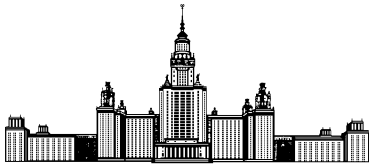
Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики

Кафедра Математических Методов Прогнозирования

КУРСОВАЯ РАБОТА СТУДЕНТА 317 ГРУППЫ

«Сравнительный анализ методов быстрого поиска ближайших соседей»

Выполнил:

студент 3 курса 317 группы

Федоров Илья Сергеевич

Научный руководитель:

д.ф-м.н., профессор

Дьяконов Александр Геннадьевич

Заведующий кафедрой

Математических Методов

Прогнозирования, академик РАН _____ Ю. И. Журавлёв

К защите допускаю

«_____» _____ 2020 г.

К защите рекомендую

«_____» _____ 2020 г.

Москва, 2020

Содержание

Аннотация

Todo

1 Введение

Одним из наиболее простых и естественных методов машинного обучения является метод ближайшего соседа. Имея набор данных, представленных в виде точек в некотором многомерном пространстве, целевая величина (будь то класс или вещественное число) прогнозируется по значениям отклика на k ближайших к запросу точках из исходного набора данных. В то время как существует множество различных подходов к усреднению данных k значений, наиболее вычислительно затратной частью алгоритмов подобного типа является именно поиск ближайших соседей. Действительно, в современных задачах объемы данных достигают колоссальных размеров, что делает алгоритмы, основанные на полном переборе, неэффективными. Задача поиска ближайших к запросу точек в некотором наборе данных встречается не только в задачах прогнозирования. Примерами приложений также могут служить задачи поиска дубликатов в больших объемах данных (или «почти» дубликатов), поиска похожих изображений и текстов. Целью данной работы является обзор современных подходов к решению задачи поиска ближайших соседей и её вариаций, а также сравнительный анализ эффективности тех или иных методов её решения в зависимости от особенностей пространства, в котором расположены данных. В исследовании представлены как классические подходы, основанные на формировании некоторых дополнительных структур данных, так и наиболее современные «приближенные» методы.

1.1 Определения и обозначения

Формализуем постановку задачи. Основным объектом нашего изучения будет пространство признаков вместе с функцией расстояния $\mathbb{X} = (\mathbb{R}^n, d)$. Важно заметить, что функция d в приложениях довольно часто может не удовлетворять формальному определению метрики, однако даже в этом случае в данной работе подобные функции, допуская некоторую вольность, будут называться метриками. К примеру, широко используемое в анализе текстов косинусное расстояние $d(x, y) = 1 - \frac{\langle x,y \rangle}{\|x\|\|y\|}$ не удовлетворяет неравенству треугольника. В данной работе в большинстве случаев будет использоваться евклидовая метрика и описанное выше косинусное расстояние.

Будем обозначать $X \in \mathbb{R}^{l \times n}$ матрицу для выборки точек из $\mathbb{X}$, где строки соответствуют объектам, а столбцы признаками. Формально задача поиска k ближайших соседей ставится следующим образом: имея множество объектов X из пространства $\mathbb{X}$ и запрос $q \in \mathbb{X}$, нужно найти в X k ближайших к q точек по метрике d. Более подробно, если посчитать расстояния между q и всеми объектами из X, а потом расположить их в отсортированном порядке

$$d(x_{i_1}, q) \leq d(x_{i_2}, q) \leq \ldots \leq d(x_{i_l}, q),$$

то алгоритм должен выдать k объектов с минимальными расстояниями: $x_{i_1}, x_{i_2}, \ldots, x_{i_k}$.

Как мы увидим далее, большинство современных методов быстрого поиска ближайших соседей на самом деле решают описанную выше задачу с некоторыми ослаблениями, что в сущности приводит к задаче «приближенного» поиска ближайших соседей. Эта задача не имеет общепризнанной формальной постановки, разные авторы могут по-разному понимать её. К примеру, довольно распространенной постановкой задачи является следующая формулировка: имея набор точек X из $\mathbb{X}$, запрос $q \in \mathbb{X}$, а также параметр $c \geq 1$, если существует точка $x \in X$, такая что $d(x, q) \leq r$, алгоритм должен вернуть точку $x^* \in X$, такую что $d(x^*, q) \leq cr$. В дальнейшем при описании конкретных методов, мы будем уточнять, какую именно задачу приближенного поиска ближайших соседей они решают.

1.2 Обзор литературы

todo, напишу после основной части

2 Обзор существующих методов поиска ближайших соседей

2.1 Прямое вычисление матрицы попарных расстояний

Самым простым и распространенным способом поиска ближайших соседей является прямой перебор. Имея набор данных X и запрос q, мы вычисляем расстояния между каждым объектом $x \in X$ и q. После этого полученные расстояния сорти-

руются, и алгоритм выдает k объектов из X, имеющих наименьшие расстояния до q.

