Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Тверской государственный университет»

СТУДЕНЧЕСКАЯ КОНФЕРЕНЦИЯ ФАКУЛЬТЕТА ПМиК. СБОРНИК ТРУДОВ

Тверь
18–29 апреля 2022 г.

Под редакцией Б. Н. Карлова

Тверь 2022

УДК 004 + 330 + 519
ББК 22.18 + 65.05я43
С88

Тверской государственный университет
Факультет прикладной математики и кибернетики

С88 Студенческая конференция факультета ПМиК. Сборник трудов. Тверь, 18–29 апреля 2022 г. / под ред. Б. Н. Карлова. — Тверь : ТвГУ, 2022. — 200 с.

Сборник содержит научные работы, представленные на студенческой конференции, проводимой факультетом прикладной математики и кибернетики Тверского государственного университета. Сборник предназначен для научных работников, аспирантов и студентов старших курсов.

УДК 004 + 330 + 519
ББК 22.18 + 65.05я43

Содержание

УДК 004.02

Решение транспортно-логистических задач в товарно-распределительной сети

Акендинов И. А.

Тверской государственный университет

Аннотация. В статье рассматривается задача оптимизации затрат на доставку товаров в интернет-магазине путем оптимизации количества арендованных автомобилей и построения для них маршрутов минимальной протяженности. В основе решения лежит метод ветвей и границ для задачи коммивояжера, распространенный на случай нескольких коммивояжеров. Описывается модифицированный метод ветвей и границ и рассматривается демонстрационный пример работы алгоритма.

Введение

За последние несколько лет электронная коммерция стала неотъемлемой частью глобальной системы розничной торговли. Поскольку доступ к Интернету и его распространение во всем мире быстро растут, число онлайн-покупателей продолжает увеличиваться с каждым годом. Благодаря преимуществам интернет-торговли многие потребители все чаще делают выбор в пользу онлайн-шопинга, отказываясь от посещения традиционных магазинов. В условиях возрастающей доли онлайн-покупок магазины вынуждены большее внимание уделять оптимизации процессов доставки, минимизации затрат на аренду транспортных средств и топливо. Также важнейшим критерием оптимизации является время доставки товара со склада конечному покупателю, так как от этого зависит репутация компании на рынке, в условиях широкой конкуренции лояльность потребителей имеет огромное значение.

В рамках исследования решается задача, которая формулируется следующим образом: имеется некоторая компания, располагающая сетью складов и выполняющая доставку интернет-заказов конечным потребителям на арендованных автомобилях. Имеется список адресов складов и пул данных об адресах доставки заказов за текущий

день. Необходимо определить оптимальное количество автомобилей, которые нужно арендовать для организации доставки, и построить для каждого из них оптимальный маршрут.

1. Задача коммивояжера и методы ее решения

Ключевая задача, которую вынуждены решать транспортные логисты — это создание оптимальных маршрутов перевозок. Такая классическая задача из теории графов носит название — задача коммивояжера.

Задача коммивояжера (TSP) — это алгоритмическая задача, заключающаяся в поиске оптимального маршрута между набором точек, каждую из которых необходимо посетить ровно один раз. Критериями оптимизации в данной задаче может быть длина маршрута, время движения, стоимость перевозки, прочие параметры и их комбинации.

В ситуациях, когда стоимость доставки может приближаться к стоимости доставляемого товара, важным является снижение затрат на аренду и топливо для транспортных средств. Также важнейшим критерием оптимизации является время доставки товара со склада конечному покупателю, так как от этого зависит репутация компании на рынке.

Задача коммивояжера изучается десятилетиями, и было предложено несколько решений. Самое простое решение — перепробовать все возможности, но это и самый трудоемкий и дорогой метод. Во многих решениях используется эвристика, которая обеспечивает вероятность результатов. Однако результаты приблизительны и не всегда оптимальны.

Метод ветвей и границ — один из методов дискретной оптимизации для решения задачи коммивояжера, являющийся развитием метода полного перебора, но отличающийся от него отсевом в процессе вычисления подмножеств неэффективных решений. Вначале данного алгоритма множество всех решений развивается на подмножества, и вычисляются их оценки. На каждом шаге выбирается подмножество с наименьшей оценкой, а остальные отсеиваются. В результате остается одна ветвь с потенциально минимальным расстоянием.

2. Модификация метода ветвей и границ

В текущей постановке задачи есть некоторые особенности, которые не позволяют решить ее стандартными методами. Во-первых, это наличие нескольких складов. Во-вторых, возможность использования нескольких коммивояжеров, так же нужно учесть, что матрица расстояний в данной задаче не симметрична, так как дорога в прямом и обратном направлении может иметь различную протяженность (особенно это актуально в городских условиях).

Доработаем алгоритм метода ветвей и границ, чтобы он рассчитывал пути сразу для заданного количества коммивояжеров. Для этого необходимо ввести ограничение, что маршруты каждого коммивояжера не должны пересекаться, то есть один коммивояжер не должен посещать точку, которая уже была посещена любым другим коммивояжером. В условиях нашей задачи это означает, что к одному и тому же клиенту не должно приезжать более одной машины.

Модифицированный алгоритм метода ветвей и границ для числа коммивояжеров n, отличного от единицы, выглядит следующим образом:

1) Составляется матрица расстояний между пунктами, куда необходимо доставить заказы (R_{ij}). Элементы на главной диагонали, то есть при $i = j$, не заполняются и не участвуют в дальнейших расчетах.

2) Из пункта, который был выбран в качестве начального, строится n маршрутов в следующие пункты, которые еще не были посещены ни одним из коммивояжеров.

3) В матрице для каждого автомобиля находятся минимальные значения расстояний по каждой строке (m_i) и выписываются в отдельный столбец.

4) Для всех матриц из значений элементов каждой строки вычитается минимум, соответствующий этой строке ($R_{ij} = R_{ij} - m_i$).

5) В матрице для каждого автомобиля находятся минимальные значения расстояний по каждому столбцу (m_j) и выписываются в отдельную строку.

6) Для всех матриц из значений элементов каждого столбца вычитается минимум, соответствующий этому столбцу ($R_{ij} = R_{ij} - m_j$).

7) По матрице для каждого автомобиля рассчитывается нижняя граница путем суммирования вычисленных ранее минимумов по строкам и столбцам $H_0 = \sum m_i + \sum m_j$ и строится новый граф решения, в который вносится корневая вершина.

8) Для всех элементов матрицы, в которых записаны нулевые значения, вычисляются оценки (e_{ij}) путем суммирования вычисленных ранее минимумов по соответствующей строке и столбцу.

9) Для каждой матрицы находится элемент, которому соответствует наибольшая оценка и совершается переход к получению пары вариантов решения задачи.

10) В каждой матрице исключаются строка и столбец, относящиеся к найденному ранее элементу с наибольшей оценкой, а также вычеркивается значение, соответствующее обратному пути.

11) Из пунктов, которые еще не были посещены ни одним коммивояжером, выбирается тот пункт, которому соответствует минимальное значение локальной нижней границы. Если при этом еще остаются непосещенные пункты, то осуществляется переход к третьему пункту алгоритма и повторяются действия, описанные в пунктах 3–11, если все пункты посещены, то осуществляется переход к пункту 1.

12) Для каждого коммивояжера строится окончательный маршрут путем сложения всех путей между пунктами. Таким образом будет получен кратчайший маршрут для каждого коммивояжера.

