. Набор подобранных гиперпараметров для обучения представлен в Приложении

Таблица 4 — Обучение аугментирующих моделей

Датасет	**Тип**	**Доля подвыборки для обучения**	**Доля подвыборки для бенч-маркинга**	**Результаты бенчмаркин-га**
Syn1	CTGAN	1	0.25	0.4044 ± 0.0028
	TVAE	1	0.25	0.3675 ± 0.0016
Syn2	CTGAN	1	0.25	0.4029 ± 0.0054
	TVAE	1	0.25	0.3776 ± 0.0029
Syn3	CTGAN	1	0.25	0.4053 ± 0.0036
	TVAE	1	0.25	0.3922 ± 0.0027
Syn4	CTGAN	1	0.25	0.394 ± 0.003
	TVAE	1	0.25	0.3598 ± 0.0023
Syn5	CTGAN	1	0.25	0.3865 ± 0.0032
	TVAE	1	0.25	0.399 ± 0.0032
Syn6	CTGAN	1	0.25	0.3738 ± 0.0034
	TVAE	1	0.25	0.4354 ± 0.0029
Forest cover	CTGAN	0.05	0.017	0.5821 ± 0.0043
	TVAE	0.05	0.017	0.5257 ± 0.0026
Sarcos	CTGAN	0.4	0.025	0.4983 ± 0.0048
	TVAE	0.4	0.025	0.6605 ± 0.0076
Airfoil	CTGAN	1	0.66	0.5369 ± 0.0325
	TVAE	1	0.66	0.1612 ± 0.0417
Stars	CTGAN	1	1	0.7639 ± 0.0234
	TVAE	1	1	0.7879 ± 0.0128
Pokerhand	CTGAN	0.75	0.04	0.7634 ± 0.0095
	TVAE	0.75	0.04	0.5688 ± 0.0106

3.2. Обучение с аугментациями

В таблице 5 приведены лучшие результаты для каждой модели для каждого датасета. Полные данные по обучению можно найти в репозитории.

3.3. Влияние пропорций аугментаций на качество

На следующих рисунках (рис. 9–41) представлено изменение качества при изменениях пропорций аугментаций. По горизонтальной оси на каждом малом графике располагается пропорция выбранной аугментации, по вертикальной - качество на выбранном датасете. Координаты малого графика на большой сетке определяют зафиксированные пропорции других двух аугментаций (подписаны снизу и справа).

- **Stars**: см. рис. 9–11
- **Forest cover**: см. рис. 12–14
- **Airfoil**: см. рис. 15–17
- **Syn6**: см. рис. 18–20
- **Syn5**: см. рис. 21–23
- **Syn4**: см. рис. 24–26
- **Syn3**: см. рис. 27–29
- **Syn2**: см. рис. 30–32
- **Syn**: см. рис. 33–35
- **Sarcos**: см. рис. 36–38
- **Pokerhand**: см. рис. 39–41

Таблица 5 — Таблица лучших результатов

Датасет, метрика	**XGBoost**	**LightGBM**	**CatBoost**	**TabNet**
Forest cover, Accuracy	0.8693 ± 0.0009, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8547 ± 0.0023, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8902 ± 0.0009, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.9093 ± 0.0037, SMOTE:0.0, CTGAN:0.2, TVAE:0.1

Таблица 5 — Таблица лучших результатов

Датасет, метрика	**XGBoost**	**LightGBM**	**CatBoost**	**TabNet**
Stars, Accuracy	1.0 ± 0.0, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	1.0 ± 0.0, SMOTE:0.0, CTGAN:0.1, TVAE:0.3,	1.0 ± 0.0, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.9917 ± 0.0102, SMOTE:0.0, CTGAN:0.1, TVAE:0.3
Airfoil, MSE	2.3563 ± 0.2099, SMOTE:0.0, CTGAN:0.1, TVAE:0.1,	3.2746 ± 0.2587, SMOTE:0.0, CTGAN:0.5, TVAE:0.1,	2.1205 ± 0.3441, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	8.4284 ± 2.0321, SMOTE:0.0, CTGAN:0.5, TVAE:0.2
Syn6, AUC	0.8813 ± 0.005, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8883 ± 0.0057, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8931 ± 0.0069, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8271 ± 0.0179, SMOTE:0.0, CTGAN:0.0, TVAE:0.0
Syn5, AUC	0.7935 ± 0.0077, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.7999 ± 0.0061, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8133 ± 0.0083, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.7444 ± 0.0278, SMOTE:0.0, CTGAN:0.0, TVAE:0.05
Syn4, AUC	0.7774 ± 0.0098, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.7928 ± 0.0088, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8093 ± 0.0121, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.6999 ± 0.0204, SMOTE:0.0, CTGAN:0.1, TVAE:0.0
Syn3, AUC	0.8917 ± 0.0082, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.9015 ± 0.0042, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.9046 ± 0.0042, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8904 ± 0.0058, SMOTE:0.0, CTGAN:0.2, TVAE:0.0
Syn2, AUC	0.8917 ± 0.0082, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8933 ± 0.0071, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8981 ± 0.0066, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.8682 ± 0.0126, SMOTE:0.0, CTGAN:0.1, TVAE:0.0
Syn1, AUC	0.6724 ± 0.0121, SMOTE:0.0, CTGAN:0.1, TVAE:0.0,	0.6791 ± 0.0083, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.7011 ± 0.0117, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.6932 ± 0.0165, SMOTE:0.0, CTGAN:0.1, TVAE:0.2

Таблица 5 — Таблица лучших результатов

Датасет, метрика	**XGBoost**	**LightGBM**	**CatBoost**	**TabNet**
Sarcos, MSE	10.9685 ± 0.1648, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	13.2374 ± 0.2658, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	7.7632 ± 0.162, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	12.6099 ± 0.2799, SMOTE:0.0, CTGAN:0.0, TVAE:0.0
Pokerhand, Accuracy	0.7353 ± 0.0056, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.6247 ± 0.0116, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.748 ± 0.0045, SMOTE:0.0, CTGAN:0.0, TVAE:0.0,	0.5442 ± 0.0025, SMOTE:0.0, CTGAN:0.3, TVAE:0.1

3.4. Self-supervised обучение TabNet

В таблице 6 приведено сравнение метрик на всех датасетах по сравнению с предобученной моделью

Таблица 6 — Таблица сравнения предобученной TabNet

Датасет, метрика	**TabNet, предобученная**	**TabNet, без предобучения, лучший результат с аугментациями**	**TabNet, без предобучения, без аугментаций**
Forest cover, Accuracy	0.8733 ± 0.0047	0.9093 ± 0.0037	0.8889 ± 0.0101
Pokerhand, Accuracy	0.5545 ± 0.0071	0.5442 ± 0.0025	0.544 ± 0.0079
Sarcos, MSE	11.6001 ± 0.841	12.6099 ± 0.2799	12.6099 ± 0.2799
Syn1, AUC	0.6753 ± 0.0126	0.6932 ± 0.0165	0.6916 ± 0.0177
Syn2, AUC	0.882 ± 0.005	0.8682 ± 0.0126	0.8487 ± 0.0129
Syn3, AUC	0.9004 ± 0.0057	0.8904 ± 0.0058	0.8869 ± 0.0046
Syn4, AUC	0.7595 ± 0.0167	0.6999 ± 0.0204	0.6798 ± 0.0174
Syn5, AUC	0.7716 ± 0.0137	0.7444 ± 0.0278	0.7357 ± 0.0303
Syn6, AUC	0.8676 ± 0.0137	0.8271 ± 0.0179	0.8271 ± 0.0179
Airfoil, MSE	23.0156 ± 9.2761	8.4284 ± 2.0321	21.7588 ± 11.2842
Stars, Accuracy	0.9083 ± 0.1009	0.9917 ± 0.0102	0.8708 ± 0.0726