Оценим вычислительную сложность такого алгоритма. Заметим, что для вычисления евклидового или косинусного расстояния между двумя объектами размерности n нужно совершить порядка n операций. Отсюда получаем, что сложность вычисления расстояний $\mathcal{O}(nl)$ (в наших обозначениях l – число объектов). Далее требуется отсортировать полученный массив, что займет $\mathcal{O}(l \log(l))$ операций. Наконец, останется совершить $\mathcal{O}(k)$ операций для выдачи результата. Учитывая, что $k \leq l$, получим итоговую сложность: $\mathcal{O}(nl + l \log(l))$. Однако на практике данную сложность можно улучшить. Дело в том, что обычно $k \ll l$, а значит бóльшая часть информации из отсортированного массива нам не нужна. Поэтому вместо сортировки можно использовать более эффективные алгоритмы для поиска k наименьших чисел в массиве. Например, это можно сделать с помощью структуры данных под названием куча. Имея массив из l элементов, можно построить кучу за $\mathcal{O}(l)$, а далее вытащить из неё k минимальных элементов за $\mathcal{O}(\log(l))$ каждый. Получаем сложность поиска k минимальных чисел в массиве $\mathcal{O}(l + k \log(l))$. Учитывая, что $k \ll l$, вторым слагаемым можно пренебречь, и оценить итоговую сложность всего алгоритма в $\mathcal{O}(nl)$.

Данный алгоритм вполне эффективен и применим, если объем данных и запросов не слишком велик. К примеру, в соревнованиях по машинному обучению довольно часто встречаются наборы данных размером порядка $10^5 - 10^6$ размерности около 100. Имея обучающую выборку размером 10^6 размерности 100, а также тестовую выборку размером 2×10^6, алгоритм прямого перебора будет работать около 24 часов на 8-ядерном процессоре. Это вполне приемлимо, если требуется решить задачу для конкретной тестовой выборки. Однако данный алгоритм обладает рядом существенных недостатков. Во-первых, если поиск ближайших соседей проводится для решения какой-то задачи машинного обучения, то все вычисления проводятся непосредственно в момент предсказания целевой величины. Поскольку алгоритм никак не обучается, его ценность с точки зрения производительности существенно падает, так как при каждом предсказании производится набор вычислений, сопоставимый по объему с обучением какого-то другого алгоритма, который, обучившись лишь единожды, может очень быстро выдавать ответы (например, линейная или логистиче-

ская регрессия). Во-вторых, если данных становится действительно много (скажем, больше 10^10), то для хоть сколько-то большой тестовой выборки уже требуется колоссальное количество времени для вычислений. Современные компьютеры могут выполнять примерно 10^8 операций в секунду, поэтому для обучающей выборки размером 10^{10} (вполне реальная цифра для больших компаний), тестовой выборки размером 10^3, размерности пространства 10 такое вычисление займет $\frac{10^{10} \times 10^3 \times 10}{10^8}$ секунд $\approx$ 278 часов $\approx$ 12 дней. Безусловно, эти цифры можно сократить, используя специализированные архитектуры компьютеров и многопоточность, однако представленные два недостатка данного алгоритма в совокупности ставят под сомнение его использование в промышленных масштабах.

2.2 Структуры данных

3 Вычислительные эксперименты

Цель данного раздела: продемонстрировать, что предложенная теория работает на практике; показать границы её применимости; рассказать о новых экспериментальных фактах.

Чисто теоретические работы могут вообще не содержать раздела экспериментов (не работает, ну и не надо — зато теория красивая). Кстати, теоретики имеют право не догадываться, где, кому и когда их теории пригодятся.

3.1 Исходные данные и условия эксперимента

Описывается прикладная задача, параметры анализируемых данных (например, сколько объектов, сколько признаков, каких они типов), параметры эксперимента (например, как производился скользящий контроль).

3.2 Результаты эксперимента

Результаты экспериментов представляются в виде таблиц и графиков. Объясняется точный смысл всех обозначений на графиках, строк и столбцов в таблицах.

3.3 Обсуждение и выводы

Приводятся выводы: в какой степени результаты экспериментов согласуются с теорией? Достигнут ли желаемый результат? Обнаружены ли какие-либо факты, не нашедшие объяснения, и которые нельзя списать на «грязный» эксперимент?

Обсуждаются основные отличия предложенных методов от известных ранее. В чем их преимущества? Каковы границы их применимости? Какие проблемы удалось решить, а какие остались открытыми? Какие возникли новые постановки задач?

4 Заключение

В квалификационных работах последний раздел нужен для того, чтобы конспективно перечислить основные результаты, полученные лично автором.

Результатами, в частности, являются:

- Предложен новый подход к...
- Разработан новый метод..., позволяющий...
- Доказан ряд теорем, подтверждающих (опровергающих), что...
- Проведены вычислительные эксперименты..., которые подтвердили / опровергли / привели к новым постановкам задач.

Цель данного раздела: доказать квалификацию автора. Даже беглого взгляда на заключение должно быть достаточно, чтобы стало ясно: автору удалось решить актуальную, трудную, ранее не решённую задачу, предложенные автором решения обоснованы и проверены.

Иногда в Заключении приводится список направлений дальнейших исследований.

Список литературы необходим в любой научной публикации. В дипломной работе он обязателен. Дурным тоном считается: ссылаться на работы только одного-двух авторов (например, себя или шефа); ссылаться на слишком малое число работ; ссылаться только на очень старые работы; ссылаться на работы, которых автор ни разу не видел; ссылаться на работы, которые не упоминаются в тексте или которые не имеют отношения к данному тексту.