3. Математическая модель затрат на доставку

Для того чтобы решить задачу нахождения оптимального количества арендованных автомобилей, необходимо учесть несколько критериев: минимизировать затраты на аренду, минимизировать затраты на топливо и минимизировать время доставки товаров

конечным потребителям, также нужно учитывать, что продолжительность маршрута для одного автомобиля не должна превышать продолжительность рабочего дня водителя, то есть 8 часов.

Для решения задачи оптимизации построим математическую модель, описывающую затраты компании на доставку дневного пула заказов. Формула затрат для одного автомобиля выглядит следующим образом:

$$S = p_{\text{ар}} + \sum_{k=1}^{N} p_{\text{т}}\left(i_{k-1}, i_k\right).$$

В данной формуле использованы следующие величины:

- S — суммарные затраты на доставку заказов одним автомобилем;
- $p_{\text{ар}}$ — стоимость суточной аренды одного автомобиля;
- $p_{\text{т}}(a, b)$ — стоимость топлива на маршрут из пункта a в пункт b;
- N — количество доставляемых заказов одним автомобилем;
- i_k — k-й пункт доставки для автомобиля.

Тогда формула суммарных затрат для n автомобилей примет следующий вид:

$$S_n = np_{\text{ар}} + \sum_{i=1}^{n} \sum_{k=1}^{N_i} p_{\text{т}}\left(i_{k-1}, i_k\right).$$

Для нахождения оптимального количества арендованных автомобилей будем минимизировать данную функцию по n.

Рассмотрим небольшой демонстрационный пример. Используя модифицированный метод ветвей и границ, рассчитаем кратчайшие маршруты для разного количества автомобилей и внесем полученные результаты расчетов в табл. 1.

Как можно видеть из приведенных данных, при количестве автомобилей, большем 5, затраты на топливо снижаются незначительно, и максимальное время доставки товара потребителю тоже уменьшается незначительно, таким образом, становится нецелесообразным привлекать к доставке большее количество автомобилей.

Количество автомобилей	Затраты на аренду	Затраты на топливо	Максимальное время доставки
1	2 000	3 619	22
2	4 000	3 176	15
3	6 000	3 032	7
4	8 000	2 911	5,5
5	10 000	2 786	4
6	12 000	2 714	3,5
7	14 000	2 754	3

Таблица 1.

Заключение

В рамках данного исследования была произведена модификация метода ветвей и границ для задачи коммивояжера с несколькими коммивояжерами. Модифицированный алгоритм позволяет определить оптимальное количество автомобилей для распределения заказов и построить для каждого из них кратчайший маршрут. Также была построена математическая модель описания суммарных затрат на доставку дневного пула заказов.

Результаты, полученные в данном исследовании, практически применимы для компаний, имеющих в своей деятельности сегмент онлайн-продаж и организующие доставку товаров со складов конечным потребителям. Разработанный модифицированный алгоритм программно реализован на языке программирования C++ с использованием библиотек Qt и как самостоятельное приложение может использоваться в реальных условиях.

Список литературы

[1] Голубков, Е. П. Методы принятия управленческих решений / 3-е изд., испр. и доп. — М. : Издательство Юрайт, 2018. — 249 с.

[2] Лукинский, С. В. Модели и методы теории логистики : учебное пособие / В. С. Лукинский, В. В. Лукинский, Ю. В. Малевич [и др.] — 2-е изд. — СПб. : Питер, 2008. — 448 с.

Библиографическая ссылка

Акендинов, И. А. Решение транспортно-логистических задач в товарно-распределительной сети // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 5–11.

Сведения об авторах

Акендинов Илья Андреевич
Студент магистратуры
Направление «Прикладная математика и информатика»

УДК 519.854.2

Система регулирования дорожного движения с использованием методов интеллектуального анализа данных

Аладьев И. Д.

Тверской государственный университет

Аннотация. В дорожно-транспортных происшествиях за 2020 год погибли свыше 16 тыс. человек, еще 183 тыс. получили ранения. Это свидетельствует о необходимости сделать дорожное движение более безопасным. Одним из способов решения данной проблемы, наряду с проведением профилактических мероприятий с населением, является оптимальное регулирование движения транспортных средств на перекрестках, контролируемых светофорами. В статье реализуются методы решения данной задачи. Результаты исследования демонстрируются заключительной таблицей.

Введение

Первый в мире светофор появился в 1868 году. Он был установлен в Лондоне для управления движением конных повозок. Сегодня светофоры можно найти на перекрестках почти в каждом городе мира, что делает безопаснее движение транспорта через них.

Светофоры имеют как минимум два состояния и используют один цвет (обычно красный) для сигнала «стоп» и еще один (обычно зеленый), чтобы указать, что автомобили могут проехать. Самые первые светофоры управлялись вручную. В настоящее время они автоматические, а это означает, что они должны быть тщательно спроектированы и рассчитаны по времени, чтобы оптимизировать общее время путешествия для всех участников дорожного движения.

1. Описание и обработка исходных данных

Каждый набор входных данных предоставляется в виде простого текстового файла. Файл содержит только ASCII символы со строка-

ми, заканчивающимися одним символом «\n». Каждые два числа в одной строке разделяются одним пробелом.

Первая строка содержит пять чисел:

1) D — продолжительность моделирования в секундах, целое число такое, что $1 \leqslant D \leqslant 10^4$.
2) I — количество перекрестков (с идентификаторами от 0 до $I-1$), целое число такое, что $2 \leqslant I \leqslant 10^5$.
3) S — количество улиц, целое число такое, что $2 \leqslant S \leqslant 10^5$.
4) V — количество автомобилей, целое число такое, что $1 \leqslant V \leqslant 10^3$.
5) F — бонусные баллы за каждую машину, которая достигает свой пункт назначения до конца симуляции D, целое число такое, что $1 \leqslant F \leqslant 10^3$.

Следующие S строк содержат описания улиц. Каждая строка содержит два целых числа B и E ($0 \leqslant B < I$, $0 \leqslant E < I$) — начальный и конечный перекресток соответственно, название улицы (строка, содержащая от 3 до 30 строчных букв a–z и символ «–»), а также целое число L ($1 \leqslant L \leqslant D$) — время, необходимое автомобилю, чтобы добраться от начала до конца этой улицы.

Следующие V строк описывают пути каждой машины. Каждая строка содержит целое число P ($2 \leqslant P \leqslant 10^3$) — количество улиц, по которым машина хочет проехать, а затем P названий улиц, описывающих маршрут. Путь машины всегда действителен, то есть улицы обязательно будут соединены перекрестками.

Набор исходных данных был взят с веб-портала kaggle.com. Изучим подробнее этот набор. Время моделирования равно 7 071 секунде, план города насчитывает 8 000 перекрестков и 63 968 улиц.

Каждая улица имеет характеристику длины. В наборе исходных данных имеем, что минимальное время прохождения улицы равно одной секунде, максимальное — 50 секундам, а среднее значение величины равно 25 секундам.

Агентами в данной системе являются машины, в наборе данных насчитывается 1 000 машин. Для каждой машины заранее известен маршрут, каждая машина имеет протяженность маршрута равную 120 улиц.

Рассчитав время прохождения каждой улицы в маршруте каждой машины, получили следующие данные: среднее время в пути машины составляет 3 051 секунду, минимальное время маршрута — 2 521 секунда, максимальное — 3 536 секунд.