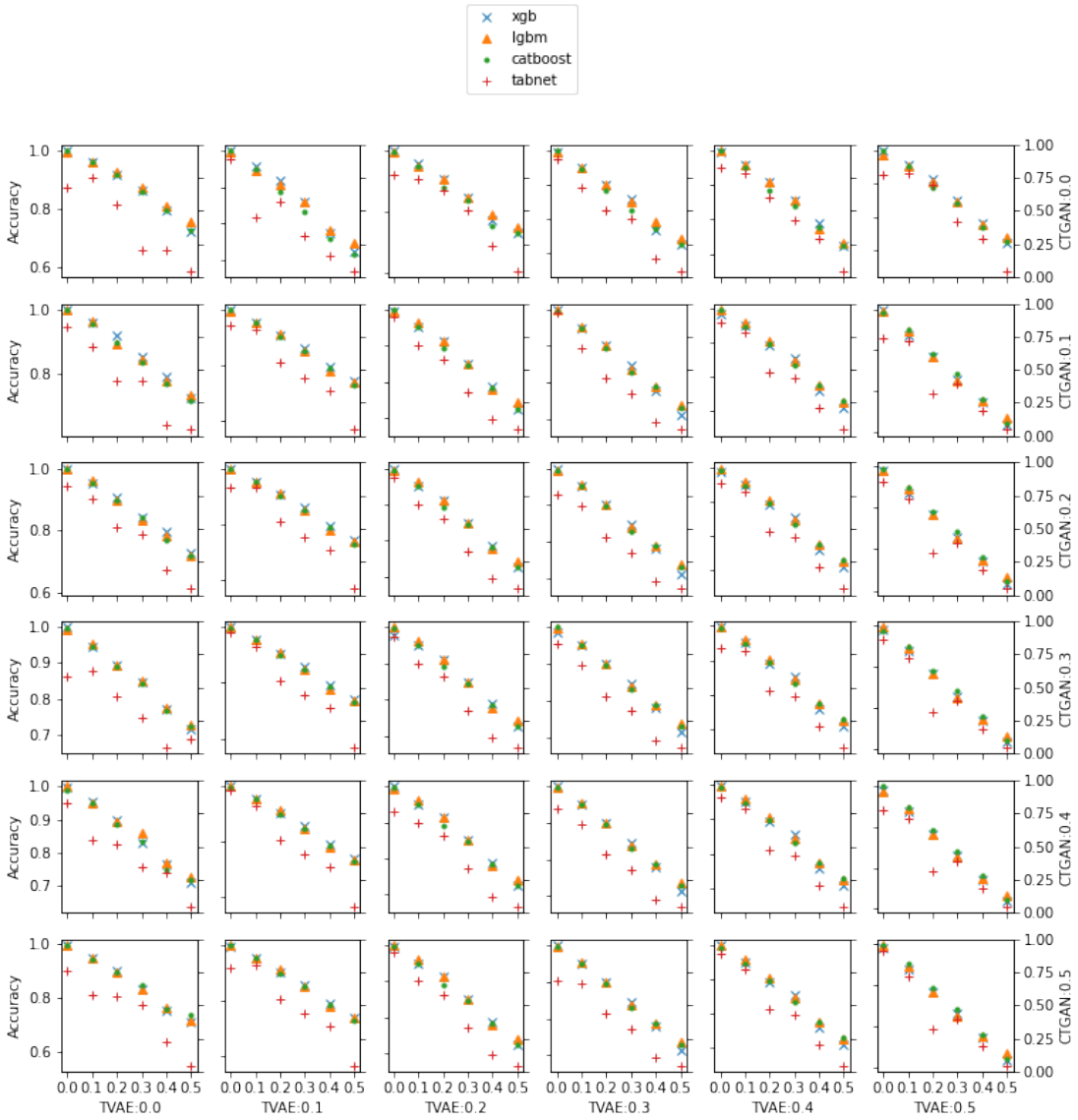


Рис. 9 — Изменения при меняющемся SMOTE, датасет Stars

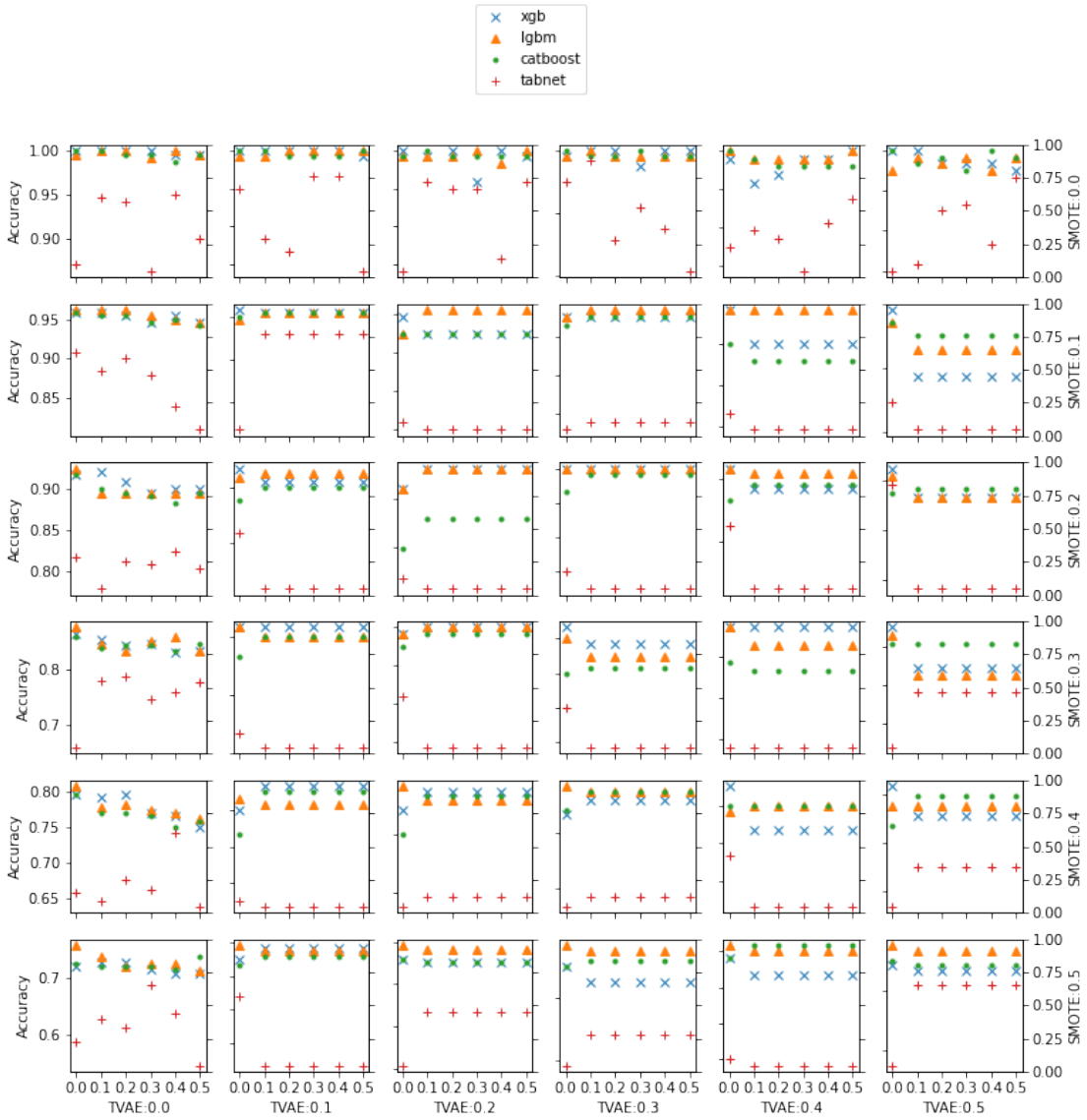


Рис. 10 — Изменения при меняющемся CTGAN, датасет Stars

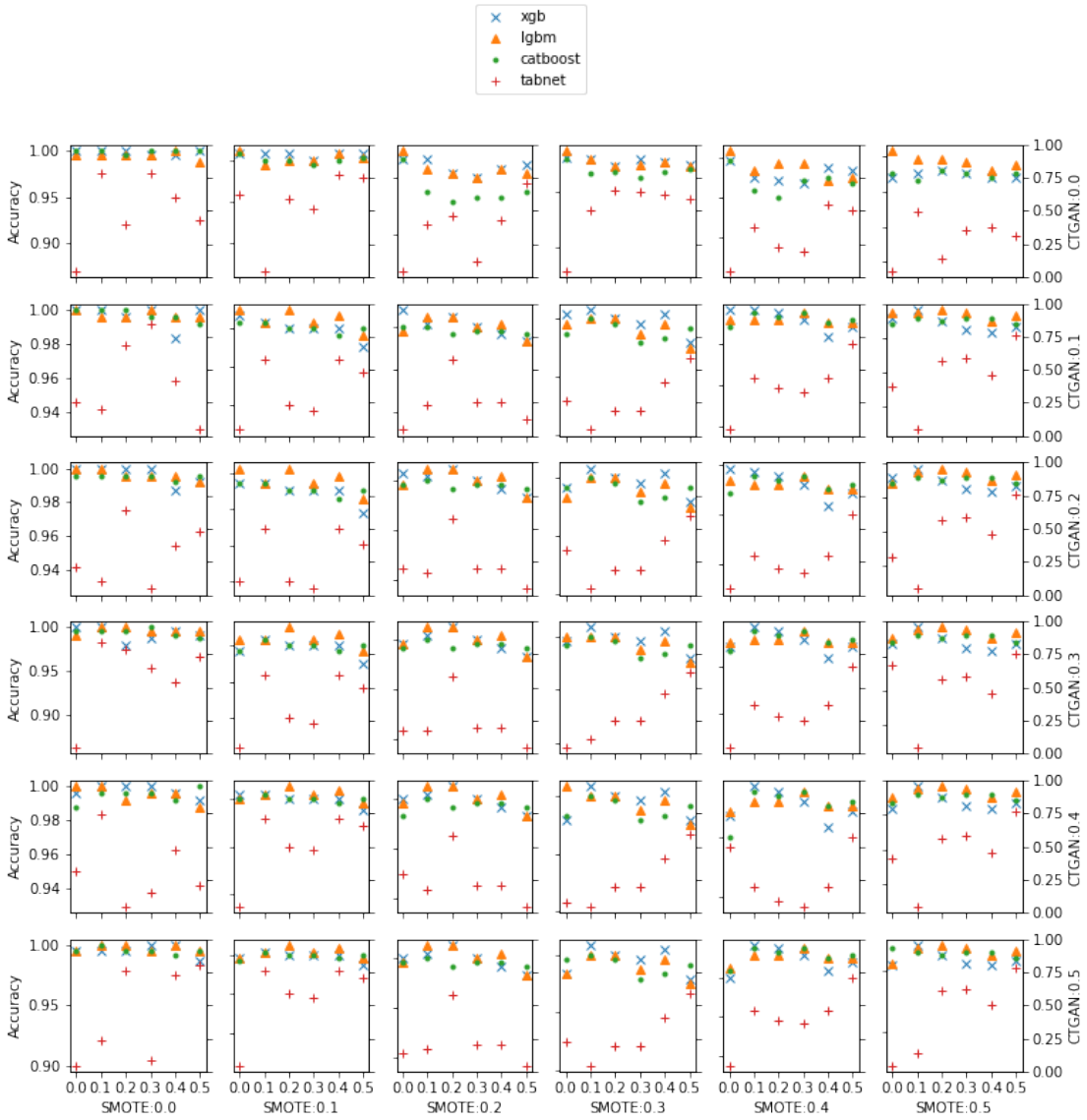


Рис. 11 — Изменения при меняющемся TVAE, датасет Stars

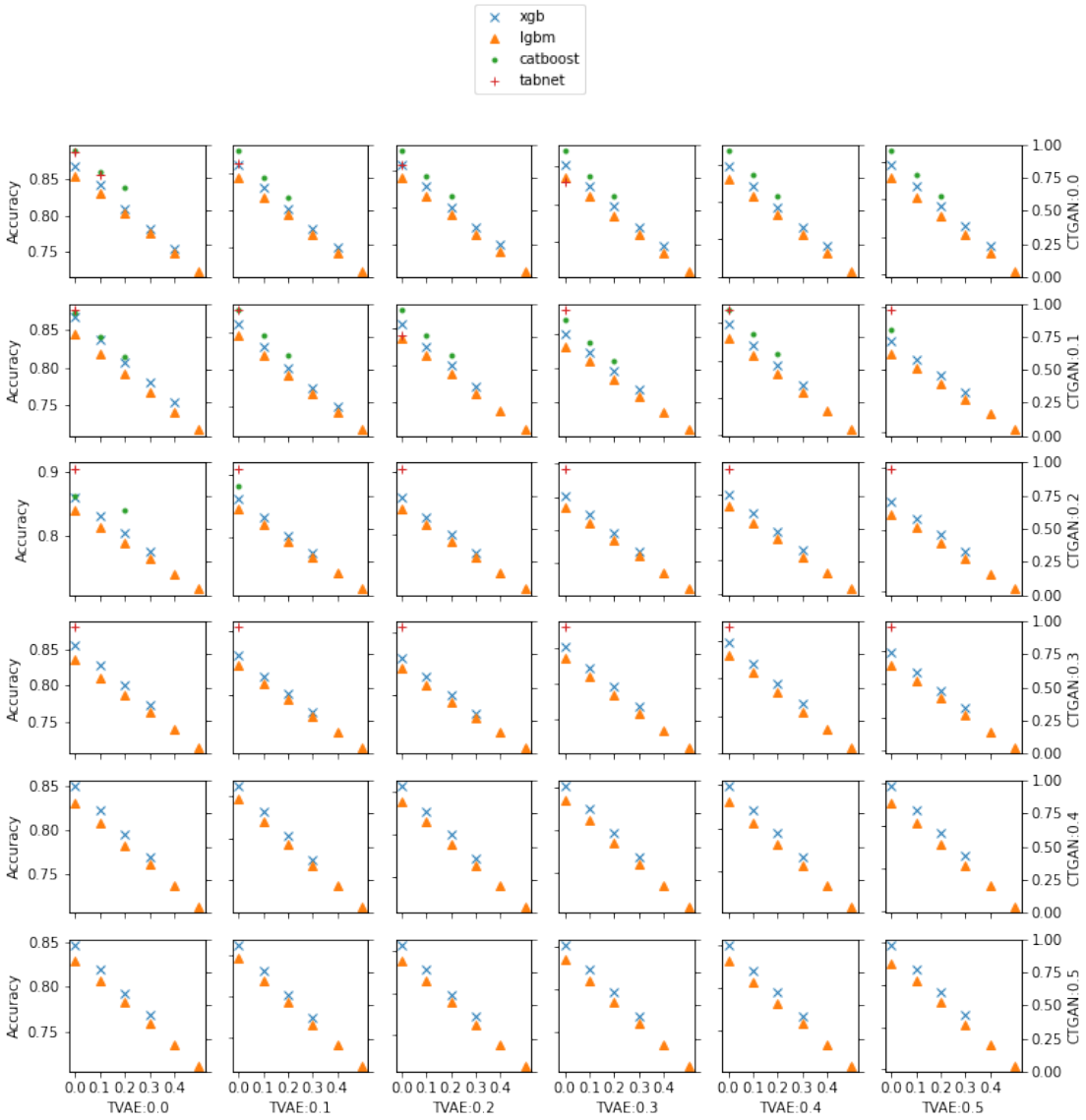


Рис. 12 — Изменения при меняющемся SMOTE, датасет Forest cover

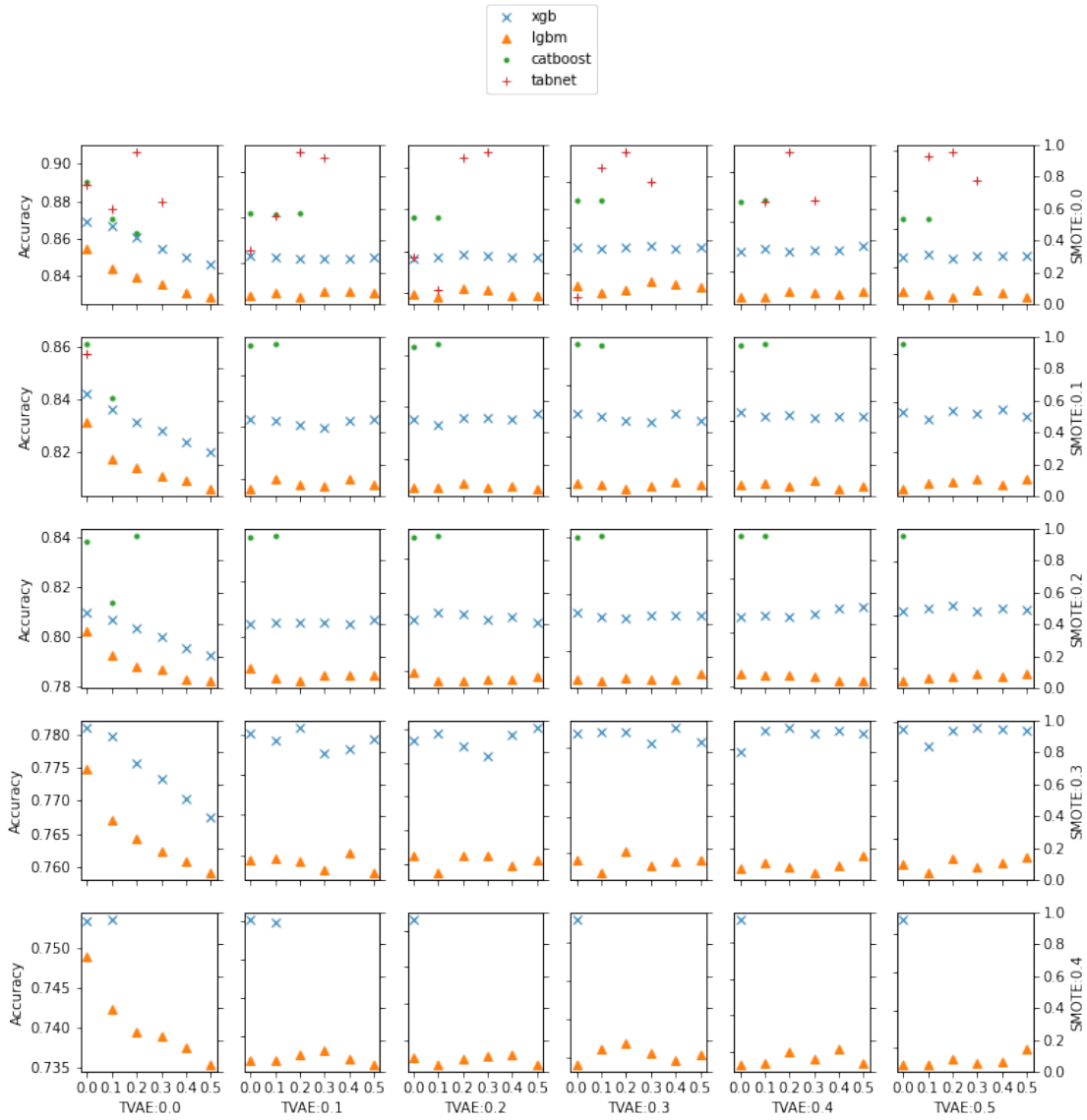


Рис. 13 — Изменения при меняющемся CTGAN, датасет Forest cover

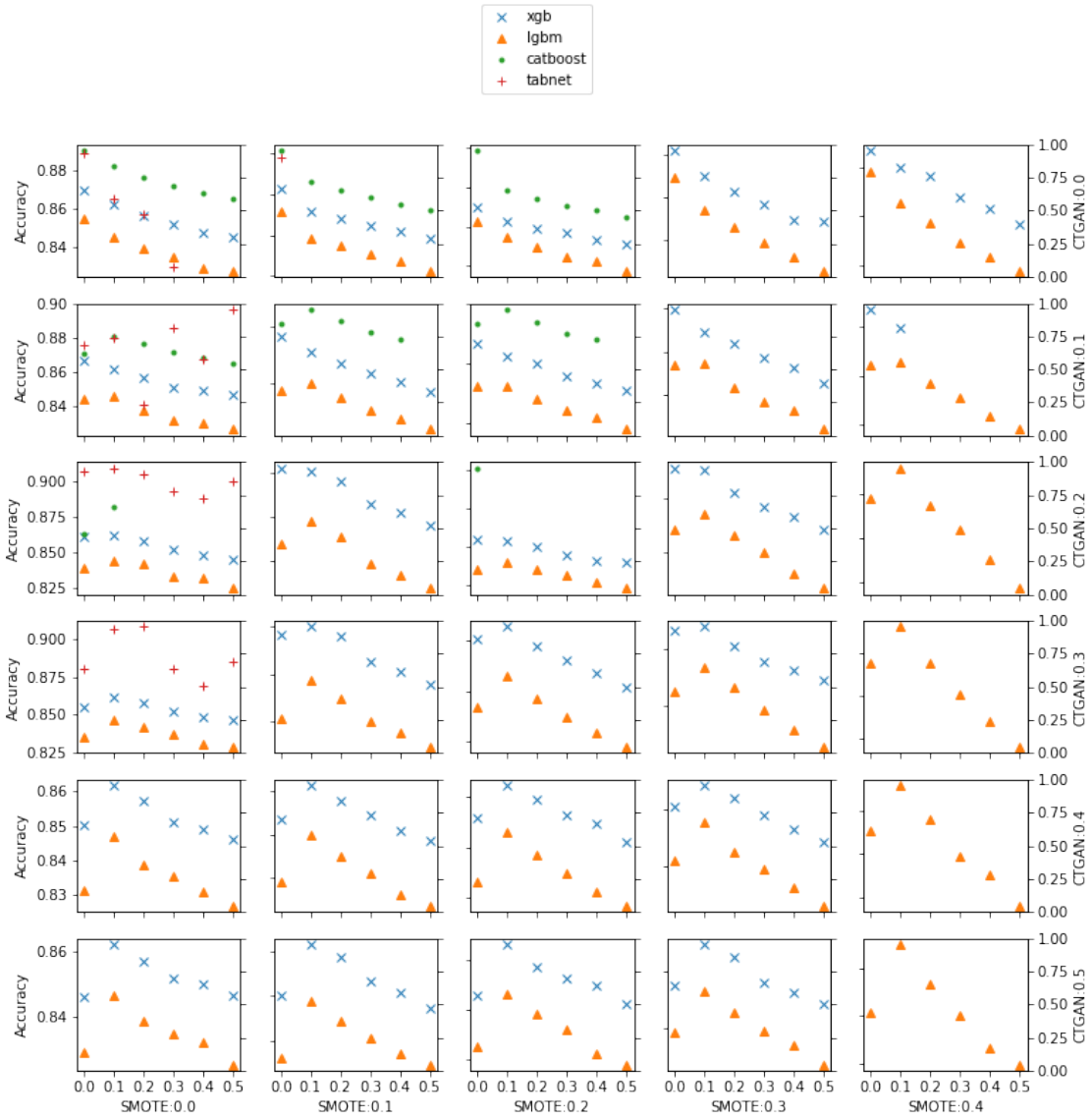


Рис. 14 — Изменения при меняющемся TVAE, датасет Forest cover

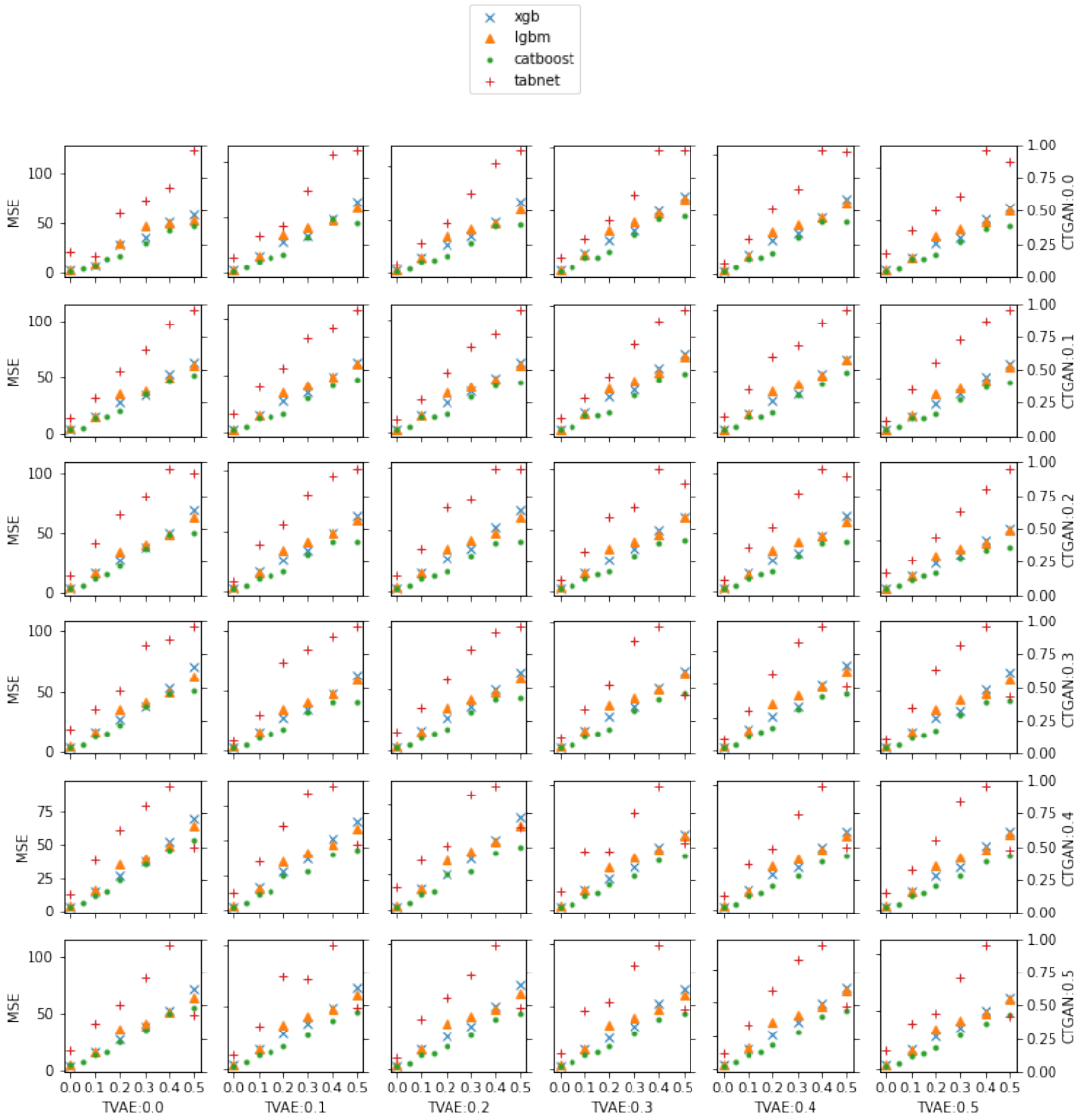


Рис. 15 — Изменения при меняющемся SMOTE, датасет Airfoil

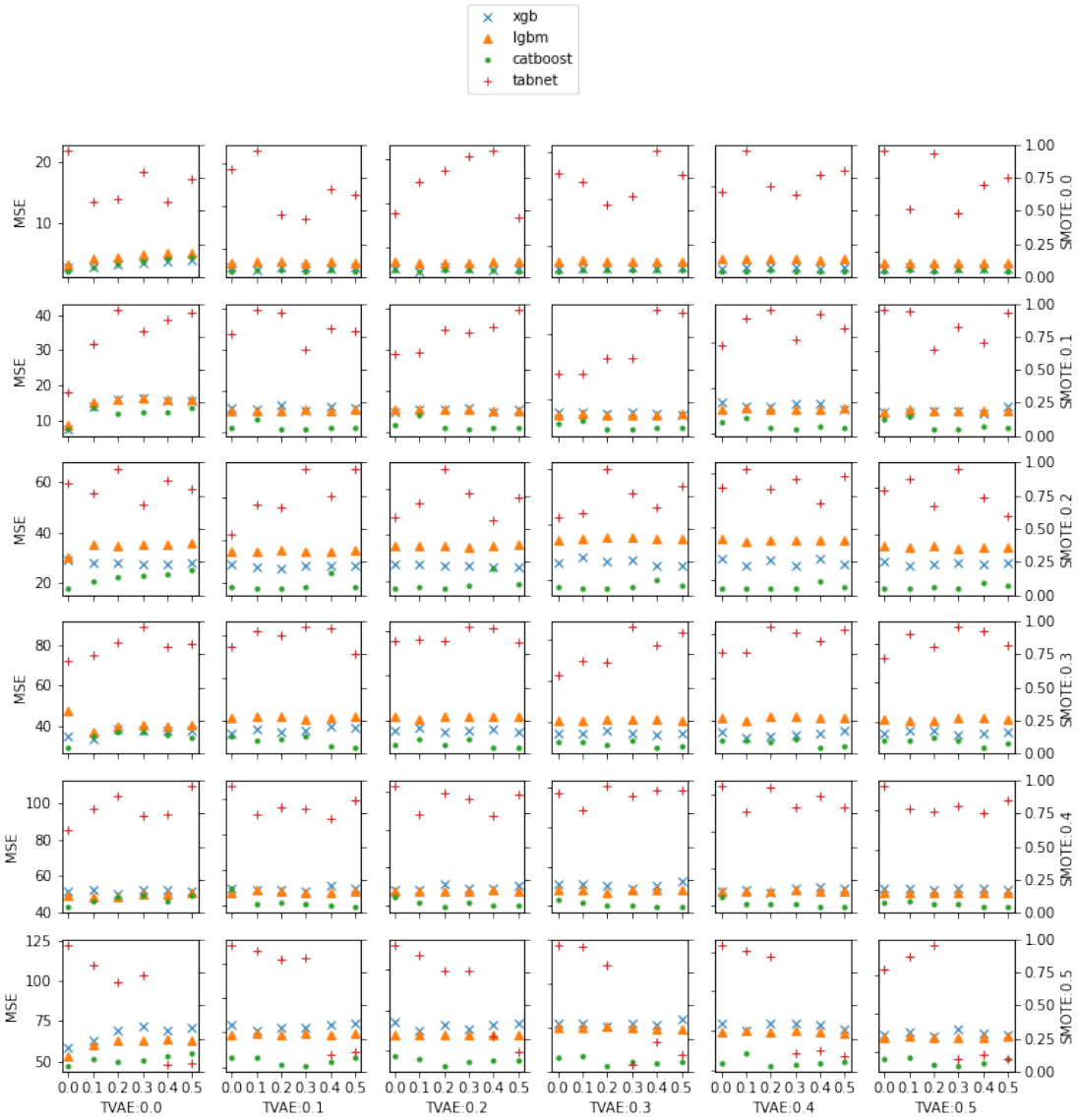


Рис. 16 — Изменения при меняющемся CTGAN, датасет Airfoil

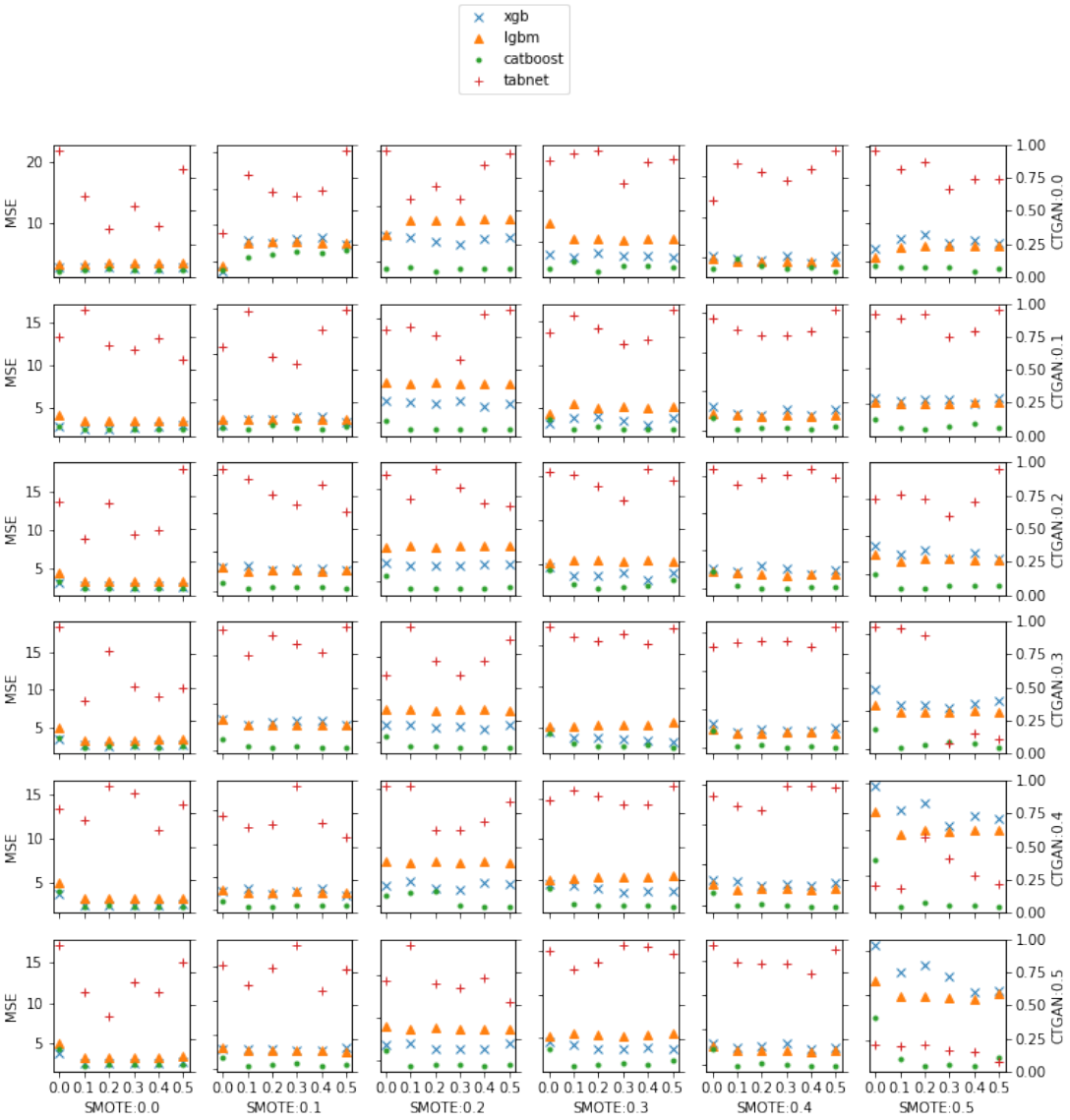


Рис. 17 — Изменения при меняющемся TVAE, датасет Airfoil

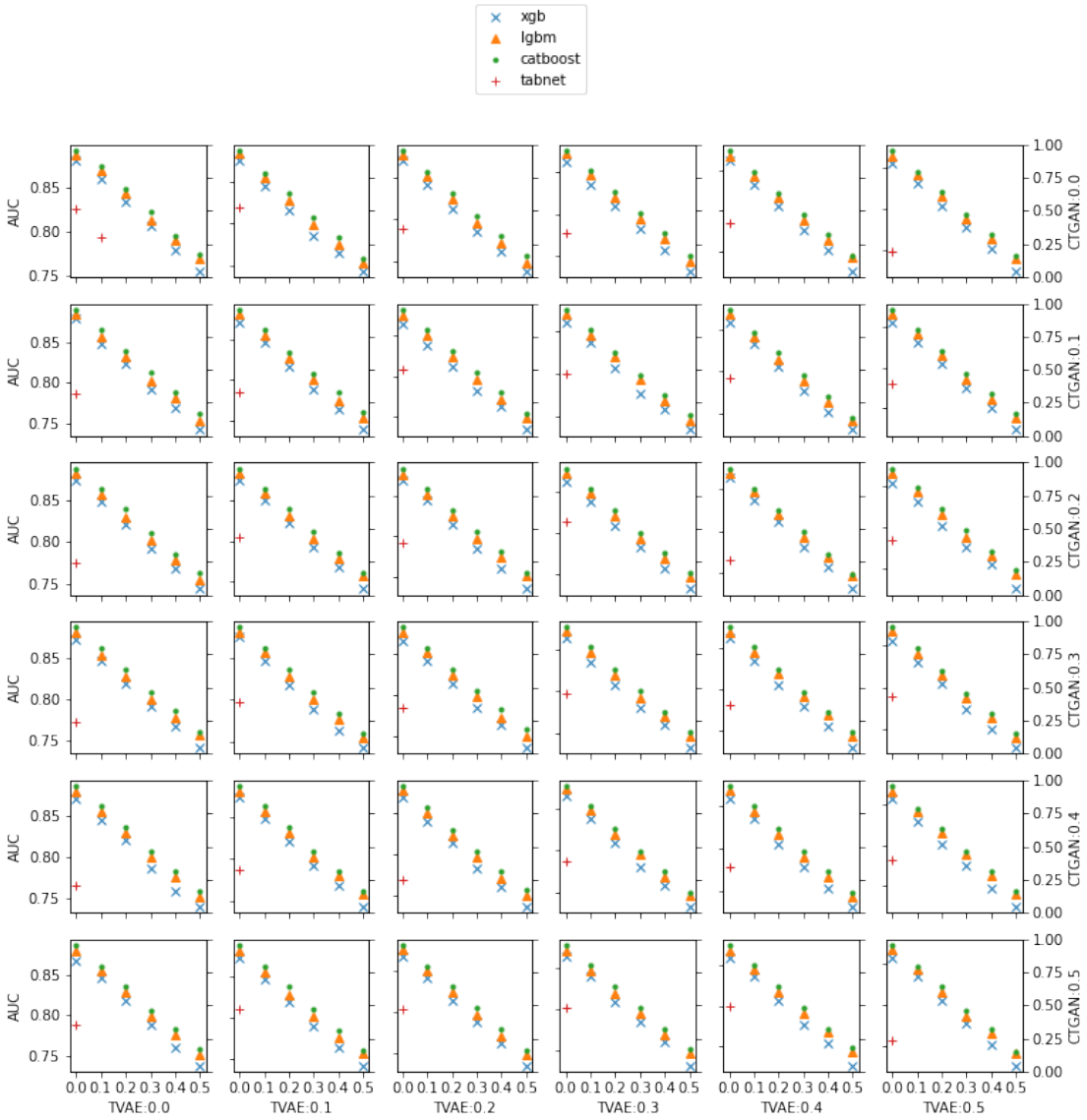


Рис. 18 — Изменения при меняющемся SMOTE, датасет Syn6

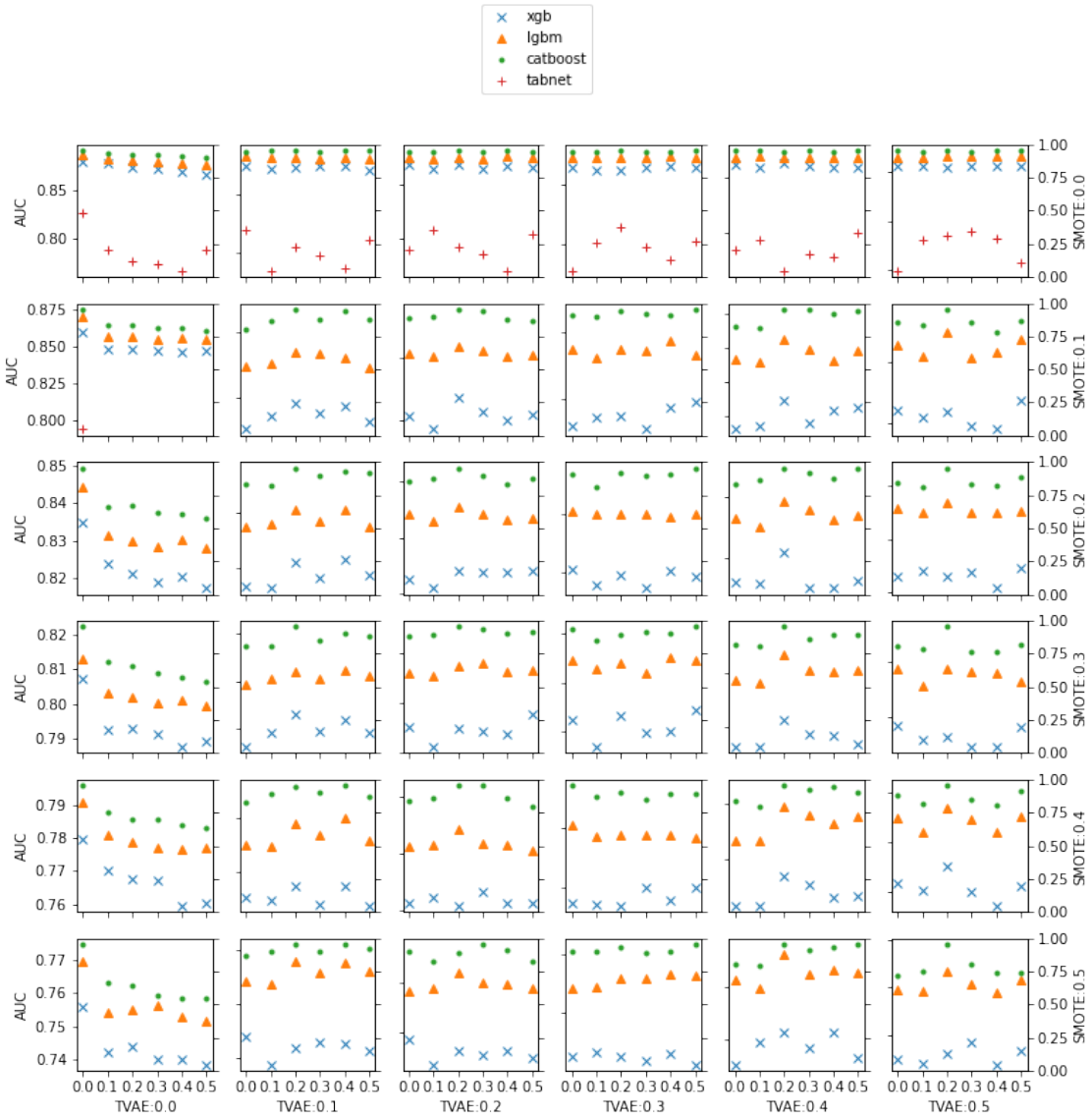


Рис. 19 — Изменения при меняющемся CTGAN, датасет Syn6

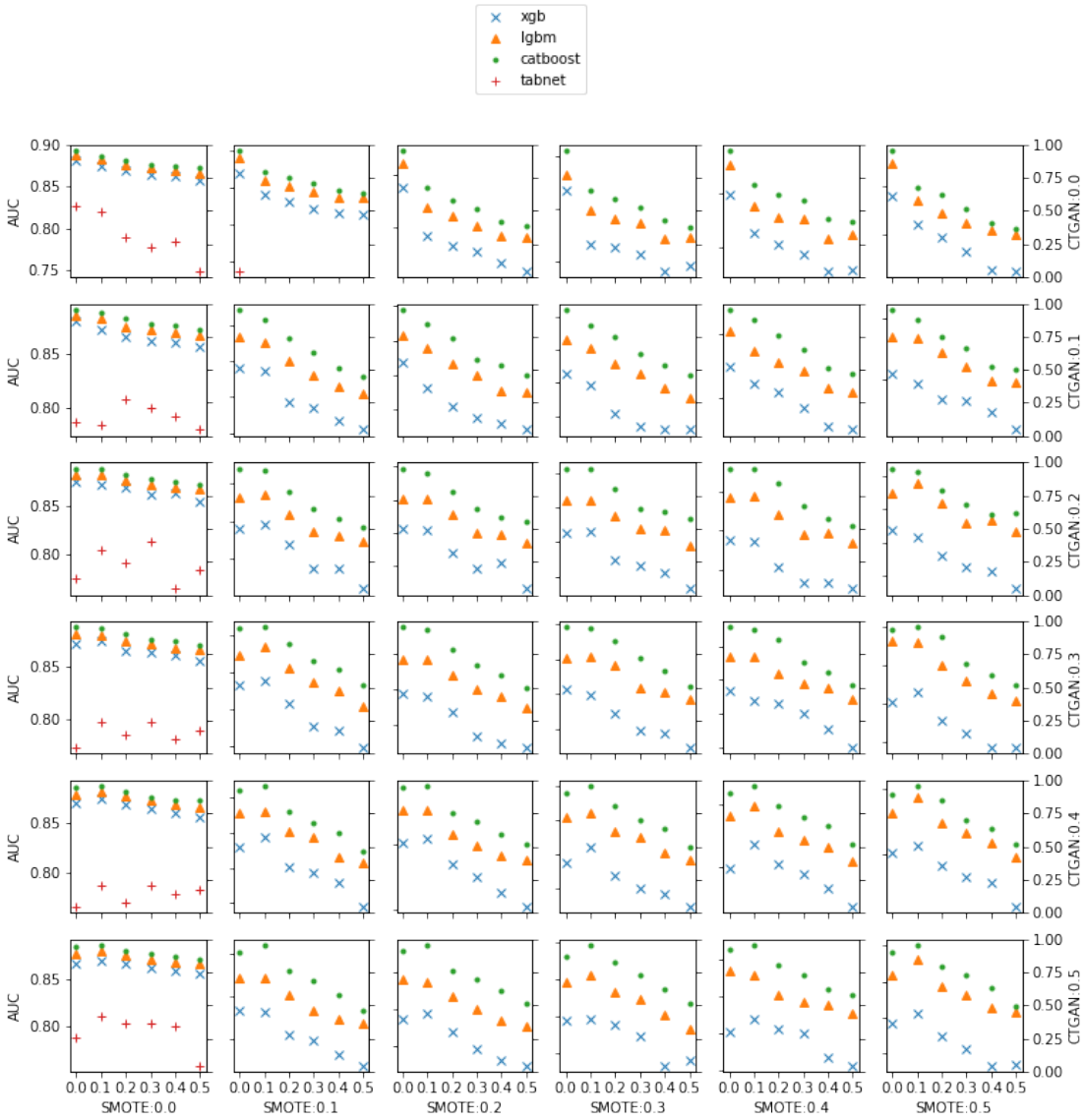


Рис. 20 — Изменения при меняющемся TVAE, датасет Syn6

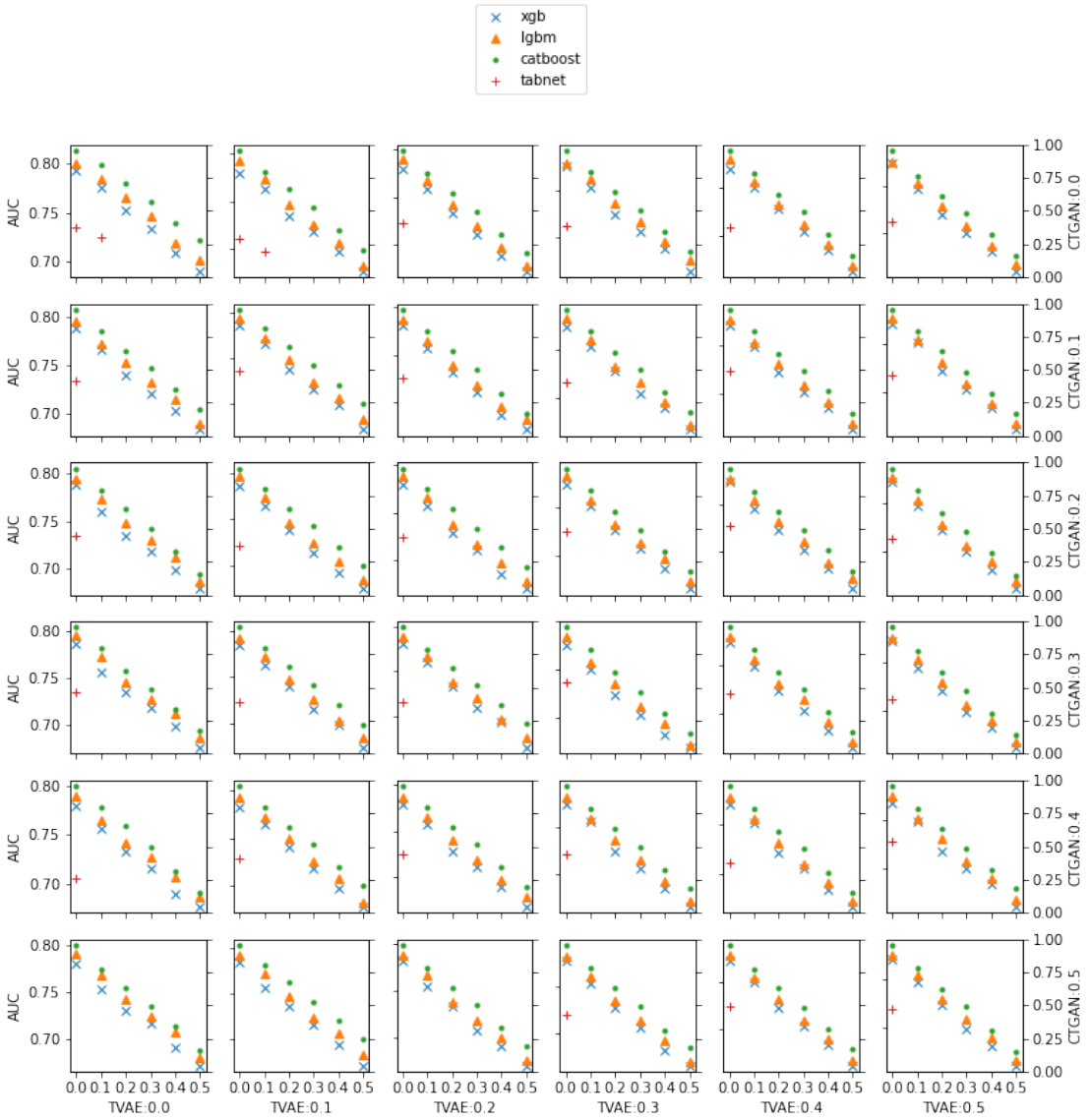


Рис. 21 — Изменения при меняющемся SMOTE, датасет Syn5

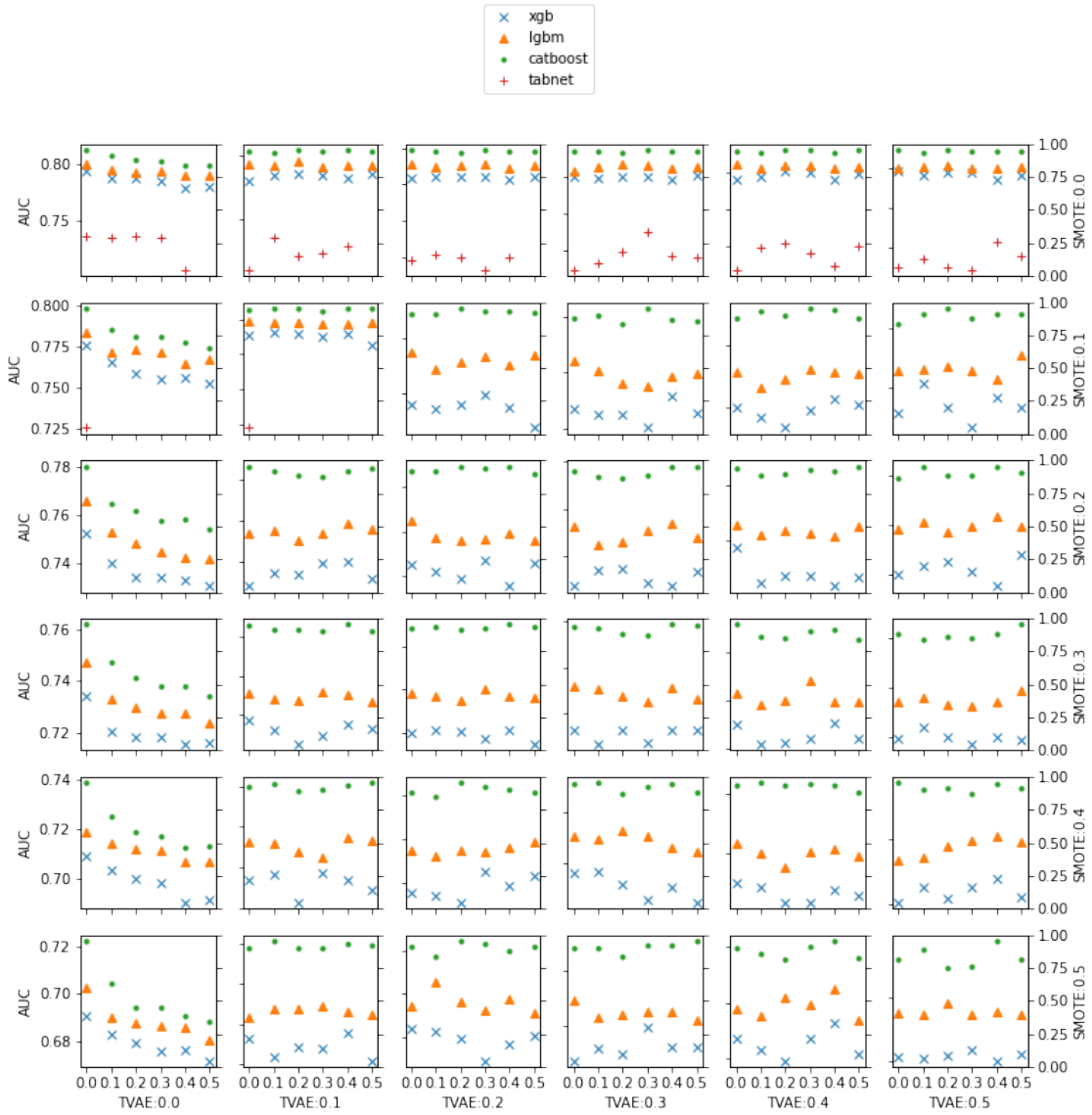


Рис. 22 — Изменения при меняющемся CTGAN, датасет Syn5

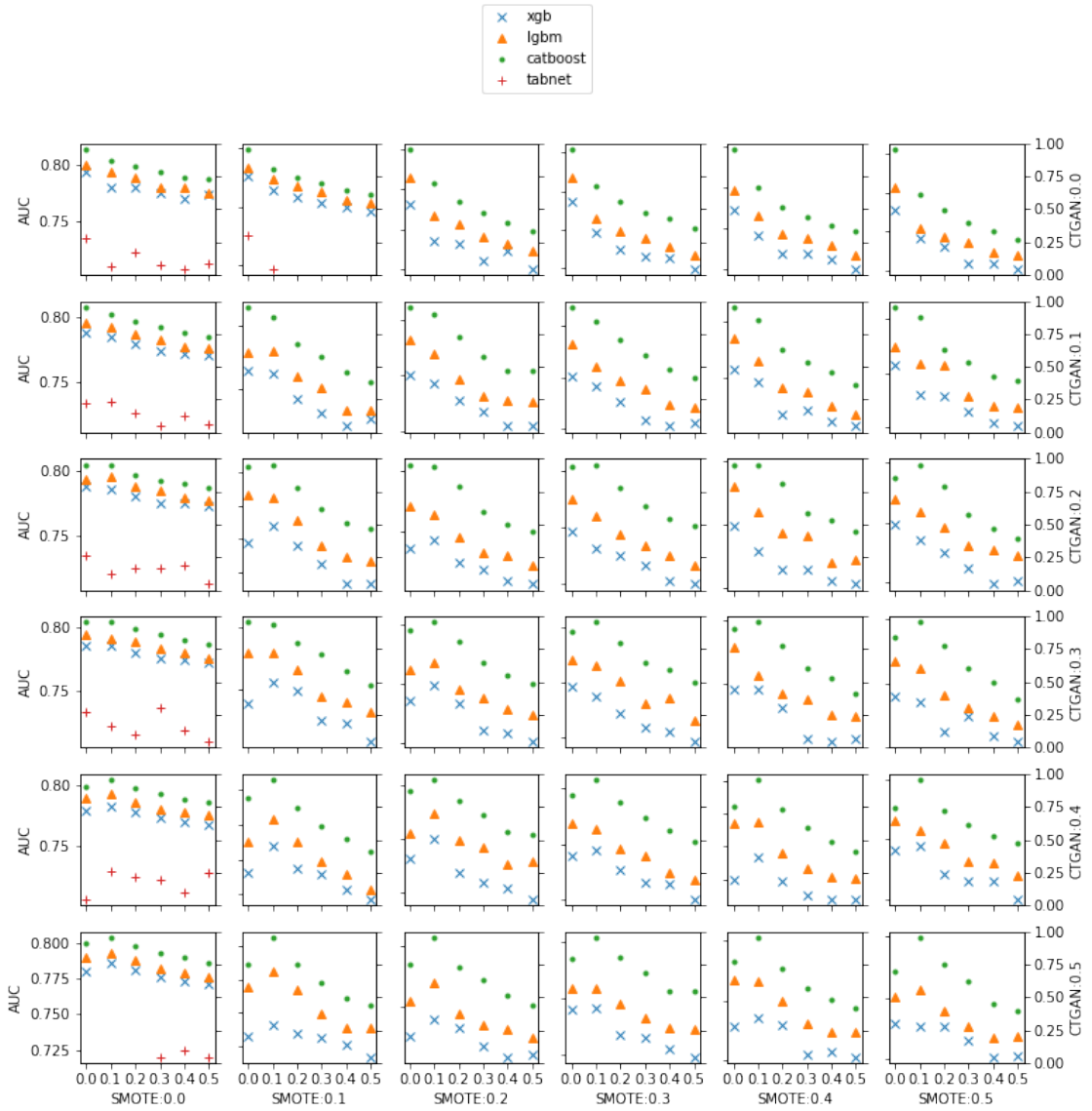


Рис. 23 — Изменения при меняющемся TVAE, датасет Syn5

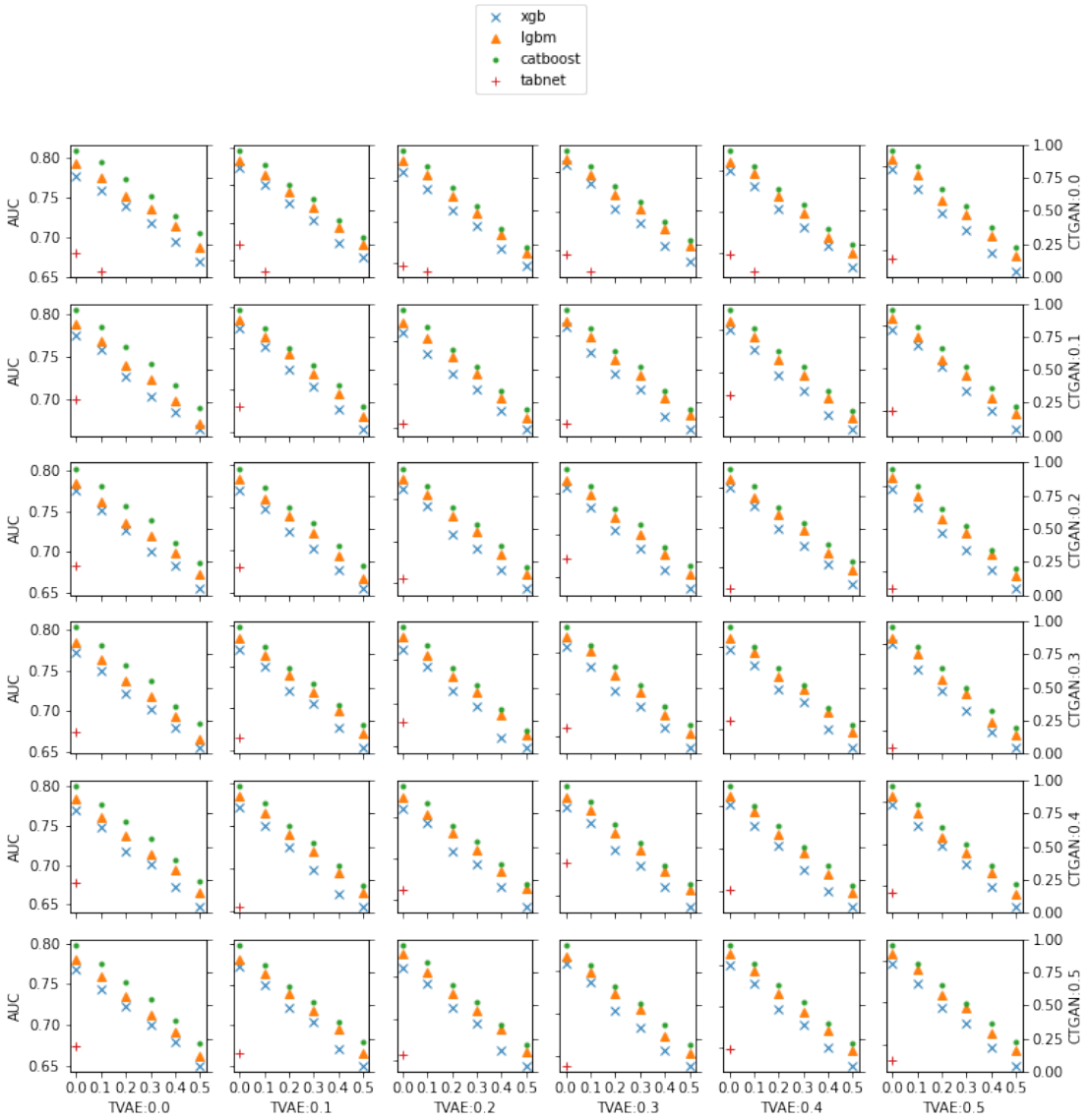


Рис. 24 — Изменения при меняющемся SMOTE, датасет Syn4

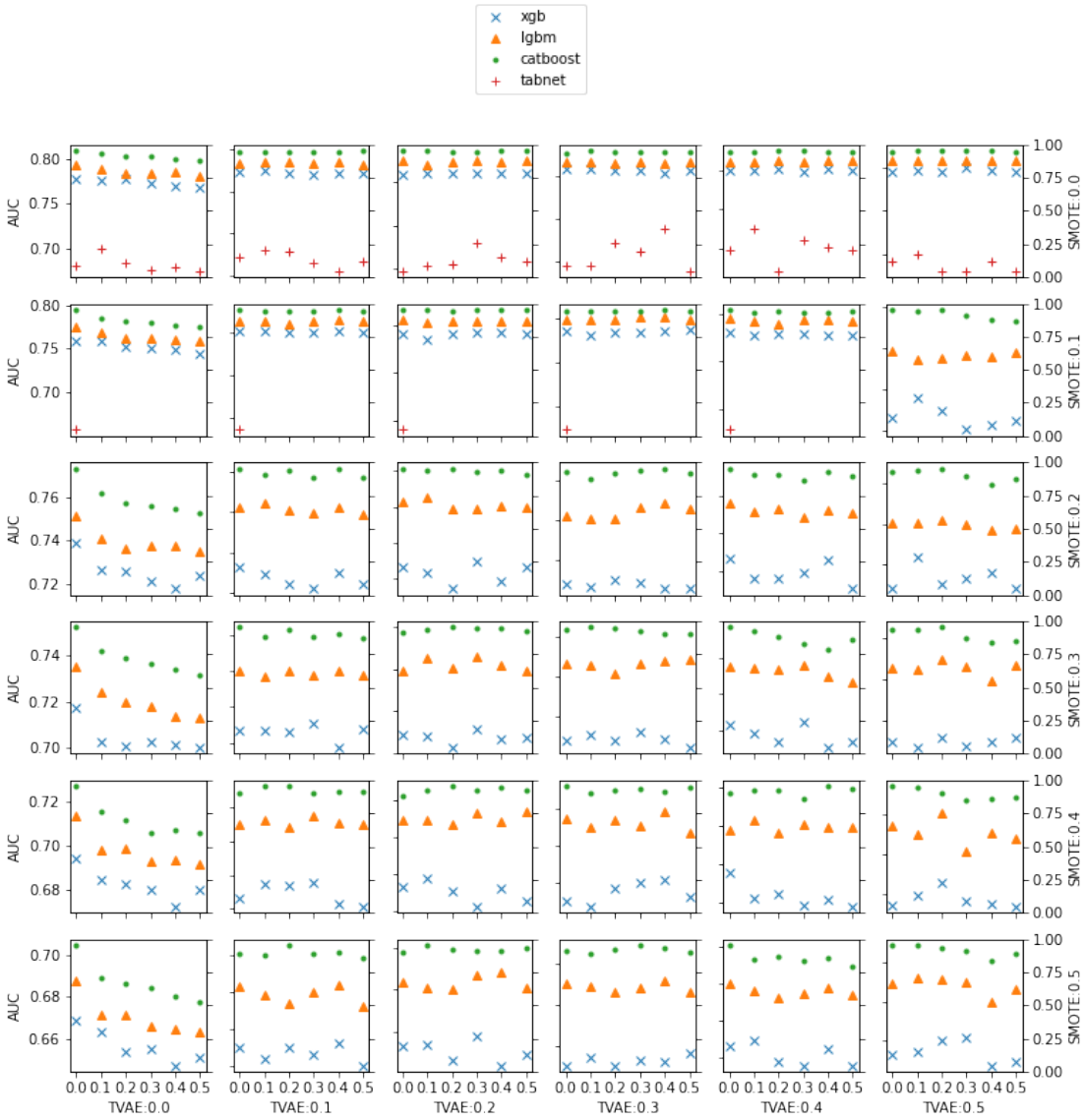


Рис. 25 — Изменения при меняющемся CTGAN, датасет Syn4

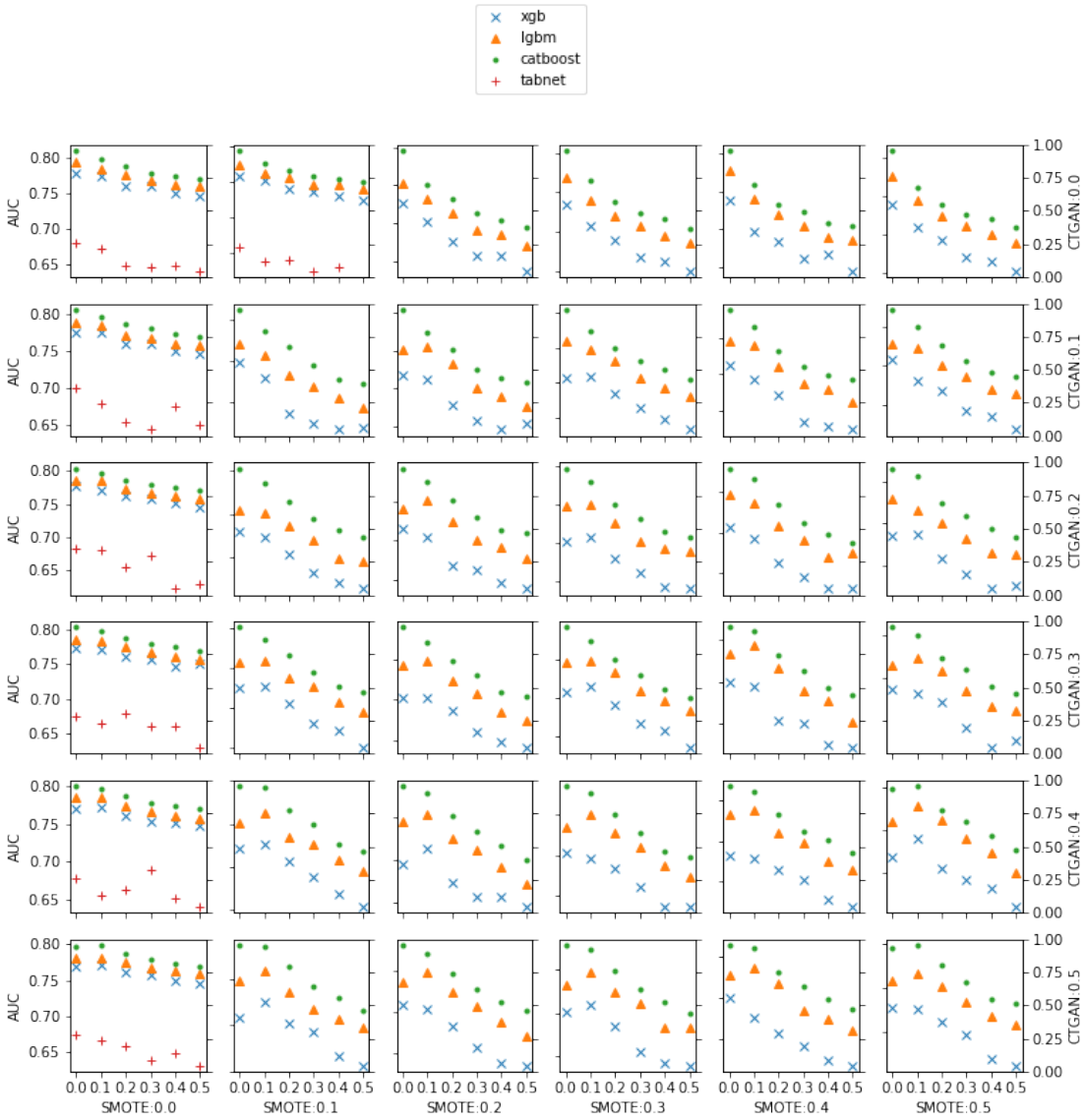


Рис. 26 — Изменения при меняющемся TVAE, датасет Syn4

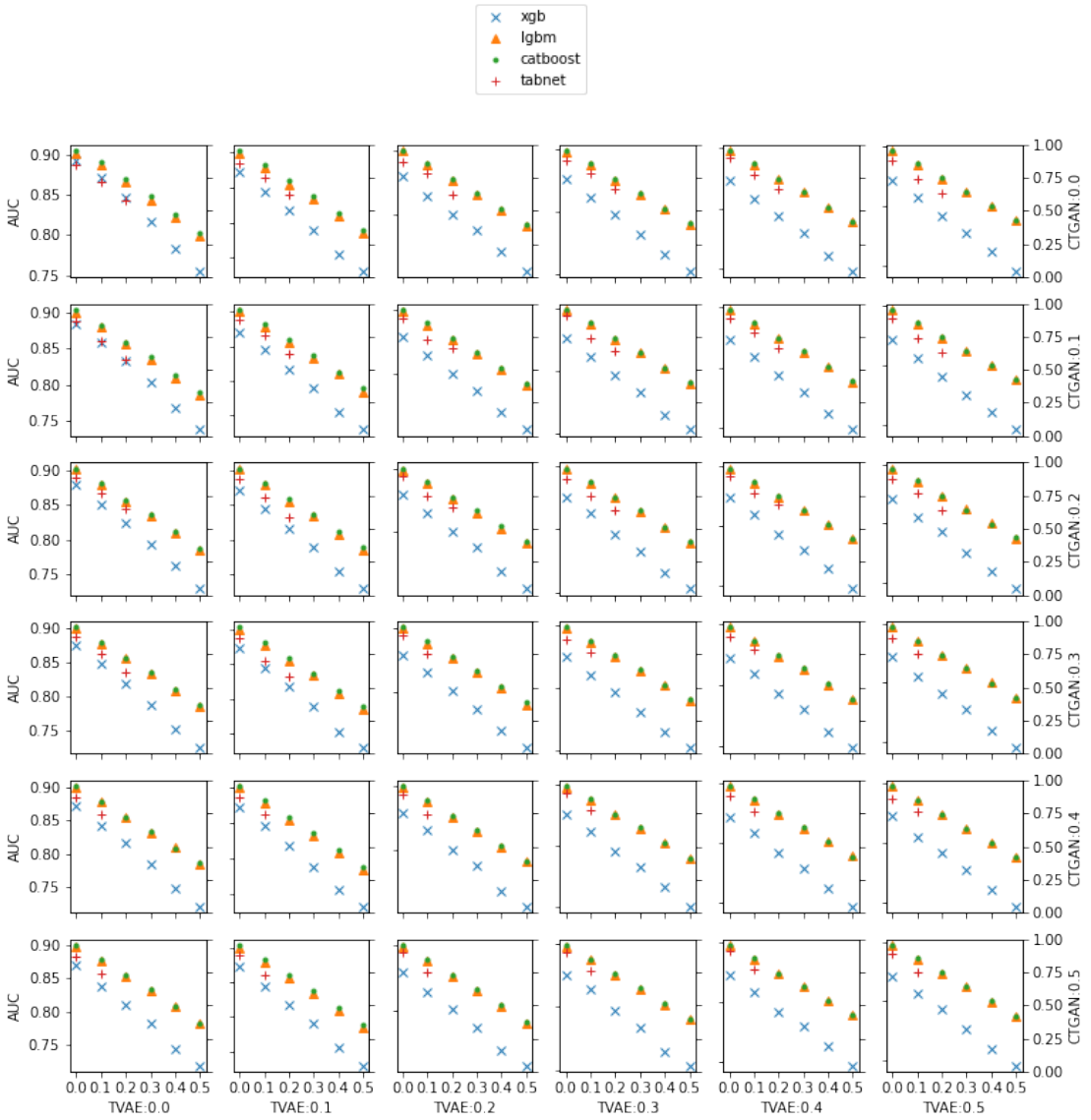


Рис. 27 — Изменения при меняющемся SMOTE, датасет Syn3

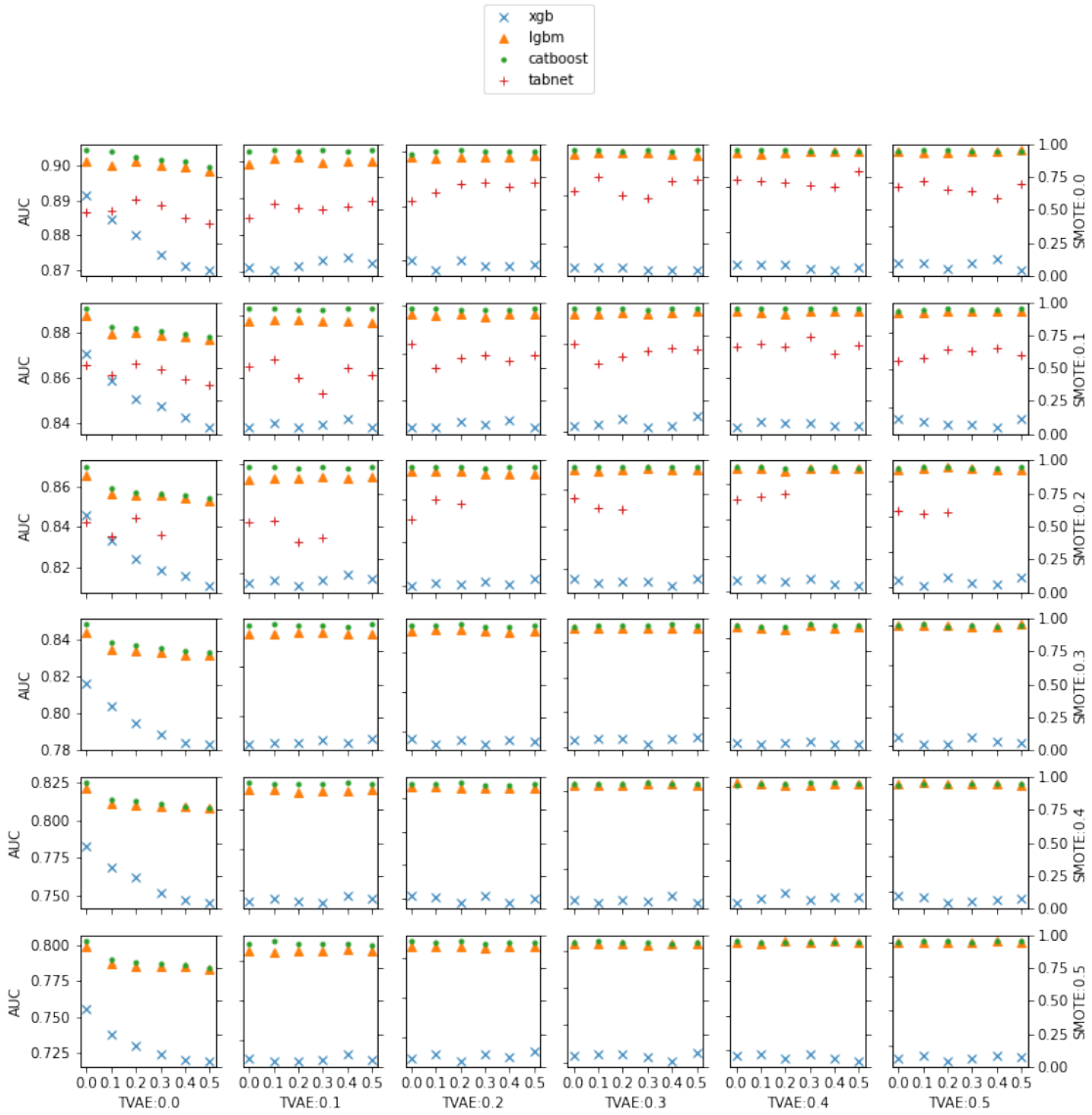


Рис. 28 — Изменения при меняющемся CTGAN, датасет Syn3

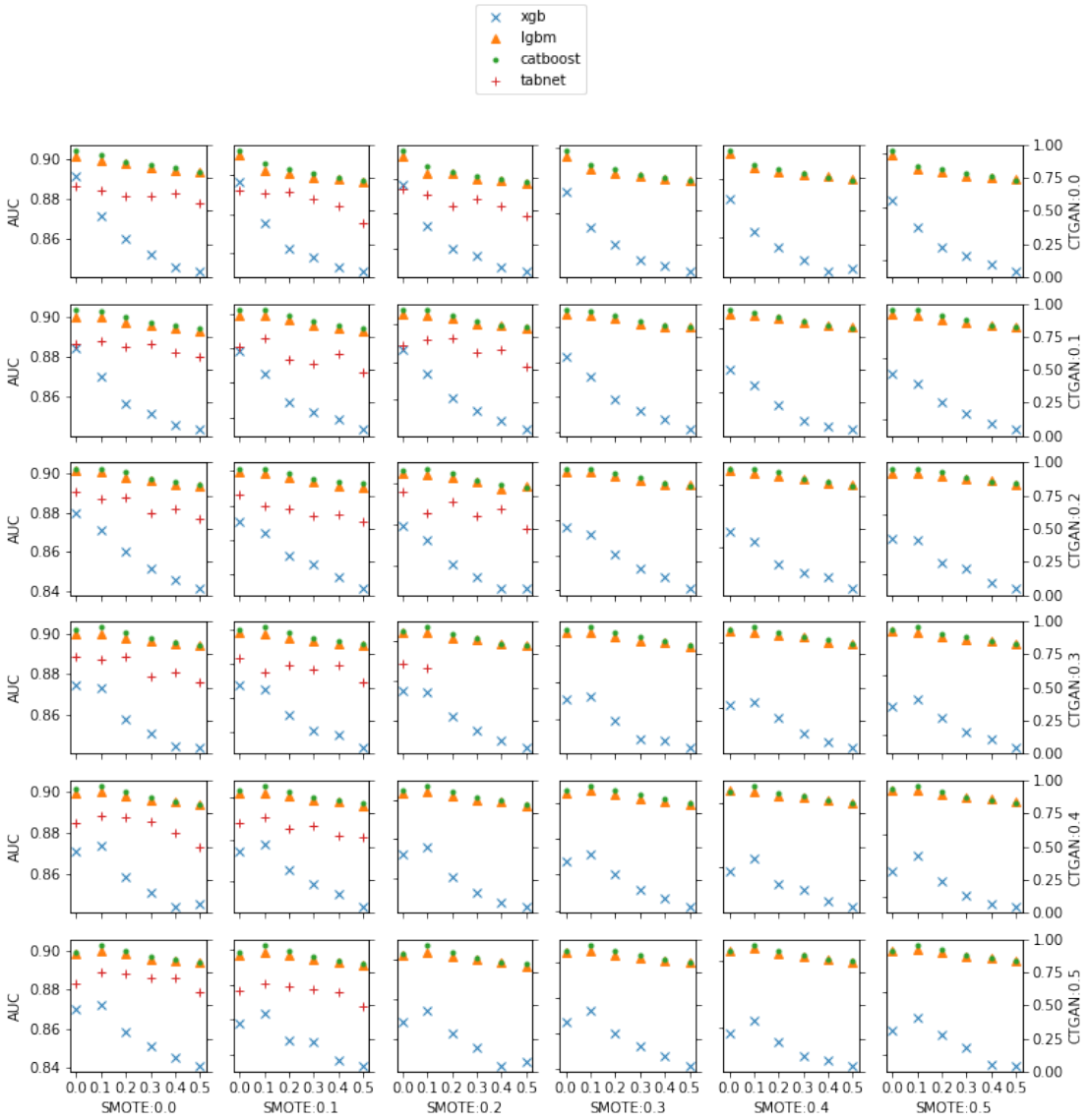


Рис. 29 — Изменения при меняющемся TVAE, датасет Syn3

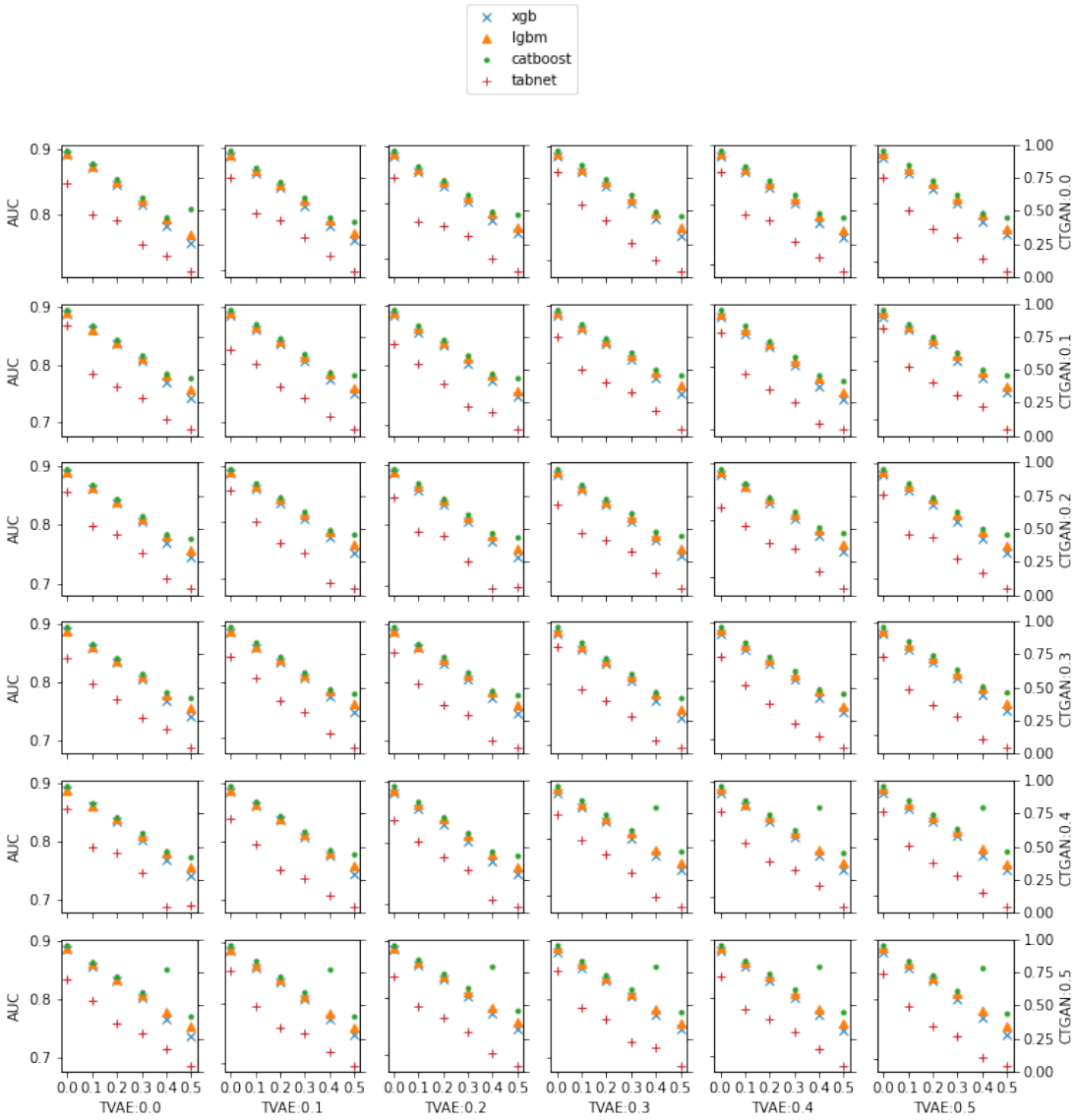


Рис. 30 — Изменения при меняющемся SMOTE, датасет Syn2

