Московский государственный университет имени М. В. Ломоносова
Кафедра МаТИС

КУРСОВАЯ РАБОТА

Классификация датасетов для оптимального выбора алгоритма заполнения пропущенных значений

Dataset Classification for Optimal Imputation Method Selection

Выполнила:
Студент группы 331
Филимонова О. Д.
Научный руководитель:
к.ф.-м.н., доцент
Миронов А. М.

Москва – 2025

Аннотация

В данной курсовой работе рассматривается проблема выбора алгорима заполнения пропущенных значений в датасете. Сравниваются более простой метод KNN и методы, основанные на приближённом поиске ближайших соседей (Approximate Nearest Neighbors, ANN). Выбор оптимального метода заполнения пропусков осуществляется на основе метапризнаков исходного датасета, которые являются входными данными модели гадиентного бустинга (eXtreme Gradient Boosting, XGBoost). Данная модель, в свою очередь, обучается решать задачу классификации: выбор алгоритма заполнения пропущенных значений для данного датасета (HNSW, FAISS или KNN).

1 Введение

1.1 Основная проблема

Пропущенное значение в табличных данных — это отсутствующая информация, которая должна быть указана в корректном виде.

Данная проблема широко распространена при работе с табличными данными. В некоторых случаях значения могут быть случайно искажены в процессе обработки, утрачены при передаче или изначально не внесены в набор данных. Причины пропусков варьируются: это могут быть ошибки при сборе (например, в опросах или измерениях), неточности при ручном вводе или технические сбои при передаче данных.

Наличие пропущенных значений негативно сказывается на качестве данных и может ухудшить работу алгоритмов машинного обучения, если их не обработать должным образом. Существует несколько мощных алгоритмов, способных работать с пропущенными значениями, но их эффективность снижается на больших объемах данных вследствие большой вычислительной сложности. Ярким примером является алгоритм k ближайших соседей. Поэтому важной задачей является выбор оптимального алгоритма заполнения пропущенных значений в больших (промышленных) датасетах.

В данной работе рассматривается именно проблема выбора подходящего алгоритма заполнения для датасета с пропущенными значениями.

1.2 Существующие подходы

Одним из самых популярных методов восстановления пропущенных значений является KNN (k Nearest Neighbors) Imputer. В методе KNN пропущенное значение признака заменяется средним значением этого же признака у k ближайших объектов в пространстве признаков, что минимизирует искажение исходного распределения данных.

В то же время существует семейство алгоритмов приближённого поиска ближайших соседей (ANN — Approximate Nearest Neighbors), разработанных для ускорения классического метода KNN. Они работают за счёт разбиения пространства поиска на отдельные области, после чего поиск ближайших соседей для каждого элемента выполняется только в соответствующей части этого пространства. В настоящее время существует множество различных методов и библиотек для приближённого поиска ближайших соседей (ANN).

В данной работе сравниваются новые методы заполнения пропусков на основе ANN и классический KNN.

1.3 Предложенный подход

Для выбора алгоритма заполнения пропущенных значений построим классификаторы, на вход которым будут подаваться статистические характеристики признаков датасета (назовем эти характиристики метапризнаками).

2 Мотивация исследования

Проблема пропущенных значений приводит к множеству негативных последствий при использовании данных для обучения моделей машинного обучения. Некоторые мощные алгоритмы способны работать с пропусками, но при этом теряют в эффективности. Другая сложность заключается

в отсутствии надежного метода выбора подходящего алгоритма заполнения пропущенных значений, тогда как наиболее эффективные из них требуют слишком много времени на обработку данных. Именно поэтому научное сообщество уделяет большое внимание задаче предсказания и заполнения пропущенных значений в данных. Для оценки новых подходов к заполнению необходимо сравнить их эффективность с предшествующими более примитивными алгоритмами. Данная проблема рассматривается в этой работе (сравнение новых методов на основе алгоритмов HNSW, FAISS и KNN — см. далее в работе).

3 Основные понятия и определения

3.1 Типы пропущенных значений

- **Пропущенные полностью случайно (MCAR — Missing Completely At Random)**

  Пропуски возникают абсолютно случайно, без связи со значениями признаков. Вероятность пропуска в любой строке данных не зависит ни от самого отсутствующего значения, ни от других переменных.

  Пример: технический сбой при записи данных, случайная потеря части информации.

- **Пропущенные случайно (MAR — Missing At Random)**

  Вероятность пропуска зависит от других наблюдаемых переменных, но не от самого пропущенного значения.

  Пример: в медицинских данных пациенты старшего возраста реже указывают вес — пропуск связан с возрастом, но не с конкретным значением веса.