2. Методы анализа

В рамках исходной постановки с учетом установленных ограничений и допущений были получены следующие модели:

$$Q \to \max,\quad \begin{cases} 1 \leqslant D \leqslant 10^4, \\ 2 \leqslant I \leqslant 10^5, \\ 2 \leqslant S \leqslant 10^5, \\ 1 \leqslant V \leqslant 10^3, \\ Q \geqslant 0, \end{cases} \tag{1}$$

$$K \to \max,\quad \begin{cases} 1 \leqslant D \leqslant 10^4, \\ 2 \leqslant I \leqslant 10^5, \\ 2 \leqslant S \leqslant 10^5, \\ 1 \leqslant V \leqslant 10^3, \\ 0 \leqslant K \leqslant V, \end{cases} \tag{2}$$

$$T \to \min,\quad \begin{cases} 1 \leqslant D \leqslant 10^4, \\ 2 \leqslant I \leqslant 10^5, \\ 2 \leqslant S \leqslant 10^5, \\ 1 \leqslant V \leqslant 10^3, \\ 0 \leqslant T \leqslant D. \end{cases} \tag{3}$$

Рассмотрим модель (1), полученную из исходной постановки задачи. Данная модель максимизирует суммарное количество очков,

получаемых за каждую завершившую свой путь машину. Следует отметить следующие положительные и отрицательные стороны данной модели: к положительным сторонам отнесем слабое влияние «выбросов» на результат, при этом решение не учитывает все исходные данные. Модели (2) и (3) получены путем введения дополнительных параметров.

Основная сложность данной задачи заключается в выборе приоритета для улицы, входящей в перекресток. Рассмотрим пять основных способов выбора приоритета.

Первый способ является интуитивным. В данном случае приоритет при проезде перекрестка отдается тем улицам, на которых разрешающий сигнал светофора ждет большее количество автомобилей.

Второй и третий способы предназначены для разгрузки системы. В данном случае приоритет отдается входящим улицам, на которых автомобилям осталось меньшее количество улиц (второй способ) или меньшее время, необходимое для достижения финиша (третий способ).

Четвертый и пятый подходы получены с учетом того факта, что в исходных данных средний путь машины не превышает половины длительности симуляции, вследствие чего было сделано предположение о том, что все машины в любом случае завершат свой маршрут. Исходя из этого предположения, четвертый способ заключается в том, чтобы отдавать приоритет на проезд перекрестка входящим улицам, на которых среднее время, необходимое для завершения пути автомобилями, максимально. Пятый способ аналогичен четвертому, разница лишь в том, что приоритет отдается улицам, на которых максимально среднее количество оставшихся улиц в маршрутах автомобилей, ожидающих разрешающий сигнал светофора.

Данную задачу в рамках одного перекрестка можно рассматривать как частный случай задачи коммивояжера, в том смысле, что каждый перекресток в конечном итоге пропустит заранее известное число автомобилей, и мы лишь можем менять местами этот порядок.

3. Метод имитации отжига

Одним из самых популярных методов решения задачи коммивояжера является метод имитации отжига. Алгоритм имитации от-

жига — общий алгоритмический метод решения задачи глобальной оптимизации, особенно дискретной и комбинаторной оптимизации. Один из примеров методов Монте-Карло. Название алгоритма является отсылкой к физическому процессу, который происходит при кристаллизации вещества, в том числе при отжиге металлов.

Алгоритм является итеративным, новое решение получаем на основе текущего следующим образом: меняем местами две улицы в маршруте, в результате чего получается новое расписание x^*. Расписание x^* становится следующим по алгоритму расписанием с вероятностью P, которая вычисляется в соответствии с распределением Гиббса:

$$P(\overline{x^*} \to \overline{x_{i+1}} \mid \overline{x_i}) = \begin{cases} 1, & F\left(\overline{x^*}\right) - F(\overline{x_i}) < 0, \\ \exp\left(-\dfrac{F\left(\overline{x^*}\right) - F(\overline{x_i})}{Q_i}\right), & F\left(\overline{x^*}\right) - F(\overline{x_i}) \geqslant 0. \end{cases}$$

Здесь F — оценочная функция, Q_i — элементы произвольной убывающей, сходящейся к нулю положительной последовательности, которая задает аналог падающей температуры в кристалле.

Алгоритм имитации отжига требует базовое решение, в качестве которого использованы расписания, полученные на основе пяти подходов, описанных в разделе 2.

4. Программная реализация

Для эффективной разработки построенных моделей была спроектирована архитектура классов, изображенная на рис. 1 в формате UML-диаграммы.

Рассмотрим подробнее спроектированную структуру классов. Абстрактным классом является класс Algorithm, отвечающий за общую логику работы алгоритма. Данный класс содержит в себе общие сведения о задаче, а именно: длительность моделирования, количество машин и перекрестки в плане города. Помимо этого, данный класс определяет стратегию решения задачи. Для использования нового метода решения задачи достаточно реализовать класс, производный от данного.

За логику поведения перекрестков отвечает класс Crossroad. Он содержит в себе все входящие в него и исходящие из него улицы. Данный класс также является базовым, аналогично классу Algorithm, а

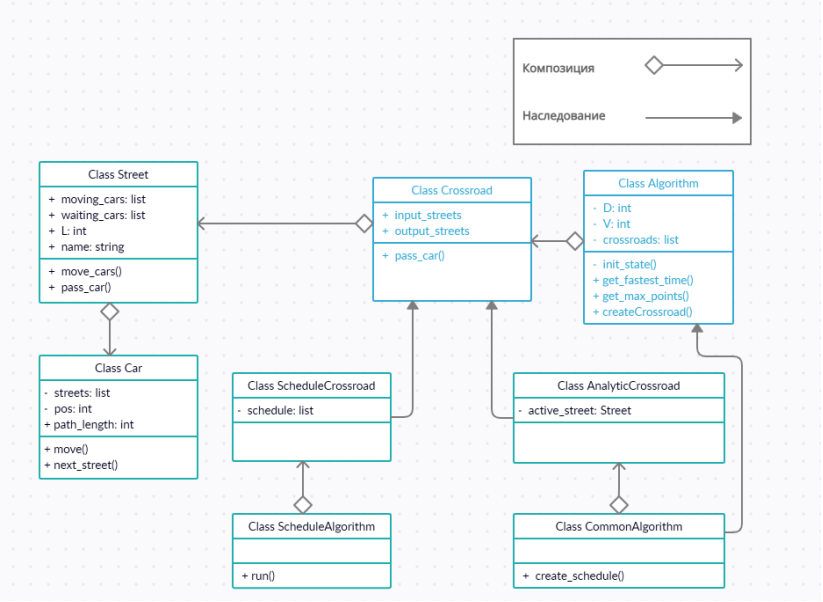

Рис. 1. Диаграмма структуры классов.

значит для конкретной стратегии решения задачи необходим соответствующий производный от Crossroad класс, описывающий логику проезда автомобилей через перекресток.

Для реализации логики поведения улиц был спроектирован класс Street. В нем хранится информация о длине и названии улицы, а также всех машинах, которые совершают движение по данной улице или закончили движение по ней и ожидают зеленого сигнала светофора.

И, наконец, класс Car содержит в себе сведения о машине, ее маршруте и текущем положении.

При описании предметной области задачи с помощью структуры классов были применены следующие паттерны.

Шаблонный метод. Рассмотрим подробнее эти паттерны. Общая структура паттерна шаблонный метод представлена ниже на рис. 2.

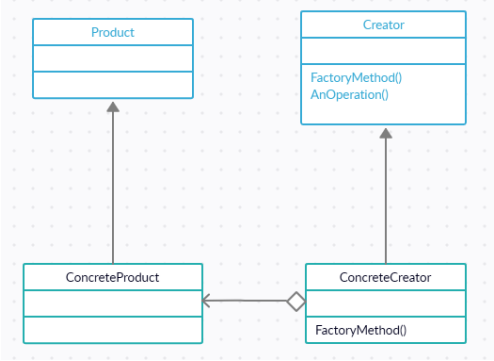

Рис. 2. Шаблонный метод.