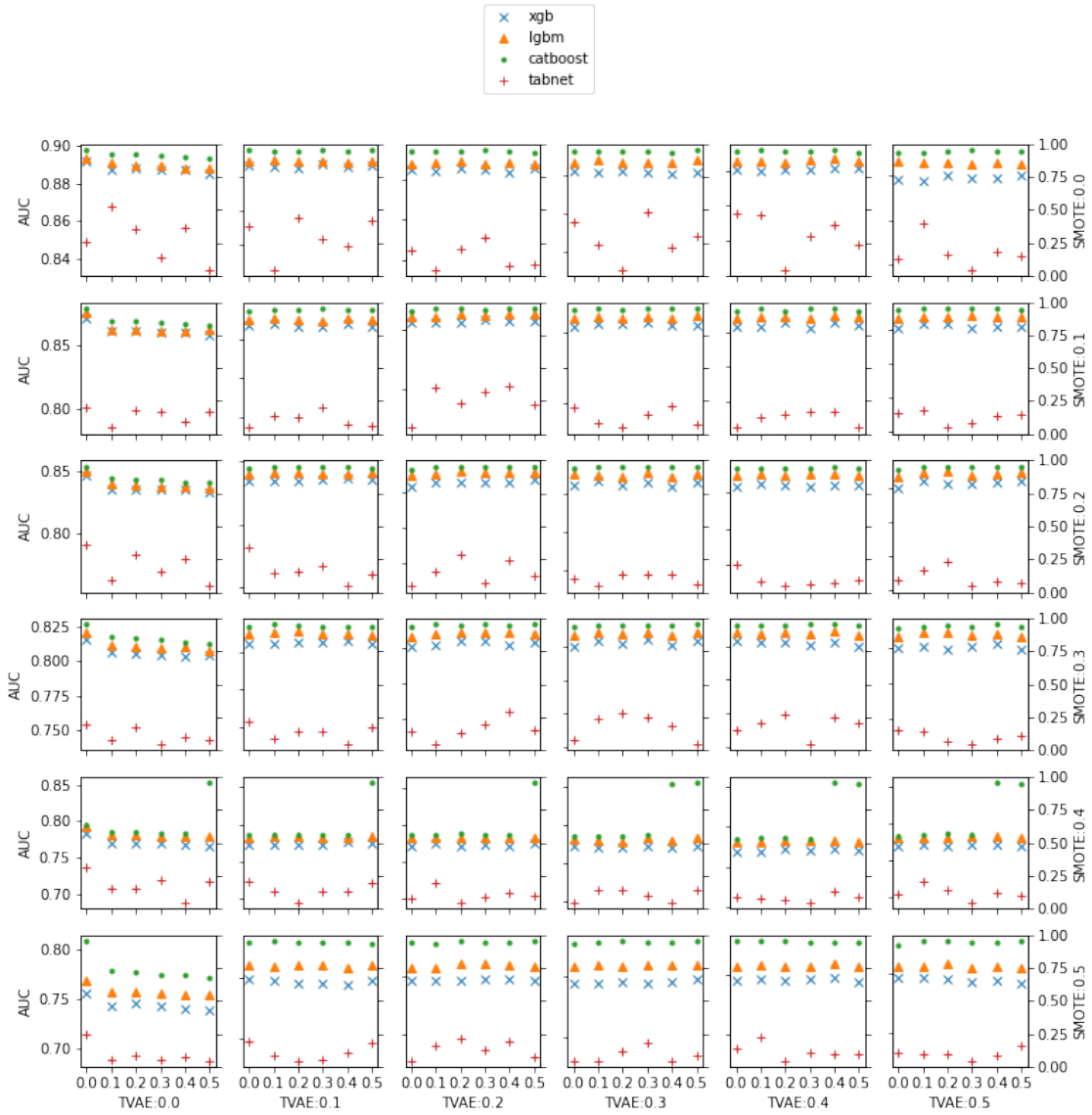


Рис. 31 — Изменения при меняющемся CTGAN, датасет Syn2

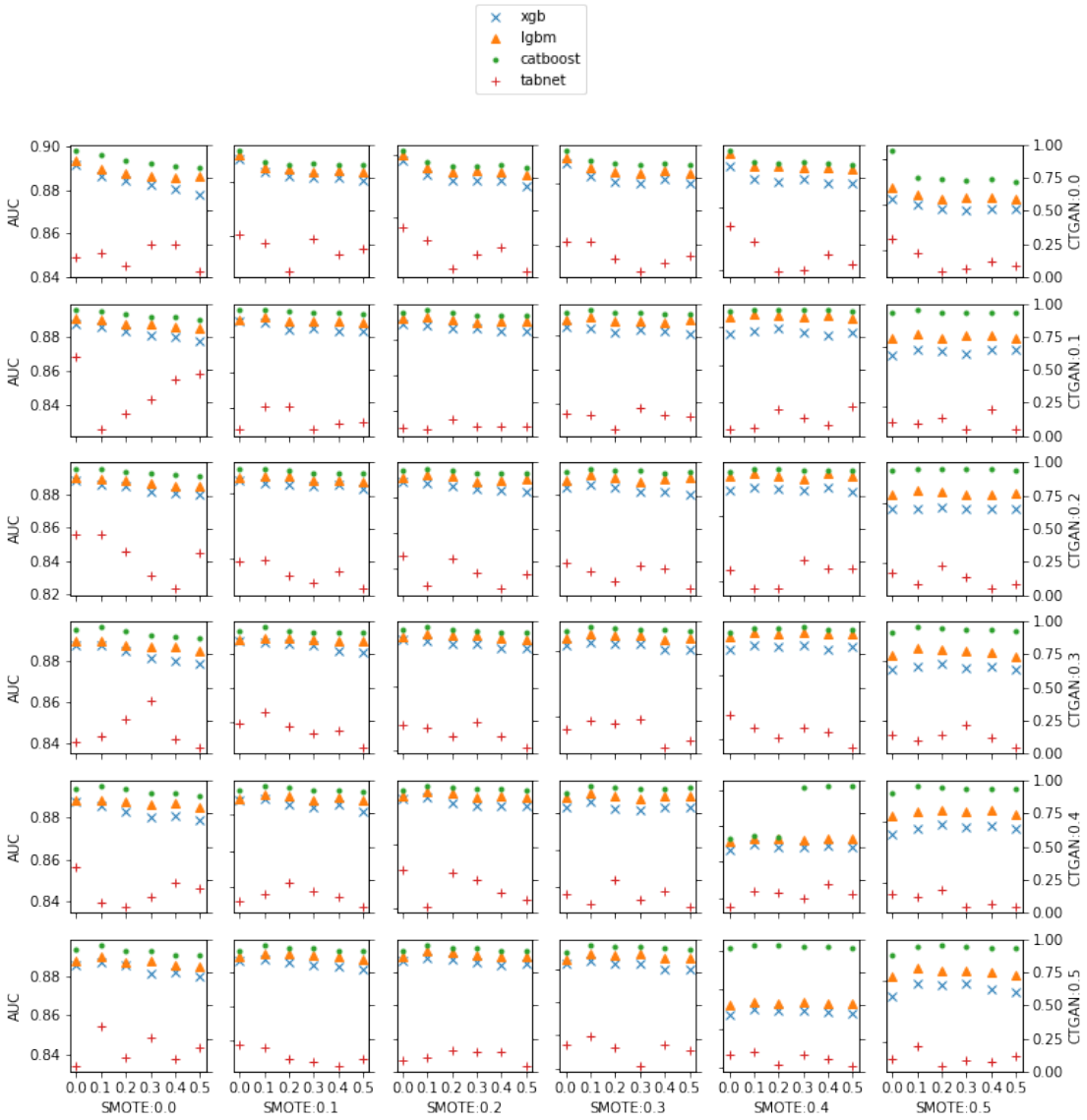


Рис. 32 — Изменения при меняющемся TVAE, датасет Syn2

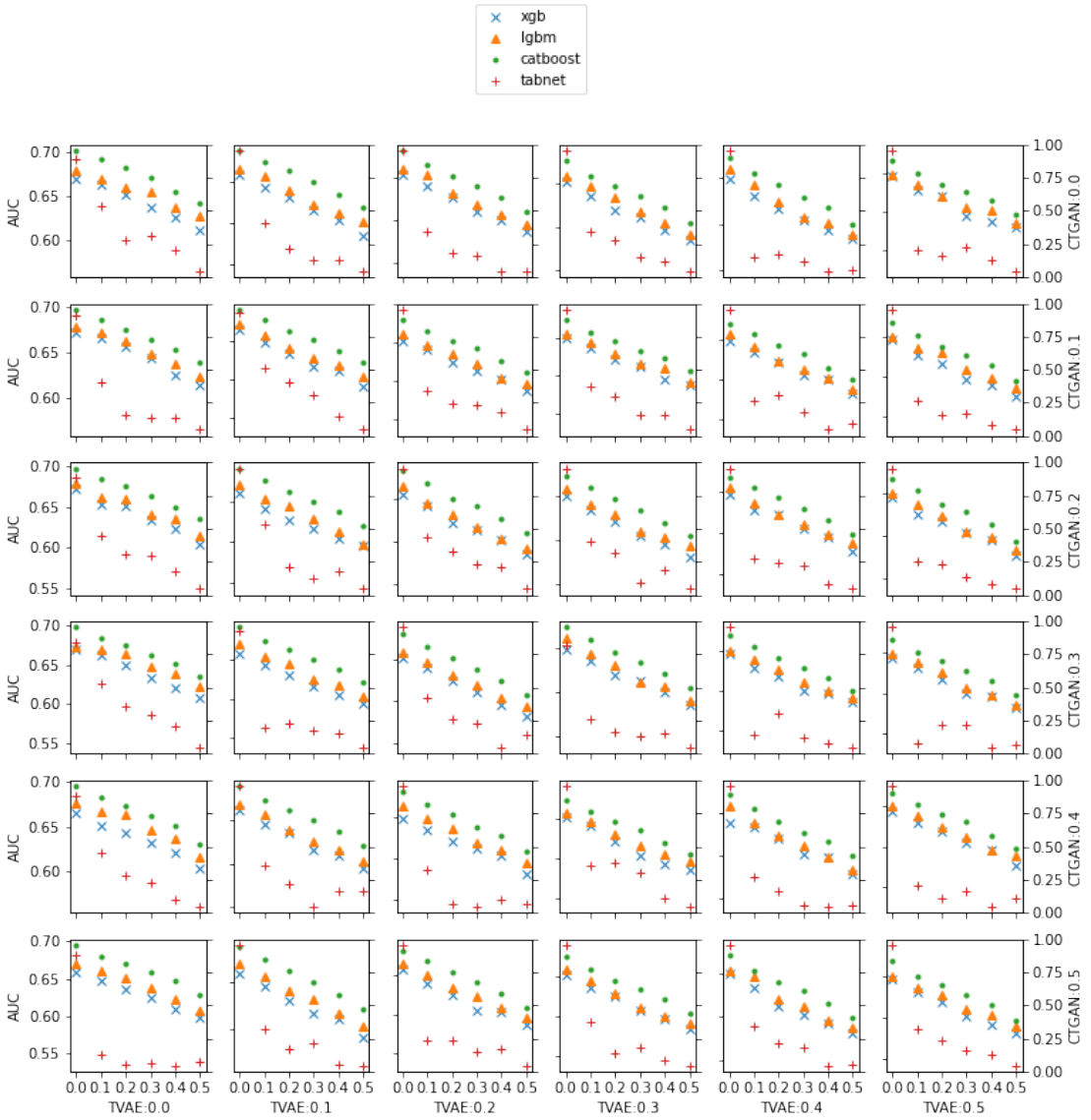


Рис. 33 — Изменения при меняющемся SMOTE, датасет Syn

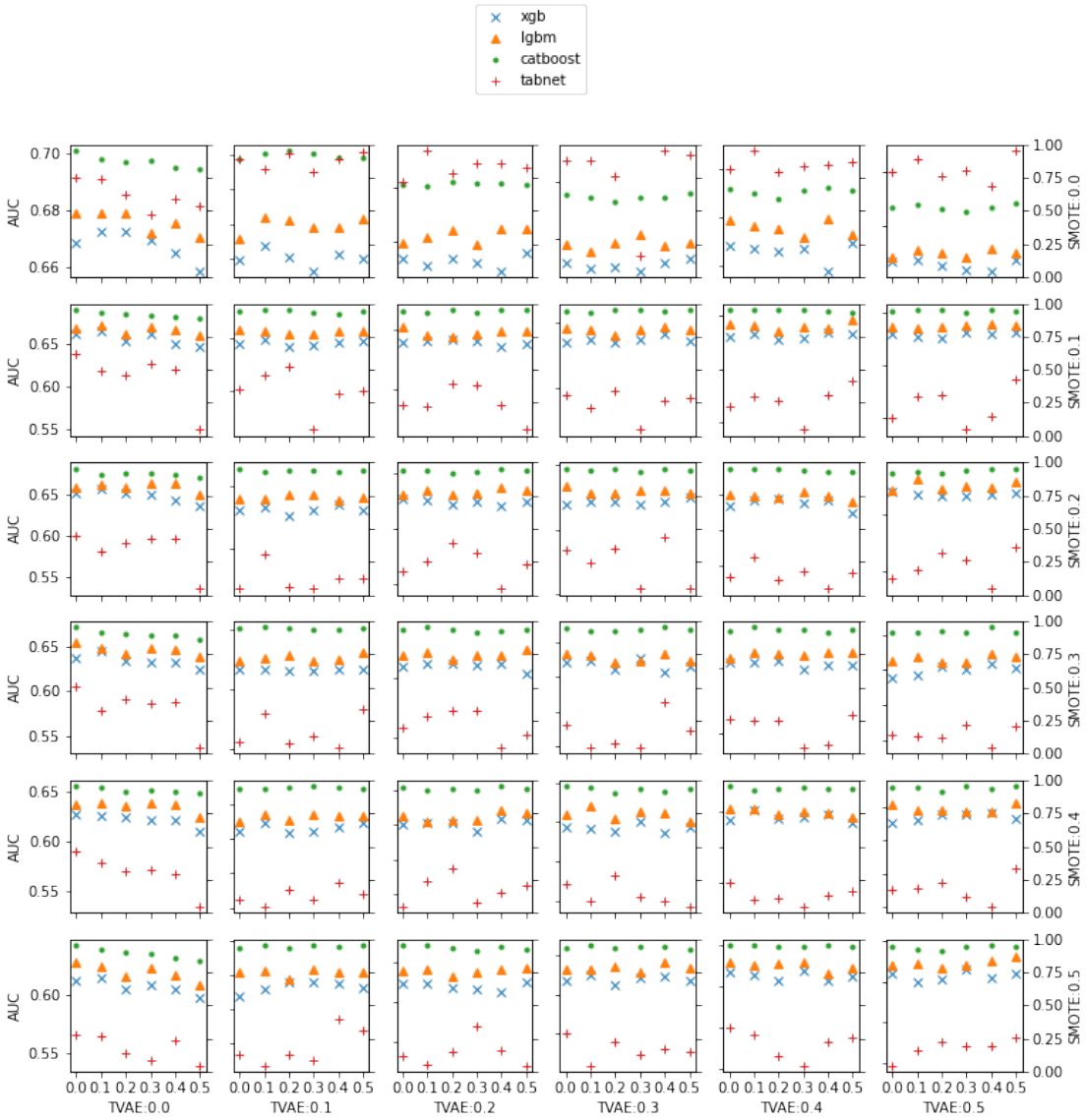


Рис. 34 — Изменения при меняющемся CTGAN, датасет Syn

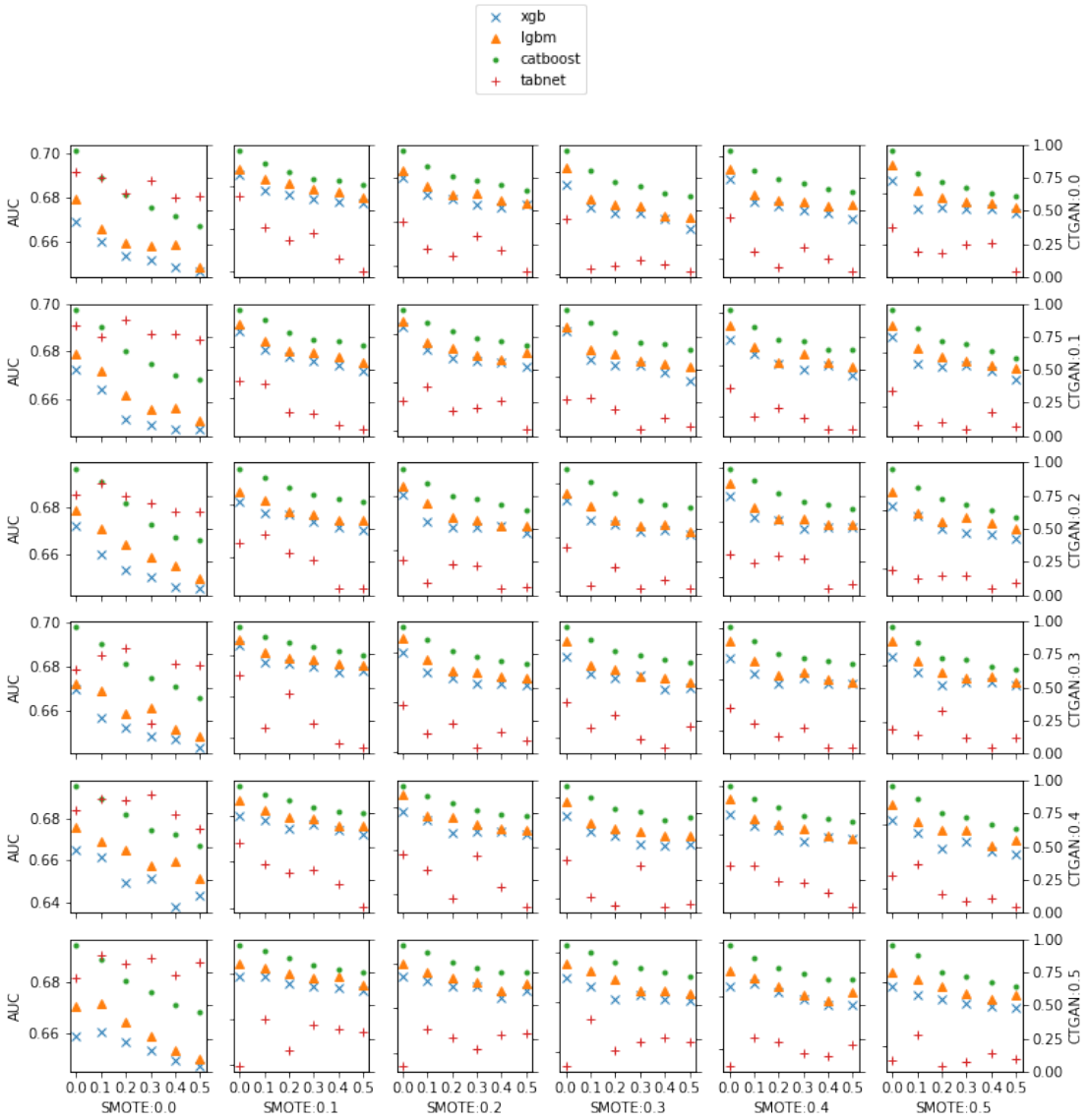


Рис. 35 — Изменения при меняющемся TVAE, датасет Syn

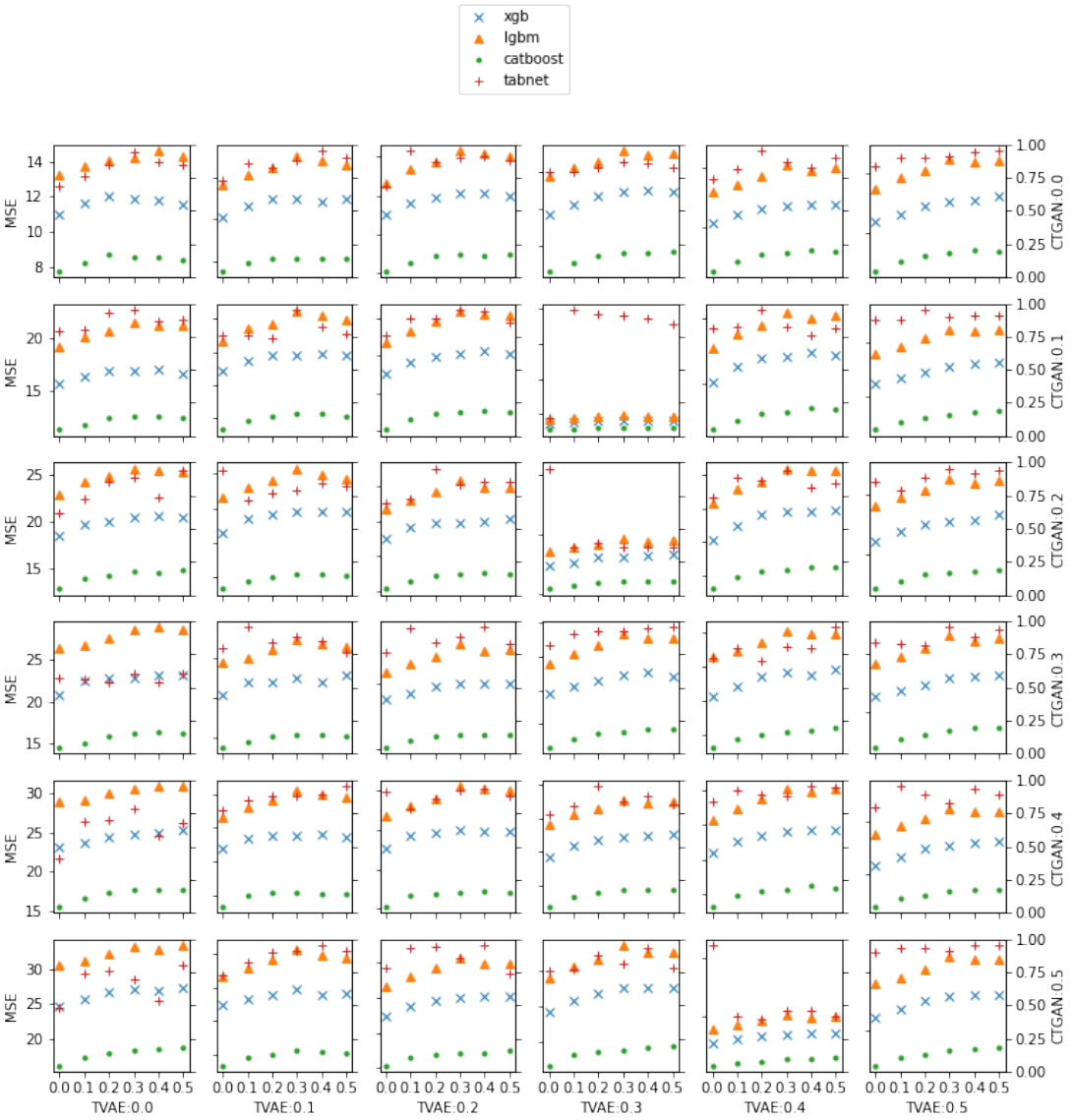


Рис. 36 — Изменения при меняющемся SMOTE, датасет Sarcos

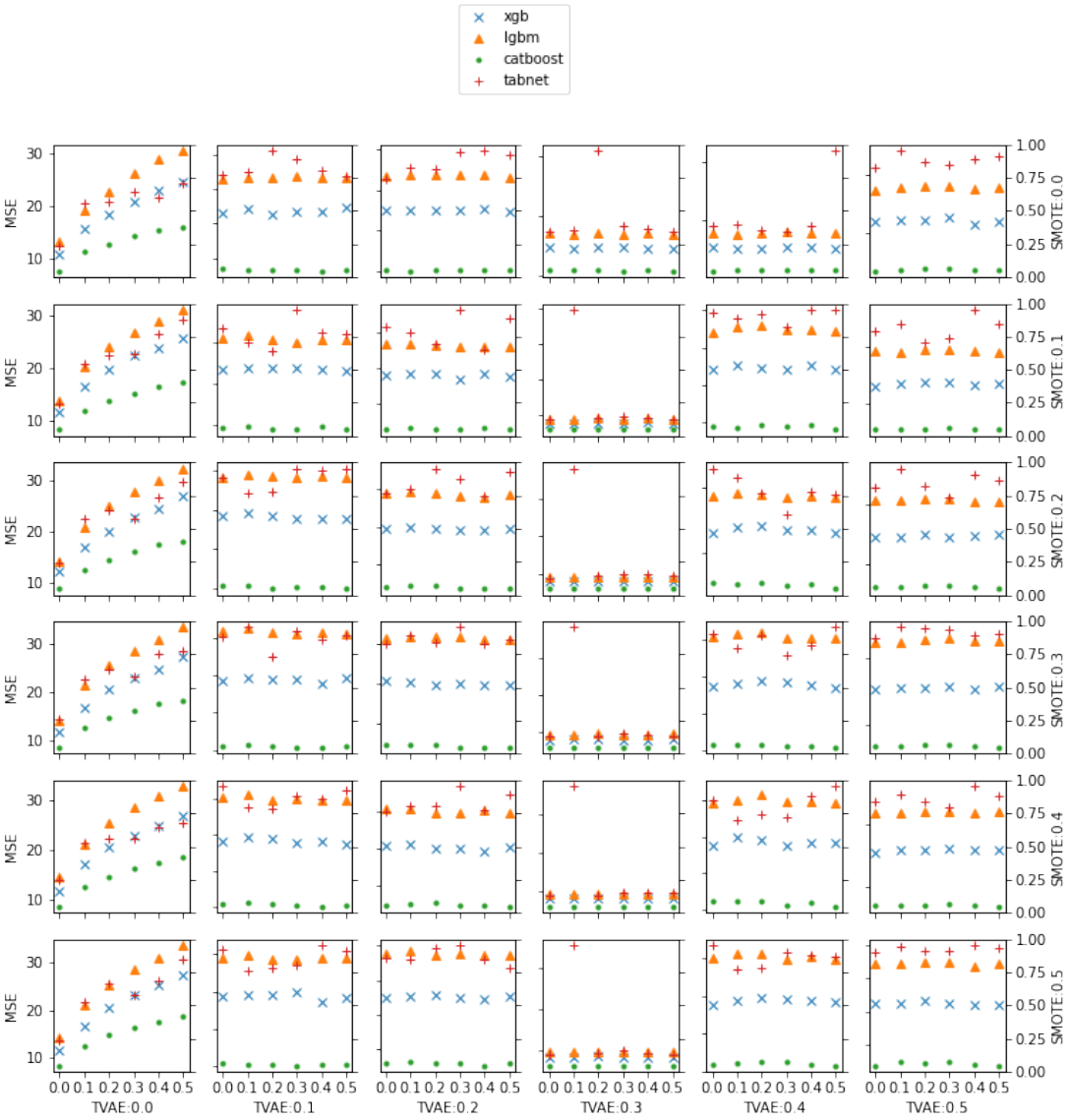


Рис. 37 — Изменения при меняющемся CTGAN, датасет Sarcos

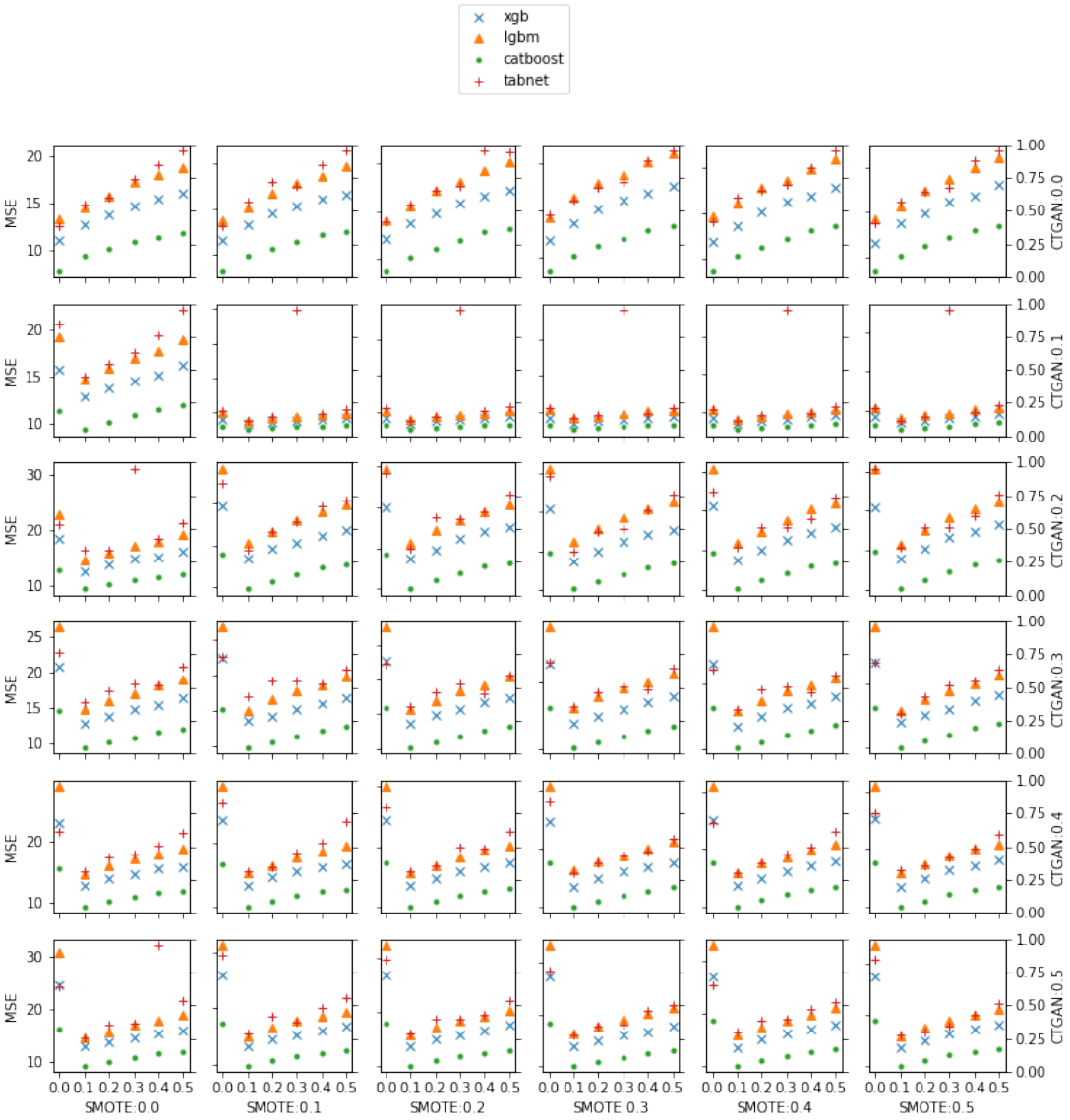


Рис. 38 — Изменения при меняющемся TVAE, датасет Sarcos

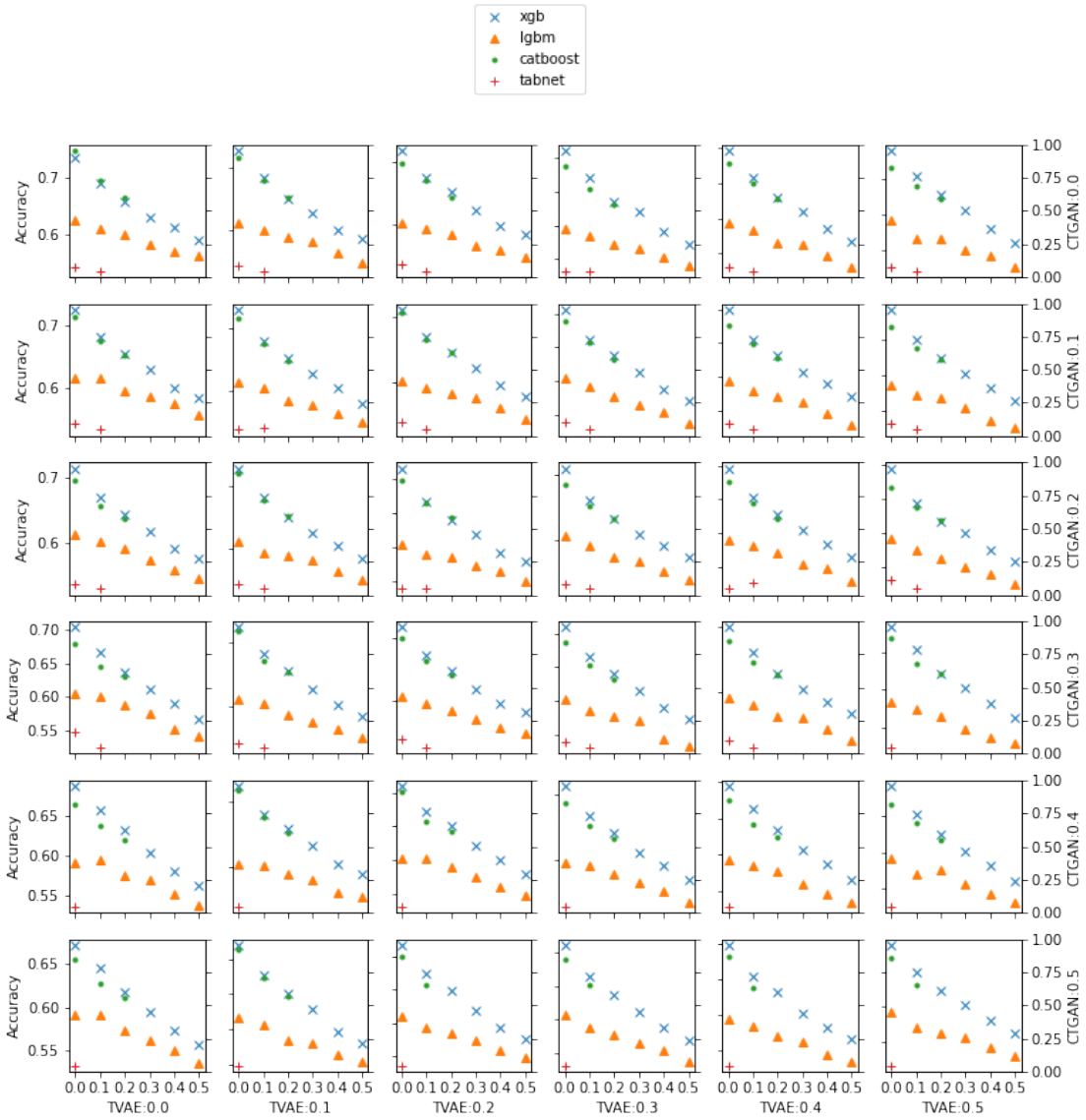


Рис. 39 — Изменения при меняющемся SMOTE, датасет Pokerhand

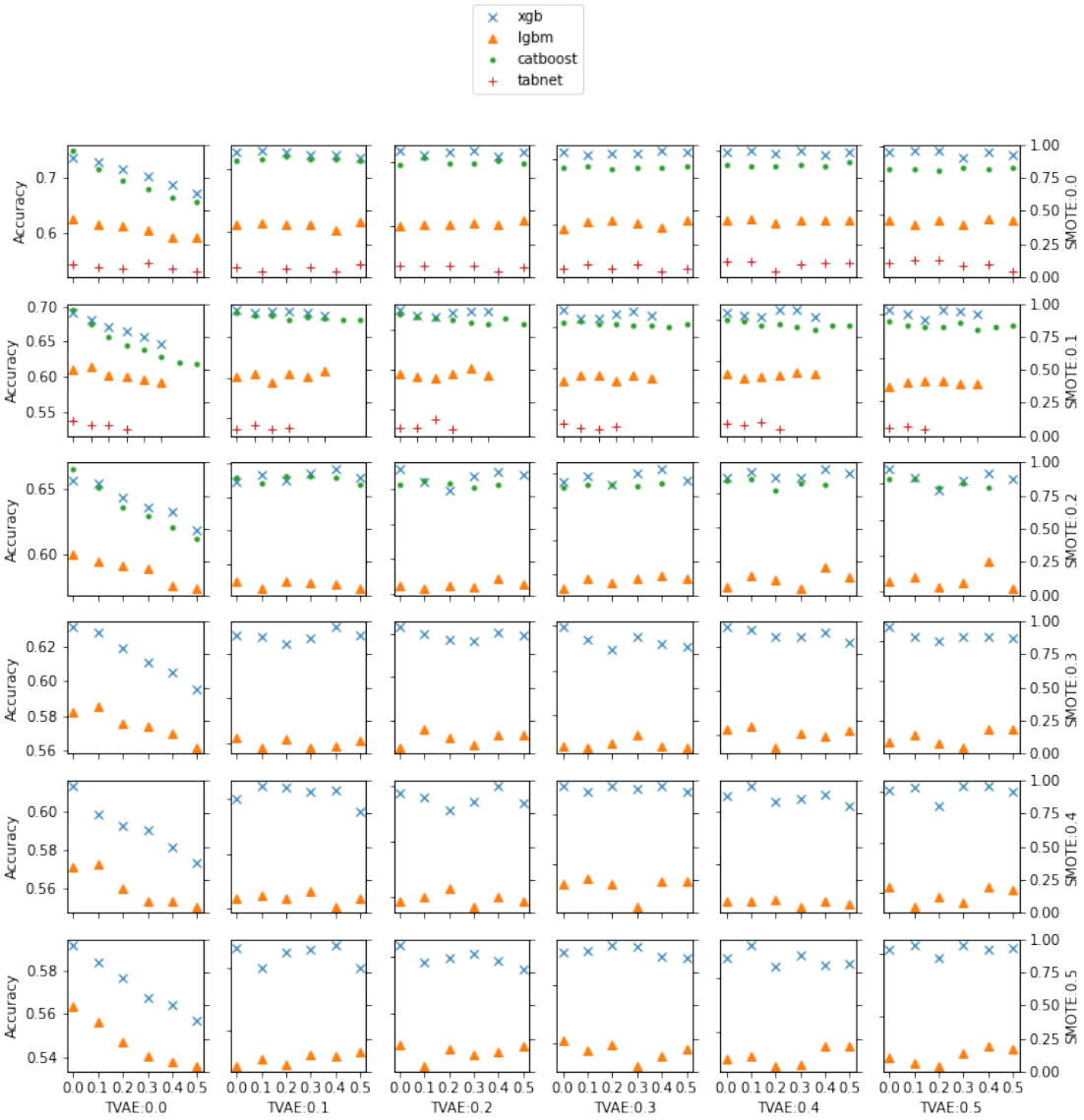


Рис. 40 — Изменения при меняющемся CTGAN, датасет Pokerhand

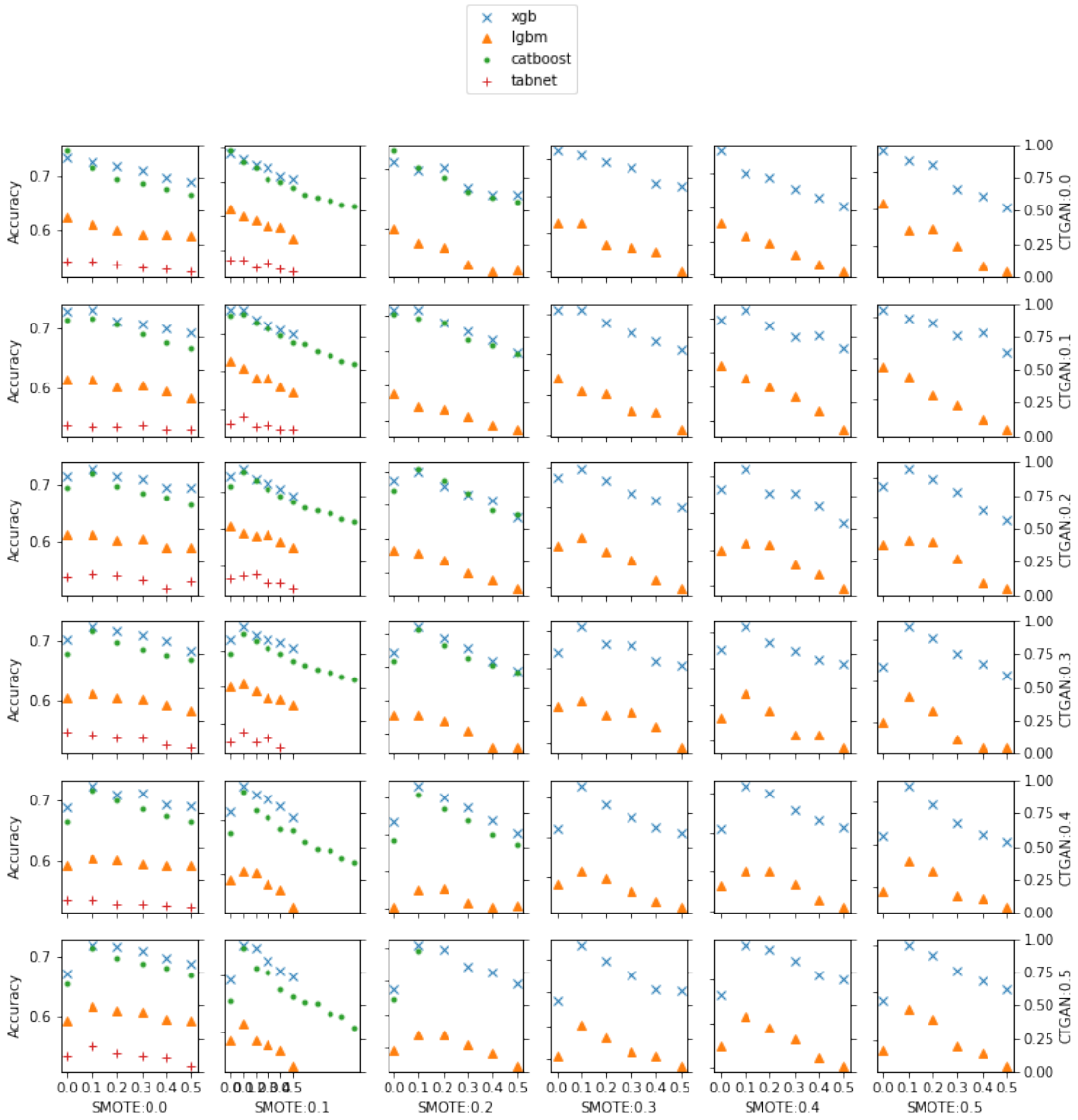


Рис. 41 — Изменения при меняющемся TVAE, датасет Pokerhand

3.5. Анализ полученных данных и выводы

На основе полученных данных можно сделать следующие выводы:

- **Аугментации могут принести выигрыш в качестве.** Как видно из таблицы 5, при определенных условиях каждая модель может получить качество лучше с аугментациями, чем без них.

- **TabNet получает наибольшую пользу из аугментаций**. Из таблицы 6 мы видим, что при одинаковых гиперпараметрах всегда выигрывает либо модель с аугментациями, либо модель с предобучением. Градиентные бустинги же, напротив, чаще всего лишь теряют в качестве из-за подмешивания аугментаций. Вероятнее всего, это связано с тем, что TabNet принципиально устроена значительно сложнее, чем остальные модели, и аугментации могут помочь ей найти недостающие взаимосвязи в данных.

- **Получить качественные аугментации тяжело.** Как показали эксперименты, обучение даже с ускорителями в виде видеокарт занимает большое количество времени с ростом количества объектов и признаков в датасете. В таблице 4 подавляющая часть моделей показала результаты ниже 0.5. Это могло сыграть свою роль при обучении соревнующихся моделей.