- **Пропущенные не случайно (MNAR — Missing Not At Random)**

  Вероятность пропуска напрямую зависит от отсутствующего значения. Если бы данные были известны, можно было бы предсказать, с какой вероятностью они пропали.

  Пример: уклонение от декларирования высоких доходов — чем больше доход, тем выше вероятность его сокрытия.

3.2 Алгоритмы импутации

Приближенный поиск ближайших соседей (Approximate Nearest Neighbors, ANN) — это семейство алгоритмов поиска ближайших соседей, которое делит пространство поиска на несколько частей, назначает каждому объекту одну из этих частей и использует это разбиение для поиска ближайших соседей рассматриваемого элемента только в соответствующей ему части пространства.

Конкретный способ разбиения пространства определяет метод ANN. Такой способ разделения обычно называют **индексированием**. Благодаря этому удается избежать множества лишних вычислений: система не измеряет расстояния между запросным вектором и большинством векторов в наборе данных, так как они относятся к разным подпространствам. Однако возникает риск ошибки, поскольку близкие объекты иногда могут попасть в разные подпространства из-за несовершенства алгоритма.

3.2.1 Выбор методов

Все методы индексирования делятся на три категории:

- **Точные (Flat) индексы.** Точные (Flat) индексы представляют собой наиболее простой и надежный подход, основанный на полном переборе всех возможных расстояний между векторами. В отличие от приближенных методов, такие индексы гарантируют стопроцентную точность результатов, поскольку не используют никаких эвристик или оптимизаций вычислений. Однако эта точность достигается ценой значительных вычислительных затрат, что делает Flat-индексы практичными только для относительно небольших наборов данных (обычно менее 10^4 векторов). Типичным примером реализации этого подхода является FAISS Flat Index, который часто используется в качестве базового уровня для сравнения эффективности других методов.

- **Графовые индексы.** Графовые индексы, такие как популярный метод HNSW (Hierarchical Navigable Small World), используют принципы теории графов для организации данных. В этих структурах каждый вектор представлен вершиной графа, а ребра соединяют наиболее похожие векторы, образуя так называемые "малые миры". При поиске алгоритм навигации по такому графу позволяет быстро находить ближайшие соседи, начиная от случайных точек и постепенно уточняя область поиска. Главными преимуществами графовых методов являются высокая скорость работы и хорошее качество поиска, особенно для данных средней размерности. Однако эти преимущества сопровождаются значительными затратами памяти и относительно долгим процессом построения индекса, что может быть критично для очень больших наборов данных.

- **IVF-индексы.** IVF-индексы (Inverted File Index) основаны на предварительной кластеризации пространства признаков, обычно с помощью алгоритма K-Means. После разбиения данных на кластеры создается обратный индекс, который для каждого кластера хранит список принадлежащих ему векторов. При поиске система сначала определяет несколько ближайших кластеров, а затем ищет соседей только среди их элементов. Такой подход обеспечивает исключительную скорость работы с очень большими наборами данных (более 10^6 векторов) и позволяет эффективно использовать память. Однако точность результатов в значительной степени зависит от качества первоначальной кластеризации и требует тщательного подбора гиперпараметров. В библиотеке FAISS этот метод представлен индексами IVFFlat.

В работе сравниваются методы из каждой категории:

- **KNN** (точный поиск) представляет Flat-индексы,
- **HNSW** — наиболее распространённый графовый метод,
- **IVF Flat из FAISS** — простейший и самый популярный метод среди IVF-индексов.

3.2.2 Сравнение с новыми методами

В этой работе предлагается сравнить классический метод KNN с новыми методами заполнения пропущенных значений, ценность которых в уменьшении времени заполнения.

Хотя алгоритм KNN имеет линейную вычислительную сложность, необходимость вычисления полного набора расстояний для каждого объекта в наборе данных приводит к значительным временным затратам из-за большого объема реальных промышленных данных.

Для снижения вычислительной сложности предлагается заменить стандартный KNN на приближенные методы (ANN):

- **Поиск на основе HNSW-индекса**,
- **Поиск на основе FAISS Flat Index**.

Это позволяет значительно ускорить процесс импутации. Задача данной работы - оценить, уменьшается ли точность импутации при использовании методов ANN по сравнению в классическим KNN.

3.3 Метапризнаки

Метапризнаки: статистические характеристики датасета. В данной работе используются коэффициент эксцесса, среднее значение, количество признаков объекта (число столбцов датасета), число объектов (число строк датасета), тип пропусков (MAR, MCAR или MNAR) и процент пропусков.