Основные участники:

- Product — продукт, определяет интерфейс объектов, создаваемых фабричным методом. В данном случае эту роль играет класс Crossroad. ConcreteProduct (AnalitycCrossroad) — конкретный продукт, реализует интерфейс Product.

- Creator — создатель. Он объявляет фабричный метод, возвращающий объект типа Product. В данном случае создателем является класс Algorithm, а фабричный метод назван createCrossroad. ConcreteCreator (CommonAlgorithm) — конкретный создатель, замещает фабричный метод, возвращая объект ConcreteProduct.

Фабричный метод избавляет проектировщика от необходимости встраивать в код зависящие от приложения классы. Код имеет дело только с интерфейсом от класса Product, а значит он может работать с любыми определенными пользователями классами конкретных продуктов.

Стратегия. Общая структура паттерна стратегия представлена на рис. 3.

Основные участники:

- Strategy — стратегия объявляет общий для всех поддерживаемых алгоритмов интерфейс. Класс Context пользуется этим

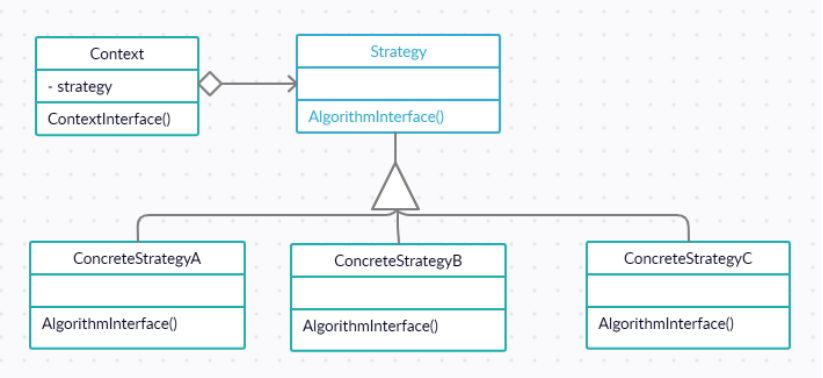

Рис. 3. Стратегия.

интерфейсом для вызова конкретного алгоритма, определенного в классе ConcreteStrategy. В данном случае эту роль играет класс Algorithm. ConcreteStrategy (CommonAlgorithm) — конкретная стратегия, реализует алгоритм, использующий интерфейс класса Strategy.

- Context — контекст. Он настраивается объектом класса ConcreteStrategy, хранит ссылку на объект класса Strategy и может определять интерфейс, который позволяет объекту Strategy обращаться к данным контекста.

Иерархия классов Strategy определяет семейство алгоритмов или вариантов поведения, которые можно повторно использовать в разных контекстах. Наследование позволяет вычленить общую для всех алгоритмов функциональность.

5. Анализ результатов

Результаты проведенного исследования представлены в таблице, отражающей данные о времени выполнения и точности рассмотренных методов (табл. 1) по рассмотренным в разделе 2 оцениваемым параметрам.

Алгоритм	Q	K	T	Время выполнения
Первый	4 415 423	1 000	3 582	13м 16с
Второй	4 414 727	1 000	3 614	14м 18с
Третий	4 415 423	1 000	3 611	15м 48с
Четвертый	4 416 731	1 000	3 566	14м 21с
Пятый	4 417 469	1 000	3 563	15м 31с

Таблица 1. Результаты тестирования.

По полученным данным, при решении моделей (1) и (3) лучше всего подходит пятый способ выбора приоритета проезда перекрестка. Для модели (2) все подходы показали одинаковый результат.

Заключение

В процессе работы был проведен анализ исходных данных, изучение необходимых алгоритмов, а также их применение к задаче регулировки дорожного движения на основе известных маршрутов транспортных средств.

Дальнейшая работа в данной области предполагает повышение точности работы алгоритма за счет применения других алгоритмов или расширения входных данных, добавив больше улиц и автомобилей. Также возможно повысить скорость работы приложения, применив новейшие архитектурные решения.

Список литературы

[1] Гамма, Э. Паттерны объектно-ориентированного проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Д. Влиссидес ; пер. с англ. А. А. Слинкина. — СПб. : Питер, 2020. — 448 с.

[2] Джонс, М. Т. Программирование искусственного интеллекта в приложениях / пер. с англ. А. И. Осипова. — М. : ДМК Пресс, 2004. — 312 с. : ил.

[3] Дорожно-транспортная аварийность в Российской Федерации за 9 месяцев 2018 года. Информационно-аналитический обзор. — М. : ФКУ НИЦ БДД МВД России, 2018. — 17 с.

Библиографическая ссылка

Аладьев, И. Д. Система регулирования дорожного движения с использованием методов интеллектуального анализа данных // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 12–21.

Сведения об авторах

Аладьев Илья Дмитриевич
Студент магистратуры
Направление «Фундаментальная информатика и информационные технологии»

УДК 519.8

Квазиэффективная граница множества инвестиционных возможностей в условиях гибридной неопределенности при запрещенных коротких продажах

Ашрафова С. А.

Тверской государственный университет

Аннотация. В данной статье разработана и исследована модель портфеля минимального риска в условиях гибридной неопределенности возможностно-вероятностного типа с ограничениями на уровень ожидаемой доходности при запрещенных «коротких» продажах. Отличительной особенностью работы является тот факт, что на ее основе построена квазиэффективная (эффективная с заданной возможностью) граница множества инвестиционных возможностей.

Введение

Цель каждого инвестора — получить наибольшую прибыль, однако в инвестировании важен не только доход, но и степень риска. Согласно теории Марковица, можно спроектировать оптимальный портфель с идеальным балансом между риском и доходностью и построить эффективную границу — набор оптимальных портфелей, которая должна помочь инвестору сделать выбор. В данной работе разработана модель портфеля минимального риска в условиях гибридной неопределенности возможностно-вероятностного типа при запрещенных «коротких» продажах и на ее основе построена квазиэффективная граница. Нечеткость позволяет более эффективно моделировать инвестиционный портфель и прогнозировать доходность при определенном уровне риска (либо, наоборот, риск при определенном уровне доходности).

1. Основные элементы теории возможности

В работе используются определения и понятия из теории возможностей [2, 3]. Введем также некоторые леммы необходимые в дальнейшем.

Рассмотрим случайную величину $X(w,\gamma) = a(w) + \delta(w)X(\gamma)$, имеющую сдвиг-масштабное представление, случайные параметры которой — случайные величины с конечными моментами первого и второго порядка, и определим ее числовые характеристики [2].