- **Простые аугментации не улучшают качество.** В таблице лучших результатов нет подмешивания SMOTE. В целом, по графикам тоже наблюдается ухудшение, причем в случаях всех моделей.

- **Для подмешивания аугментаций, вероятно, есть оптимальная пропорция.** На графиках зачастую можно увидеть рост качества, пик, а затем спад с увеличением числа аугментаций.

- **Предобучение TabNet не всегда улучшает качество.** Предобученная TabNet может терять в качестве, если гиперпараметры подобраны неудачно

- **TabNet все еще очень зависит от гиперпараметров, а также имеет наибольший разброс.** Гиперпараметры в данном исследовании были жестко зафиксированы для каждого датасета, причем во многих случаях - в сторону уменьшения размеров по сравнению с оригинальной статьей [6]. При этом TabNet все еще может показывать хорошее качество, соревнуясь на равных с бустингами.

- **Градиентные бустинги легче и быстрее.** Они занимают меньше памяти, чаще всего не требуют подбора гиперпараметров и обучаются значительно быстрее, чем TabNet.

По итогу данной работы можно сформулировать некоторые эмпирические правила для работы с табличными данными:

- **Выбор модели зависит от ресурсов для обучения и сложности датасета.** В случае простых датасетов побеждают легкие в обучении бустинги, в случае сложно устроенных датасетов и достаточного количества вычислительных мощностей возможен выбор в пользу TabNet.

- **Аугментации оправданы**, если датасет достаточно сложно устроен или есть возможность подобрать гиперпараметры для аугментирующих моделей.

Вероятно, TabNet сможет преодолеть градиентные бустинги, но для этого нужны дополнительные исследования, в том числе - в направлении аугментаций табличных данных и self-supervised подхода, а также их совмещении.

Выводы по главе

В третьей главе были проанализированы полученные результаты экспериментов и сделаны выводы. Кроме того, были предложены подходы к задачам, связанным с табличными данными и дальнейшие направления исследований.

Заключение