3.4 Модель градиентного бустинга XGBoost

Градиентный бустинг (Gradient Boosting) — это метод машинного обучения, основанный на последовательном построении ансамбля слабых моделей (обычно деревьев решений), где каждая следующая модель компенсирует ошибки предыдущих.

3.4.1 Постановка задачи

Пусть задана обучающая выборка $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, где $\mathbf{x}_i \in \mathbb{R}^m$ — признаки, $y_i \in \mathbb{R}$ — целевая переменная. Предсказание модели на шаге t обозначим как $F_t(\mathbf{x})$.

Алгоритм градиентного бустинга минимизирует функцию потерь $\mathcal{L}(y, F(\mathbf{x}))$:

$$F^* = \arg\min_F \mathbb{E}_{(\mathbf{x},y)} \mathcal{L}(y, F(\mathbf{x})) \tag{1}$$

На каждом шаге t строится новая модель $h_t(\mathbf{x})$, которая аппроксимирует антиградиент функции потерь:

$$h_t(\mathbf{x}) \approx - \left. \frac{\partial \mathcal{L}(y, F(\mathbf{x}))}{\partial F(\mathbf{x})} \right|_{F=F_{t-1}} \tag{2}$$

Обновление предсказаний происходит по правилу:

$$F_t(\mathbf{x}) = F_{t-1}(\mathbf{x}) + \eta \cdot h_t(\mathbf{x}) \tag{3}$$

где η — темп обучения (learning rate).

3.4.2 Алгоритм градиентного бустинга

1. Инициализация: $F_0(\mathbf{x}) = \arg\min_\gamma \sum_{i=1}^n \mathcal{L}(y_i, \gamma)$
2. Для $t = 1$ до T:
   - Вычисление псевдо-остатков:
   $$r_{it} = - \left. \frac{\partial \mathcal{L}(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)} \right|_{F=F_{t-1}}, \quad i = 1, \ldots, n$$
   - Построение модели $h_t(\mathbf{x})$ на данных $\{(\mathbf{x}_i, r_{it})\}_{i=1}^n$
   - Поиск оптимального шага:
   $$\rho_t = \arg\min_\rho \sum_{i=1}^n \mathcal{L}(y_i, F_{t-1}(\mathbf{x}_i) + \rho h_t(\mathbf{x}_i))$$
   - Обновление модели:
   $$F_t(\mathbf{x}) = F_{t-1}(\mathbf{x}) + \eta \cdot \rho_t h_t(\mathbf{x})$$
3. Возврат итоговой модели $F_T(\mathbf{x})$

3.5 Подбор гиперпараметров модели с помощью библиотеки Optuna

Optuna — это фреймворк для автоматического подбора гиперпараметров моделей машинного обучения.

3.5.1 Постановка задачи

Пусть задана модель $f(\mathbf{x}, \theta)$, где:

- $\mathbf{x}$ — входные данные
- $\theta \in \Theta$ — гиперпараметры из пространства поиска Θ

Цель найти оптимальные θ^*, минимизирующие функцию потерь $\mathcal{L}$:

$$\theta^* = \arg\min_{\theta \in \Theta} \mathcal{L}(f(\mathbf{x}, \theta), y) \tag{4}$$

3.6 Основные алгоритмы в Optuna

3.6.1 Древовидный структурированный парзеновский оцениватель (TPE)

Метод основан на байесовской оптимизации:

$$p(\theta|\mathcal{L}) = \begin{cases} l(\theta) & \text{если } \mathcal{L} < \mathcal{L}^* \\ g(\theta) & \text{если } \mathcal{L} \geq \mathcal{L}^* \end{cases} \tag{5}$$

где $l(\theta)$ и $g(\theta)$ — ядерные оценки распределений для "хороших"и "плохих"параметров соответственно.

3.6.2 Критерий ожидаемого улучшения (EI)

Используется для выбора следующей точки:

$$EI(\theta) = \mathbb{E}[\max(\mathcal{L}^* - \mathcal{L}(\theta), 0)] \tag{6}$$

4 Результаты

4.1 Создание обучающей выборки

Для обучения классификаторов использовались результаты экспериментов на датасетах из открытого источника OpenML.

Каждый эксперимент проводился следующим образом:

1. Выбирался датасет без пропущенных значений (NaN), с количеством строк от 1000 до 99000, долей численных призаков не меньше 70%. Обозначим этот датасет D_0.

2. Некоторые элементы D_0 заполнялись пропущенными значениями в соответствии с выбранным типом пропусков и процентом пропусков. Обозначим полученный датасет D_1.