Лемма 1. *Пусть $X(w,\gamma) = a(w) + \delta(w)X(\gamma)$, где $a(w)$ и $\delta(w)$ — случайные величины с математическими ожиданиями a^0, δ^0 и дисперсиями σ_a^2, σ_δ^2 соответственно, $a(w)$ и $\delta(w)$ — некоррелированные случайные величины, $X(\gamma)$ — нечеткая величина. Тогда $X(w,\gamma)$ — нечеткая случайная величина, имеющая математическое ожидание $a^0 + \delta^0 t$ и дисперсию $\sigma_a^2 + \sigma_\delta^2 t^2$ с возможностью $\mu_X(t)$.*

Лемма 2. *Пусть*

$$X_i(w,\gamma) = a_i(w) + \delta_i(w)X_i(\gamma), \quad i = \overline{1,n},$$

$X_i(\gamma)$ — минисвязные нечеткие величины, $a_i(w)$ и $\delta_i(w)$ — случайные величины, имеющие конечные моменты второго порядка, $\sum_{i=1}^{n} \omega_i = 1$. Тогда дисперсия взвешенной суммы нечетких случайных величин исчисляется по формуле

$$D\left(X(\omega,w,\gamma)\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} \operatorname{cov}(X_i(w,\gamma), X_j(w,\gamma))\omega_i\omega_j,$$

где ковариации и дисперсии вычисляются по формуле Фенга [3]:

$$\operatorname{cov}(X,Y) = \frac{1}{2}\int_0^1 \big(\operatorname{cov}(X^-(r), Y^-(r)) + \operatorname{cov}(X^+(r), Y^+(r))\big)dr.$$

2. Портфель минимального риска при ограничениях на уровень ожидаемой доходности

Рассмотрим модель минимизации риска портфеля с ограничениями на ожидаемую доходность. В соответствии с классическим

подходом требуется построить функцию риска, а доходность внести в систему ограничений. Для снятия неопределенности добавим в систему ограничений, определяющую множество допустимых портфелей, ограничение на приемлемый для инвестора уровень ожидаемой доходности, после чего построим эквивалентный детерминированный аналог системы ограничений. Для начала определим все компоненты модели.

Доходность инвестиционного портфеля при нашем подходе есть нечеткая величина. Она представляется линейной функцией долей капитала $\omega = (\omega_1, \omega_2 \dots \omega_n)$ и имеет следующий вид:

$$R(\omega, w, \gamma) = \sum_{i=1}^{n} R_i(w, \gamma)\omega_i,$$

где $R_i(w, \gamma)$ — доходности отдельных активов. Они моделируются с помощью сдвиг-масштабного представления нечетких случайных величин:

$$R_i(w, \gamma) = a_i(w) + \delta_i(w)X_i(\gamma).$$

Условимся, что нечеткие величины $X_i(\gamma) \in Tr(c_i, d_i)$ взаимно минисвязны, то есть связаны t-нормой T_m, а $a_i(w)$, $\delta_i(w)$ — коэффициенты сдвига и масштаба — независимо равномерно распределенные случайные величины, имеющие равномерное распределение на отрезке $[0, 1]$:

$$R_i(w, \gamma) \in Tr(a_i(w) + \delta_i(w)c_i, \delta_i(w)d_i).$$

В таком случае доходность инвестиционного портфеля также является нечеткой величиной триангулярного типа, а точнее

$$R(\omega, w, \gamma) \in Tr\Big(\sum_{i=1}^{n} \left(a_i(w) + \delta_i(w)c_i\right)\omega_i, \sum_{i=1}^{n} \delta_i(w)d_i\omega_i\Big).$$

В соответствии со свойствами математического ожидания мы можем определить ожидаемую доходность портфеля. Она будет определяться следующей формулой:

$$E\{R(\omega, w, \gamma)\} = \sum_{i=1}^{n} \Big(\widetilde{a}_i(w) + \widetilde{\delta}_i(w)X_i(\gamma)\Big)\,\omega_i.$$

Вычислим ожидаемую доходность портфеля при сделанных нами предположениях:

$$E\{R(\omega, w, \gamma)\} = \sum_{i=1}^{n} \left(\frac{1}{2} + \frac{1}{2} X_i(\gamma) \right) \omega_i.$$

Теперь, используя все приведенные обозначения и результаты, мы построим модель допустимых портфелей в возможностном контексте:

$$\begin{cases} \pi \left\{ E \left\{ R(\omega, w, \gamma) \right\} \geqslant r \right\} \geqslant \alpha, \\ \sum\limits_{i=1}^{n} \omega_i = 1, \\ \omega_i \geqslant 0, \ i = 1, \ldots, n, \end{cases} \tag{1}$$

где $E\left\{R(\omega, w, \gamma)\right\}$ — ожидаемая доходность, π — мера возможности, α — уровень возможности, r — уровень доходности, приемлемый для инвестора.

В соответствии с результатами, полученными в [1], получаем эквивалентный детерминированный аналог системы (1) в возможностном контексте:

$$\begin{cases} \sum\limits_{i=1}^{n} \widetilde{R}_i^+(\alpha) \omega_i \geqslant r, \\ \sum\limits_{i=1}^{n} \omega_i = 1, \\ \omega_i \geqslant 0, \ i = 1, \ldots, n, \end{cases}$$

где $\widetilde{R}_i^+(\alpha)$ — правые границы α-уровневых множеств математических ожиданий доходностей отдельных активов.

Вычислим правую границу α-уровневого множества для системы в возможностном контексте. В соответствии с [2]

$$\widetilde{R}_i^+(\alpha) = \widetilde{a}_i(w) + \widetilde{\delta}_i(w) c_i + \frac{\widetilde{\delta}_i(w) d_i}{2} (1 - \alpha).$$

При сделанных предположениях

$$\widetilde{R}_i^+(\alpha) = \frac{1}{2} \left(1 + c_i + \frac{1}{2} d_i (1 - \alpha) \right).$$

Для завершения построения модели определим функцию риска портфеля. При сделанных предположениях и обозначениях риск портфеля оценивается четкой дисперсией, согласно формуле Фенга [3]:

$$D\{R(\omega,w,\gamma)\} = \sum_{i=1}^{n}\omega_i^2\mathrm{cov}\{R_i(w,\gamma),R_i(w,\gamma)\}+ \\ +2\sum_{i<j}^{n}\omega_i\omega_j\mathrm{cov}(R_i(w,\gamma),R_j(w,\gamma)).$$

При сделанных нами предположениях

$$D\{R(\omega,w,\gamma)\} = \frac{1}{12}\sum_{i=1}^{n}\omega_i^2\left(1+c_i+\frac{d_i}{12}\right).$$

После определения всех компонент модель портфеля минимального риска может быть записана в следующем виде:

$$D(R(\omega,w,\gamma))\to\min,\\ \begin{cases}\displaystyle\sum_{i=1}^{n}\widetilde{R}_i^{+}(\alpha)\omega_i\geqslant r,\\ \displaystyle\sum_{i=1}^{n}\omega_i=1,\\ \omega_i\geqslant 0,\ i=1,\dots,n.\end{cases} \tag{2}$$

3. Построение квазиэффективной границы портфеля минимального риска с запрещенными короткими продажами в возможностном контексте

Из [1] нам уже известен метод построения эффективной границы классического портфеля минимального риска. В настоящей работе мы обобщим этот метод на случай портфеля минимального риска в условиях гибридной неопределенности с нечеткими отношениями в ограничении. С этой целью построим функцию Лагранжа в возможностном контексте:

$$L(\omega, w, \gamma) = \sum_{i=1}^{n} \omega_i^2 \mathrm{cov}(R_i(w,\gamma), R_i(w,\gamma)) + \\ + 2\sum_{i<j}^{n} \omega_i\omega_j \mathrm{cov}(R_i(w,\gamma), R_j(w,\gamma)) + \\ + \lambda_1\Big(\sum_{i=1}^{n} \widetilde{R}_i^+(\alpha)\omega_i - r\Big) + \lambda_2\Big(\sum_{i=1}^{n}\omega_i - 1\Big) - \sum_{i=1}^{n}\mu_i\omega_i.$$

Понятно, что для построения эффективной границы нам нужно найти решение, зависящее от параметра r.