В данной работе были исследовано влияние аугментаций на качество обучения TabNet, а также self-supervised подход к обучению TabNet. Было показано, что при определенных условиях аугментации могут улучшить качество предсказаний, а предобучение не всегда приводит к положительному результату. Впоследствии были сделаны выводы из полученных результатов и предложены эмпирические правила для подхода к задачам, связанным с табличными данными. Кроме того, были обозначены направления дальнейшей работы, связанной с данной тематикой.

Список использованных источников

1. *Shwartz-Ziv R.*, *Armon A.* Tabular Data: Deep Learning is Not All You Need // CoRR. — 2021. — Т. abs/2106.03253.
2. Deep residual learning for image recognition / K. He [и др.] // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2016. — С. 770—778.
3. *Iwana B. K.*, *Frinken V.*, *Uchida S.* DTW-NN: A novel neural network for time series recognition using dynamic alignment between inputs and weights // Knowledge-Based Systems. — 2020. — Т. 188. — С. 104971.
4. Deep speech 2: End-to-end speech recognition in english and mandarin / D. Amodei [и др.] // International conference on machine learning. — PMLR. 2016. — С. 173—182.
5. *WilliamVorhies.* Has Deep Learning Made Traditional Machine Learning Irrelevant? — https://www.datasciencecentral.com/ [Электронный ресурс]. — 2016.
6. *Arık S. O.*, *Pfister T.* Tabnet: Attentive interpretable tabular learning //. — 2021.
7. Цитирования TabNet на Google Scholar. — [Online; дата обращения: 22-April-2022].
8. *Shwartz-Ziv R.*, *Armon A.* Tabular data: Deep learning is not all you need // Information Fusion. — 2022. — Т. 81. — С. 84—90.
9. *Katzir L.*, *Elidan G.*, *El-Yaniv R.* Net-DNF: Effective deep modeling of tabular data // International Conference on Learning Representations. — 2020.
10. *Boughorbel S.*, *Jarray F.*, *Kadri A.* Fairness in tabnet model by disentangled representation for the prediction of hospital no-show // arXiv preprint arXiv:2103.04048. — 2021.
11. Deep neural networks and tabular data: A survey / V. Borisov [и др.] // arXiv preprint arXiv:2110.01889. — 2021.
12. *Galton F.* Regression towards mediocrity in hereditary stature. // The Journal of the Anthropological Institute of Great Britain and Ireland. — 1886. — Т. 15. — С. 246—263.
13. *Cramer J. S.* The origins of logistic regression. — 2002.