3. К датасету с пропущенными значениями D_1 применялись алгоритмы импутации: KNN, FAISS, HNSW.

4. Полученный алгоритмом заполненный датасет сравнивается с изначальным датасетом D_0 по метрикам accuracy, precision, recall, f1 для классификации и MSE, RMSE, R2 для регресии.

5. Эксперименту ставится в соотвествие метка y, т. ч.

$$y = \begin{cases} 1, & \text{результат HNSW/FAISS лучше результата KNN} \\ 0, & \text{иначе} \end{cases}$$

Далее формировался метадатасет . Каждому датасету D_1 ставилась в стоответствие строка S статистик этого датасета следующим образом:

1. Для каждого признака датасета M рассчитывались статистики: коэффициент эксцесса, скошенность, медиана, стандартное отклонение. В датасет M добавлялись максимальное значение, минимальное и медиана данной каждой из этих статистик. Таким образом, в S добавились 12 признаков.

2. Также в S включались такие признаки как: количество объектов и признаков D_1, тип пропущенных значений (MAR, MNAR, MCAR), процент пропущенных значений от общего количества объектов.

Каждой строке S ставится в соответствие число k - гиперпараметр алгоритма импутации (k ближайших соседей). $k \in \{15, 100, 500, 1000\}$.

Таким образом, metadataset содержит 17 признаков и 1004 объекта (результаты 1004 эксперимента).

4.2 Построение и обучение моделей классификации

Метадатасет был разделен на тестовую и тренировочную выборку в соотношении 0.2. Были обучены 3 классификатора:

1. $model_1$ и $model_2$ - бинарные классификаторы с целевыми перменными
   $y_1 = \begin{cases} 1, & \text{результат HNSW лучше результата KNN} \\ 0, & \text{иначе} \end{cases}$ и $y_2 = \begin{cases} 1, & \text{результат FAISS лучше результата KNN} \\ 0, & \text{иначе} \end{cases}$
   соответственно.
2. $model_3$ - классификация на 3 класса с целевой переменной $y_3 = \begin{cases} 0, & \text{результат KNN является лучшим} \\ 1, & \text{результат HNSW является лучшим} \\ 2, & \text{результат FAISS является лучшим} \end{cases}$

В качестве классификаторов использовались модели градиентного бустинга XGBoost. Гиперпараметры подбирались с помощью техники Optuna со следующими аргументами:

1. Темп обучения (learning rate): [0.01, 0.1]
2. Максимальная глубина решающего дерева: $\{3, 4, 5\}$
3. Количество решающих деревьев: $\{10, ..., 100\}$

4.3 Оценка работы алгоритмов

Удалось добиться следующих значений метрик: $auc = 0.74$ для $model_1$, $auc = 0.63$ для $model_2$, $auc = 0.67$ для $model_3$ при гиперпараметрах, указанных в Таблице 1.

Таблица 1: Подобранные с помощью Optuna гиперпараметры моделей

Модель	**Лучший результат auc**	**Гиперпараметры**
$model_1$	0.737	n_estimators: 10 max_depth: 4 learning_rate: 0.059
$model_2$	0.633	n_estimators: 16 max_depth: 5 learning_rate: 0.018
$model_3$	0.674	n_estimators: 84 max_depth: 3 learning_rate: 0.034

Далее представлены ROC-кривые (Receiver Operating Characteristic curves), демонстрирующие качество бинарных классификационных моделей (рис. (1), (2)).