Введем обозначения для более удобной записи. Пусть

$$\sigma_{ij} = \mathrm{cov}\left(R_i(w,\gamma), R_j(w,\gamma)\right),$$
$$\sigma_i^2 = \mathrm{cov}\left(R_i(w,\gamma), R_i(w,\gamma)\right),$$
$$\overline{r}_i = \widetilde{R}_i^+(\alpha).$$

Тогда с использованием этих обозначений функция Лагранжа принимает следующий вид:

$$L(\omega, w, \gamma) = \sum_{i=1}^{n} \omega_i^2\sigma_i^2 + 2\sum_{i<j}^{n} \omega_i\omega_j\sigma_{ij} + \\ + \lambda_1\Big(\sum_{i=1}^{n} \overline{r}_i\omega_i - r\Big) + \lambda_2\Big(\sum_{i=1}^{n}\omega_i - 1\Big) - \sum_{i=1}^{n}\mu_i\omega_i.$$

На основе функции Лагранжа мы получаем следующую систему линейных уравнений:

$$\begin{cases} 2\sigma_1^2\omega_1 + 2\sigma_{12}\omega_2 + \ldots + 2\sigma_{1n}\omega_n + \lambda_1\overline{r}_1 + \lambda_2 - \mu_1 = 0, \\ 2\sigma_{11}\omega_1 + 2\sigma_2^2\omega_2 + \ldots + 2\sigma_{2n}\omega_n + \lambda_1\overline{r}_2 + \lambda_2 - \mu_2 = 0, \\ \qquad\qquad \ldots \\ 2\sigma_{1n}\omega_1 + 2\sigma_{2n}\omega_2 + \ldots + 2\sigma_n^2\omega_n + \lambda_1\overline{r}_n + \lambda_2 - \mu_n = 0, \\ \omega_1 + \omega_2 + \ldots + \omega_n = 1, \\ \overline{r}_1\omega_1 + \overline{r}_2\omega_2 + \ldots + \overline{r}_n\omega_n = r, \end{cases}$$
$$\mu_i\omega_i = 0,\ \mu_i \geqslant 0,\ \omega_i \geqslant 0,\ i = 1, \ldots, n.$$

ω_1	$\dots$	ω_n	λ_1	λ_2	μ_1	μ_2	$\dots$	μ_n	
1	$\dots$	0	0	0	c_{11}	c_{12}	$\dots$	c_{1n}	$a_1 + b_1 r$
0	$\dots$	0	0	0	c_{21}	c_{22}	$\dots$	c_{2n}	$a_2 + b_2 r$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
0	$\dots$	1	0	0	c_{n1}	c_{n2}	$\dots$	c_{nn}	$a_n + b_n r$
0	$\dots$	0	1	0	$c_{n+1,1}$	$c_{n+1,2}$	$\dots$	$c_{n+1,n}$	$a_{n+1} = b_{n+1} r$
0	$\dots$	0	0	1	$c_{n+2,1}$	$c_{n+2,2}$	$\dots$	$c_{n+2,n}$	$a_{n+2} = b_{n+2} r$

Таблица 1.

ω_1	$\dots$	ω_n	μ_1	μ_2	$\dots$	μ_n	
1	$\dots$	0	c_{11}	c_{12}	$\dots$	c_{1n}	$a_1 + b_1 r$
0	$\dots$	0	c_{21}	c_{22}	$\dots$	c_{2n}	$a_2 + b_2 r$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
0	$\dots$	1	c_{n1}	c_{n2}	$\dots$	c_{nn}	$a_n + b_n r$

Таблица 2.

Расширенную матрицу приведенной выше системы уравнений можно привести к другой матрице (табл. 1) с помощью метода Гаусса.

Мы можем сократить данную систему, вычеркнув два столбца λ_1 и λ_2 и последние две строки, так как на неизвестные λ_1 и λ_2 не наложено никаких ограничений (табл. 2).

Числа a_i, b_i зависят только от элементов ковариационной матрицы и ожидаемых доходностей $\overline{r}_1, \ \dots, \overline{r}_n$. Значит, какое бы ни было число r, оптимальное решение имеет вид

$$\overline{\omega}_{\min}(r) = \{a_1 + b_1 r, a_2 + b_2 r, \dots, a_n + b_n r\}.$$

Однако у нас есть дополнительное условие на неотрицательность, поэтому среди решений приведенной системы линейных уравнений необходимо выбрать те, которые удовлетворяют условию

$$\mu_i \omega_i = 0, \quad \mu_i \geqslant 0, \quad \omega_i \geqslant 0, \quad i = 1, \dots, n.$$

В соответствии с полученными результатами следует отметить, что решение задачи с помощью метода Гаусса с формальной точки

зрения выглядит также, как в случае с классическим вариантом, соответственно мы можем привести те же следствия, что указаны в [1].

Следствие 3. *Множество инвестиционных возможностей определяется уравнением вида*

$$\sigma = (Ar^2 + Br + C)^{1/2},$$

где $A > 0$, $C > 0$ *и* $B^2 - 4AC < 0$. *А точнее*

$$A = \sum_{i=1}^{n} \sigma_i^2 b_i^2 + 2\sum_{i<j}^{n} \sigma_{ij} b_i b_j,$$
$$B = 2\sum_{i=1}^{n} \sigma_i^2 a_i b_i + 2\sum_{i<j}^{n} \sigma_{ij}(a_i b_j + a_j b_i),$$
$$C = \sum_{i=1}^{n} \sigma_i^2 a_i^2 + 2\sum_{i<j}^{n} \sigma_{ij} a_i a_j.$$

Следствие 4. *Эффективная граница множества инвестиционных возможностей при разрешенных коротких продажах ценных бумаг определяется уравнением вида*

$$\sigma = (Ar^2 + Br + C)^{1/2}, \text{ где } r \geqslant -\frac{B}{2A}.$$

Следствие 5. *В случае построения эффективной границы при запрещенных «коротких» продажах для выбора решений необходимо рассмотреть несколько случаев:*

1) *Пусть* $\mu_1 = \mu_2 = \ldots = \mu_n = 0$. *Тогда*

$$\overline{\omega_1}(r) = (a_1 + b_1 r, a_2 + b_2 r, \ldots, a_n + b_n r)$$

является решением системы линейных уравнений, приведенной в табл. 2. Тогда $S_1 = \{r \in R \mid a_i + b_i r \geqslant 0,\ i = 1, \ldots, n\}$. *При* $r \in S \cap S_1$ *вектор* $\overline{\omega_1}(r)$ *определяет инвестиционную возможность, которая принадлежит эффективной границе, где* $S = [r^*, \min r_i]$.

2) *Пусть* $\mu_1 \neq 0, \mu_2 = \ldots = \mu_n = 0$, *затем* $\mu_2 \neq 0, \mu_1 = \mu_3 = \ldots = \mu_n = 0$ *и т. д. Тогда на каждом шаге мы находим множество* S_i *и вектор* $\overline{\omega_i}(r)$ *такие, что при* $r \in S \cap S_i$ *вектор определяет инвестиционную возможность, которая принадлежит эффективной границе. Перебор проводится до тех пор, пока не окажется, что* $(S \cap S_1) \cup (S \cap S_2) \cup \ldots \cup (S \cap S_n) = S$.