14. Logistic regression was as good as machine learning for predicting major chronic diseases / S. Nusinovici [и др.] // Journal of Clinical Epidemiology. — 2020. — Т. 122. — С. 56—69.
15. *Quinlan J. R.* Induction of decision trees // Machine learning. — 1986. — Т. 1, № 1. — С. 81—106.
16. *Breiman L.* Bagging predictors // Machine learning. — 1996. — Т. 24, № 2. — С. 123—140.
17. *Friedman J. H.* Greedy function approximation: a gradient boosting machine // Annals of statistics. — 2001. — С. 1189—1232.

18. *Friedman J. H.* Stochastic gradient boosting // Computational statistics & data analysis. — 2002. — Т. 38, № 4. — С. 367—378.

19. *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. — С. 785—794.

20. Higgs Boson Machine Learning Challenge. — Accessed: 2022-02-20. `https://www.kaggle.com/c/higgs-boson`.

21. *Dorogush A. V.*, *Ershov V.*, *Gulin A.* CatBoost: gradient boosting with categorical features support // arXiv preprint arXiv:1810.11363. — 2018.

22. *Kohavi R.*, *Li C.-H.* Oblivious decision trees, graphs, and top-down pruning // IJCAI. — Citeseer. 1995. — С. 1071—1079.

23. Lightgbm: A highly efficient gradient boosting decision tree / G. Ke [и др.] // Advances in neural information processing systems. — 2017. — Т. 30.

24. Репозиторий LightGBM на Github [Электронный ресурс]. — Accessed: 2022-02-20. `https://github.com/Microsoft/LightGBM/blob/master/docs/Features.rst#leaf-wise-best-first-tree-growth`.

25. *Kinnander M.* Predicting profitability of new customers using gradient boosting tree models: Evaluating the predictive capabilities of the XGBoost, LightGBM and CatBoost algorithms. — 2020.

26. *Al Daoud E.* Comparison between XGBoost, LightGBM and CatBoost using a home credit dataset // International Journal of Computer and Information Engineering. — 2019. — Т. 13, № 1. — С. 6—10.

27. *Kilincer I. F.*, *Ertam F.*, *Sengur A.* A comprehensive intrusion detection framework using boosting algorithms // Computers & Electrical Engineering. — 2022. — Т. 100. — С. 107869.

28. DeepTables framework repository [Электронный ресурс]. — `https://github.com/DataCanvasIO/DeepTables`.

29. Wide & deep learning for recommender systems / H.-T. Cheng [и др.] // Proceedings of the 1st workshop on deep learning for recommender systems. — 2016. — С. 7—10.