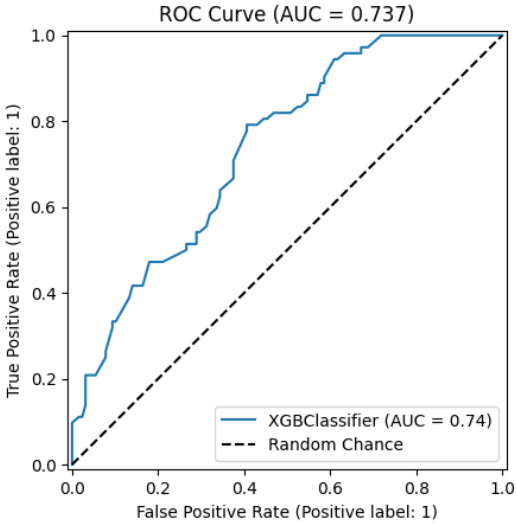

Рис.1 ROC-кривая $model_1$

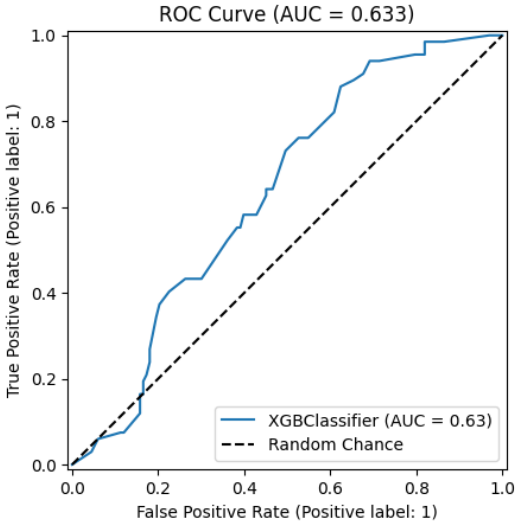

Рис.2 ROC-кривая $model_2$

Также были проанализированы важности метапризнаков (feature importances) при обучении алгоритмов градиентного бустинга (рис. (3),(4),(5)). Исходя из графиков, решающим признаком является количество ближайших соседей (k list), используемых в алгоритмах заполненрия пропущенных значений. Остальные обозначения: kurtosis — эксцесс, std — стандартное отклонение, median — медиана, percent — процент пропущенных значений, sample count — число объектов датасета.

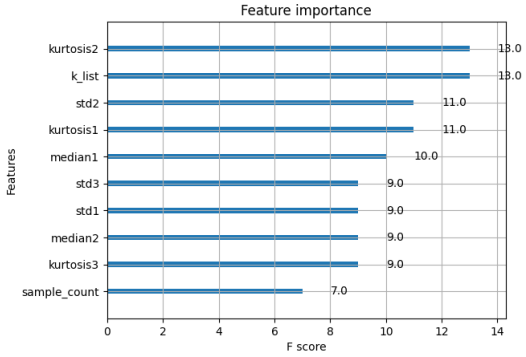

Рис. 3 Важности метапризнаков для $model_1$

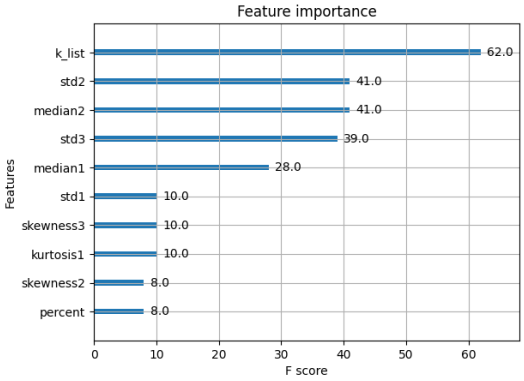

Рис. 4 Важности метапризнаков для $model_2$

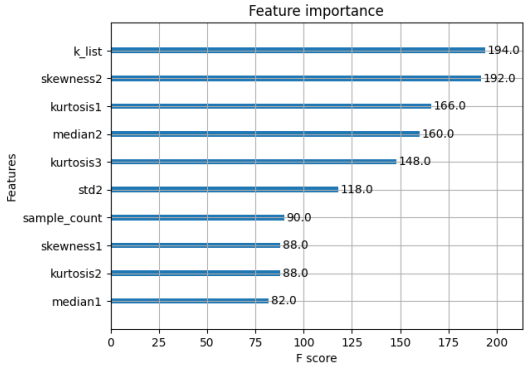

Рис. 5 Важности метапризнаков для $model_3$

5 Заключение

- Собран и предобработан набор датасетов из открытого источника OpenML.
- Проведены 1004 эксперимента с различными настройками методов HNSW и FAISS.
- Сформирован метадатасет, содержащий метапризнаки исходных данных (коэффициент эксцесса, среднее, асимметрия и другие характеристики).
- На полученном метадатасете обучена модель XGBoost, предсказывающая, какой метод восстановления (KNN, HNSW, FAISS) обеспечит наилучшее качество для конкретного датасета.

Список литературы

[1] A. C. Acock. What to do about missing values. In *Handbook of Psychology, Second Edition*, pages 27–50. American Psychological Association, 2012.

[2] E. Acuña and C. Rodriguez. The treatment of missing values and its effect on classifier accuracy. In *Classification, Clustering, and Data Mining Applications*, pages 639–647. Springer, 2004.

[3] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631. ACM, 2019.

[4] P. Brazdil, J. N. van Rijn, C. Soares, and J. Vanschoren. *Metalearning: Applications to Automated Machine Learning and Data Mining*. Cognitive Technologies. Springer, 2009.

[5] OpenML. Openml: An open science platform for machine learning. https://www.openml.org.