4. Модельный пример

Рассмотрим в качестве модельного примера трехмерный портфель. Пусть $X_1 \in \mathfrak{T}_{\mathfrak{r}}(2.6, 0.4)$, $X_2 \in \mathfrak{T}_{\mathfrak{r}}(3.8, 0.2)$, $X_3 \in \mathfrak{T}_{\mathfrak{r}}(4.5, 1)$, $\alpha = 0.6$, $a_i(w)$, $\delta_i(w)$ — независимые случайные величины с равномерным распределением на отрезке $[0, 1]$. При данных условиях эквивалентный детерминированный аналог портфеля минимального риска (2) в возможностном контексте принимает вид

$$\frac{109}{360}\omega_1^2 + \frac{289}{720}\omega_2^2 + \frac{67}{144}\omega_3^2 \to \min,$$
$$\begin{cases} \dfrac{46}{25}\omega_1 + \dfrac{121}{50}\omega_2 + \dfrac{57}{20}\omega_3 \geqslant r, \\ \displaystyle\sum_{i=1}^{n} \omega_i = 1, \\ \omega_i \geqslant 0,\ i = 1, \ldots, n. \end{cases}$$

Построим функцию Лагранжа:

$$L = \frac{109}{360}\omega_1^2 + \frac{289}{720}\omega_2^2 + \frac{67}{144}\omega_3^2 + \\ + \lambda_1 \left(\frac{46}{25}\omega_1 + \frac{121}{50}\omega_2 + \frac{57}{20}\omega_3 - r \right) + \\ + \lambda_2 \Big(\sum_{i=1}^{3} \omega_i - 1 \Big) - \sum_{i=1}^{3} \mu_i \omega_i.$$

Продифференцировав функцию Лагранжа и приравняв к нулю результаты дифференцирования, мы получаем систему линейных уравнений. Ее решение представлено в табл. 3.

ω_1	ω_2	ω_3	μ_1	μ_2	μ_3	
1	0	0	-0.2189	0.5152	-0.2955	$2.3065 - \frac{1576}{2015}r$
0	1	0	0.3638	-0.8544	0.4906	$\frac{326}{2015}r - 0.0774$
0	0	1	-0.1447	$\frac{1367}{4030}$	-0.195	$\frac{250}{403}r - 1.229$

Таблица 3.

После подстановки решения в формулу для дисперсии, мы получаем выражение для определения параболы:

$$\sigma = (0.3748r^2 - 0.906r + 2.3159)^{\frac{1}{2}},$$
$$r \geqslant 1.2086,\ S \in [1.2086,\ 2.85].$$

Теперь рассмотрим несколько случаев.

Случай 1. $\mu_1 = \mu_2 = \mu_3 = 0$:

$$S_1 = \left\{ r \in R \left| \begin{array}{l} 2.3065 - \frac{1576}{2015}r \geqslant 0, \\ \frac{326}{2015}r - 0.0774 \geqslant 0, \\ \frac{250}{403}r - 1.229 \geqslant 0 \end{array} \right. \right\} = [1.9812,\ 2.949],$$
$$S \cap S_1 = [1.9812, 2.85].$$

Получаем, что при $r \in [1.9812, 2.85]$ портфели вида

$$\overline{\omega_1}(r) = \left\{ 2.3065 - \frac{1576}{2015}r, \frac{326}{2015}r - 0.0774, \frac{250}{403}r - 1.229 \right\}$$

определяют инвестиционные возможности, принадлежащие квазиэффективной границе.

Случай 2. $\mu_1 \neq 0,\ \mu_2 = \mu_3 = 0$:

$$S_1 = \left\{ r \in R \left| \begin{array}{c} 0.264 + 0.2488r \geqslant 0, \\ -0.2483r + 0.735 \geqslant 0, \\ -2.8339r + 5.6144 \geqslant 0 \end{array} \right. \right\} = [0.98,\ 1.9812],$$

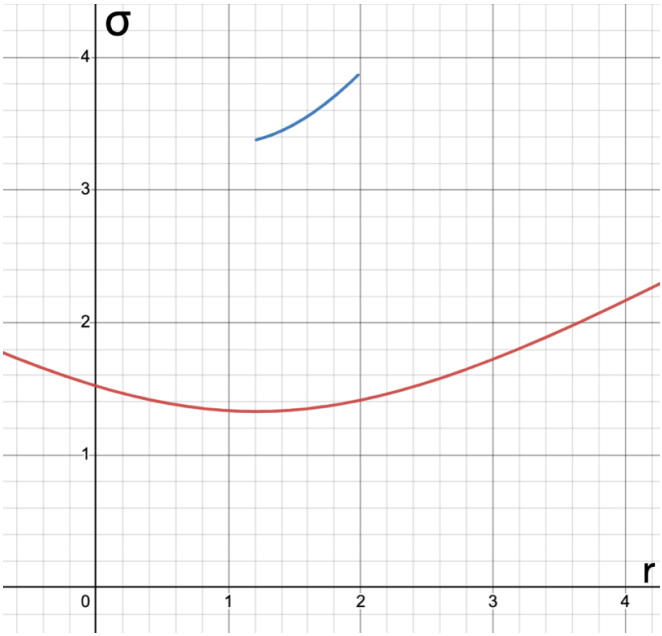

Рис. 1. Квазиэффективная граница в возможностном контексте.

$$S \cap S_2 = [1.2086, 1.9812].$$

Получаем, что при $r \in [1.2086, 1.9812]$ портфели вида

$$\overline{\omega_2}(r) = \{0.264 + 0.2488r, -0.2483r + 0.735, -2.8339r + 5.6144\}$$

определяют инвестиционные возможности, принадлежащие квазиэффективной границе.

Таким образом, квазиэффективная граница задается уравнением

$$\sigma = \begin{cases} (3.7801r^2 - 7.4562r + 14.9042)^{\frac{1}{2}}, & r \in [1.2086, 1.9812], \\ \left(0.3748r^2 - 0.906r + 2.3159\right)^{\frac{1}{2}}, & r \in [1.9812, 2.85]. \end{cases}$$

Квазиэффективная граница представлена на рис. 1.

Заключение

В статье исследована модель портфеля минимального риска в условиях гибридной неопределенности возможностно-вероятностного типа с ограничениями на уровень ожидаемой доходности при запрещенных «коротких» продажах, а также приведен алгоритм построения квазиэффективной границы для спроектированной модели. Также приведен модельный пример — на конкретных данных показан принцип построения квазиэффективной границы.

Список литературы

[1] Барбаумов, В. Е. Финансовые инвестиции : учебное пособие / В. Е. Барбаумов, И. М. Гладких, А. С. Чуйко ; Рос. экон. акад. им. Г. В. Плеханова. — М. : Финансы и статистика, 2003. — 544 с.

[2] Язенин, А. В. Основные понятия теории возможностей: математический аппарат для принятия решений в условиях гибридной неопределенности. — М. : ФИЗМАТЛИТ, 2016. — 144 с.

[3] Feng, Y. The variance and covariance of fuzzy random variables and their applications / Y. Feng, L. Hu, H. Shu // Fuzzy Sets and Systems. — 2001. — Vol. 120, № 2. — P. 487–497.

Библиографическая ссылка

Ашрафова, С. А. Квазиэффективная граница множества инвестиционных возможностей в условиях гибридной неопределенности при запрещенных коротких продажах // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 22–33.

Сведения об авторах

Ашрафова Сабина Алекперовна
Студентка магистратуры
Направление «Фундаментальная информатика и информационные технологии»

УДК 004.931

Метод распознавания формы объектов на основе генетического алгоритма

Барыкин М. М.

Тверской государственный университет

Аннотация. Данная работа посвящена задаче распознавания объектов на изображении с учетом информации о форме этих объектов. В статье описаны используемые входные данные, продемонстрированы шаги по предварительной обработке изображений. Ключевым моментом является описание метода решения, основанного на использовании генетического алгоритма. Для анализа полученных результатов был проведен вычислительный эксперимент и сделаны соответствующие выводы.