30. Deep & cross network for ad click predictions / R. Wang [и др.] // Proceedings of the ADKDD'17. — 2017. — С. 1—7.

31. Attentional factorization machines: Learning the weight of feature interactions via attention networks / J. Xiao [и др.] // arXiv preprint arXiv:1708.04617. — 2017.

32. Autoint: Automatic feature interaction learning via self-attentive neural networks / W. Song [и др.] // Proceedings of the 28th ACM International Conference on Information and Knowledge Management. — 2019. — С. 1161—1170.

33. Language modeling with gated convolutional networks / Y. N. Dauphin [и др.] // International conference on machine learning. — PMLR. 2017. — С. 933—941.

34. *Martins A.*, *Astudillo R.* From softmax to sparsemax: A sparse model of attention and multi-label classification // International conference on machine learning. — PMLR. 2016. — С. 1614—1623.

35. *Shavitt I.*, *Segal E.* Regularization learning networks: deep learning for tabular datasets // Advances in Neural Information Processing Systems. — 2018. — Т. 31.

36. *Balestriero R.* Neural decision trees // arXiv preprint arXiv:1702.07360. — 2017.

37. *Yang Y.*, *Morillo I. G.*, *Hospedales T. M.* Deep neural decision trees // arXiv preprint arXiv:1806.06988. — 2018.

38. *Ucar T.*, *Hajiramezanali E.*, *Edwards L.* SubTab: Subsetting Features of Tabular Data for Self-Supervised Representation Learning // Advances in Neural Information Processing Systems. — 2021. — Т. 34.

39. *Lewoniewski W.*, *Węcel K.*, *Abramowicz W.* Quality and Importance of Wikipedia Articles in Different Languages // Information and Software Technologies / под ред. G. Dregvaite, R. Damasevicius. — Cham : Springer International Publishing, 2016. — С. 613—624.

40. Revisiting deep learning models for tabular data / Y. Gorishniy [и др.] // Advances in Neural Information Processing Systems. — 2021. — Т. 34.

41. *Shorten C.*, *Khoshgoftaar T. M.* A survey on image data augmentation for deep learning // Journal of big data. — 2019. — Т. 6, № 1. — С. 1—48.

42. SMOTE: synthetic minority over-sampling technique / N. V. Chawla [и др.] // Journal of artificial intelligence research. — 2002. — Т. 16. — С. 321—357.

43. *Darabi S.*, *Elor Y.* Synthesising multi-modal minority samples for tabular data // arXiv preprint arXiv:2105.08204. — 2021.

44. *Gazzah S.*, *Amara N. E. B.* New oversampling approaches based on polynomial fitting for imbalanced data sets // 2008 the eighth iapr international workshop on document analysis systems. — IEEE. 2008. — С. 677—684.

45. *Xu L.*, *Veeramachaneni K.* Synthesizing tabular data using generative adversarial networks // arXiv preprint arXiv:1811.11264. — 2018.

46. Modeling tabular data using conditional gan / L. Xu [и др.] // Advances in Neural Information Processing Systems. — 2019. — Т. 32.

47. Репозиторий SDV на Github.

48. Репозиторий TabNet на Github [Электронный ресурс]. — Accessed: 2022-02-20. `https://github.com/dreamquark-ai/tabnet`.

49. Wandb [Электронный ресурс]. — `https://wandb.ai/`.

50. *Van Der Walt S.*, *Colbert S. C.*, *Varoquaux G.* The NumPy array: a structure for efficient numerical computation // Computing in science & engineering. — 2011. — Т. 13, № 2. — С. 22—30.

51. pandas: a foundational Python library for data analysis and statistics / W. McKinney [и др.] // Python for high performance and scientific computing. — 2011. — Т. 14, № 9. — С. 1—9.

52. Scikit-learn: Machine learning in Python / F. Pedregosa [и др.] // the Journal of machine Learning research. — 2011. — Т. 12. — С. 2825—2830.

53. Learning to Explain: An Information-Theoretic Perspective on Model Interpretation / J. Chen [и др.]. — 2018.

54. *Vijayakumar S.*, *Schaal S.* Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space // Proceedings of the seventeenth international conference on machine learning (ICML 2000). Т. 1. — Morgan Kaufmann. 2000. — С. 288—293.

55. UCI machine learning repository / D. Dua, C. Graff [и др.]. — 2017.

56. Датасет Stars [Электронный ресурс]. — `https://www.kaggle.com/datasets/brsdincer/star-type-classification`.

57. *Kullback S.*, *Leibler R. A.* On Information and Sufficiency // The Annals of Mathematical Statistics. — 1951. — Т. 22, № 1. — С. 79—86.

58. Optuna: A next-generation hyperparameter optimization framework / T. Akiba [и др.] // Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. — 2019. — С. 2623—2631.