Введение

Компьютерное зрение — это область компьютерных наук, которая стремится расширить возможности компьютеров по идентификации и определению объектов и людей на изображениях и видео. Как и другие типы искусственного интеллекта, компьютерное зрение ориентируется на выполнение и автоматизацию задач, имитирующих человеческие возможности. В этом случае компьютерное зрение старается имитировать зрение и восприятие человека.

Технология компьютерного зрения нашла свое применение во многих сферах человеческой деятельности, и их количество продолжает увеличиваться [1]. Наиболее популярными направлениями являются медицина (обнаружение тяжелых болезней, анализ качества проводимого лечения), промышленность (контроль качества продукции), в том числе и военная (обнаружение солдат и техники противника), автономная техника (роботы, беспилотный транспорт и прочее) и так далее.

Классическая задача в компьютерном зрении, обработке изображений и машинном зрении это определение, содержат ли видеоданные некоторый характерный объект, особенность или активность. Люди распознают множество объектов на изображениях без особых

усилий, несмотря на то, что изображение объектов может несколько отличаться в разных точках зрения, в разных размерах и масштабах или даже при их переводе или повороте. Объекты могут быть распознаны даже тогда, когда они частично скрыты от взгляда. Эта задача все еще является проблемой для систем компьютерного зрения.

1. Постановка задачи

Под задачей распознавания объектов понимается задача идентификации объекта или определения каких-либо свойств по его изображению или другим характеристикам. Одним из характерных отличительных признаков каждого объекта является его форма.

Механизм распознавания объекта по заданному эталонному контуру выглядит следующим образом. Шаблон, имеющий установленную форму, перемещается по всему входному изображению. После каждого перемещения происходит подсчет некоторого показателя, характеризующего степень совпадения шаблона с тем, что находится на изображении. Если полученное значение превышает установленный дискретный порог, то это означает, что в текущей области входного изображения может находиться искомый объект.

В рамках работы была поставлена следующая задача: необходимо распознать на изображении объект, зная его форму. То есть дать ответ, есть ли объект заданной формы на изображении.

Входные данные к задаче:

- входное изображение, на котором присутствует объект (или объекты);
- шаблон, который представляет собой полигональный объект и характеризующий форму искомого объекта;
- объект имеет произвольные ориентацию и размеры, то есть совсем не обязательно, что он совпадает с шаблоном.

2. Обработка изображений

Для того, чтобы начать процесс распознавания объекта на изображении, исходное изображение требуется подготовить. Делается

Рис. 1. Исходное изображение.

это с помощью специальной обработки изображений. Происходит обработка в два этапа:

1) удаление шума с изображения;
2) выделение контуров объектов на изображении.

П е р в ы й э т а п выполняется при использовании сглаживающих фильтров и фильтров для удаления шума, таких как фильтр Гаусса, медианный фильтр и другие. Результат применения комбинации фильтров для исходного изображения (рис. 1) представлен на рис. 2.

На в т о р о м э т а п е, после удаления посторонних вещей с входного изображения, применяются фильтры для выделения контуров, такие как фильтр Кирша, Собеля, Превитта и другие. Результат применения фильтра Кирша представлен на рис. 3.

Стоит отметить, что фильтры выделения контуров могут иметь разный размер ядра (3×3, 5×5 и другие), поэтому параметры линии контура (например, толщина) может изменяться. Чем больше размер ядра, тем толще и ярче становится линия. Толстая и яркая линия способствует более качественному поиску объекта на изображении.

3. Описание разработанного алгоритма

Классический генетический алгоритм был предложен в работах Джона Холланда. Изначально генетические алгоритмы разраба-

Рис. 2. Изображение после удаления шума.

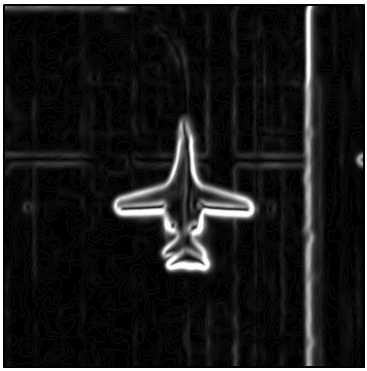

Рис. 3. Изображение после применения фильтра Кирша.

тывались для изучения адаптивных систем, а в настоящее время их активно используют в качестве эффективного инструмента для решения различных оптимизационных задач [2].

Для решения поставленной задачи был разработан модифицированный генетический алгоритм.

Хромосомы представляют собой четверку вида

$$(x, y, angle, scale)\,,$$

где:

- x, y — координаты точки входного изображения;
- $angle$ — величина угла поворота;
- $scale$ — коэффициент масштаба.

Угол поворота и коэффициент масштаба влияют на используемый шаблон, и, соответственно, на построение маски по этому шаблону. Первые два параметра используются для отметки точки, относительно которой производится наложение маски на входное изображение.

Начальная популяция генерируется случайным образом. Значение каждого из четырех параметров для хромосомы задается произвольным значением из диапазона $[0 \dots 1]$.

Кроссовер представлен в виде случайного выбора у одной из родительских особей значения для каждого из четырех параметров.

В качестве мутации применяется механизм случайной перестановки значений параметров в хромосоме.

Функция приспособленности характеризует, насколько хорошо подходят текущие четыре параметра из хромосомы в качестве решения. Получение оценки каждого решения происходит по следующему алгоритму.

1) Декодирование значений параметров хромосомы — перевод значений из диапазона $[0 \dots 1]$ в необходимый диапазон (для x — [0... ширина исходного изображения — 1], для y — [0... высота исходного изображения — 1], для $angle$ — $[0 \dots 360]$, для $scale$ — изменение между установленными дискретными величинами).
2) Создание маски на основе значения угла поворота и коэффициента масштаба.
3) Вычисление функции свертки.

Расчет функции свертки производится по формуле (1):

$$R(x,y) = \frac{\sum\limits_{u,v} \left(M(u,v) \cdot I(x+u, y+v) \right)}{\sum\limits_{u,v} M(u,v)}, \qquad (1)$$

где

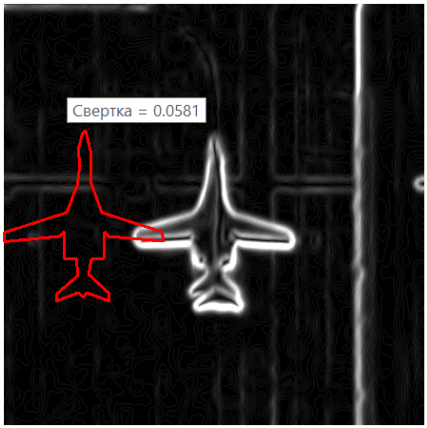

Рис. 4. Полное несоответствие.

- R — результат свертки, $[0 \dots 1]$;
- M — маска шаблона (ядро свертки);
- I — исходное изображение;
- u, v — ширина и высота маски;
- (x, y) — точка входного изображения.

Значение свертки около 0 говорит о полном несоответствии полученной маски с участком исходного изображения, то есть искомый объект не найден (рис. 4). Значение, близкое к 1, соответствует успеху — объект найден (рис. 5). Значение, находящееся между 0 и 1, говорит о частичном соответствии — объект может быть где-то рядом (рис. 6).

Стоит отметить, что на значение результата свертки оказывает влияние качество выполненной предварительной обработки (удаление шума и выделение контуров).

В качестве критериев останова работы алгоритма используются следующие:

- достижение максимального количества эпох;

Рис. 5. Полное соответствие.

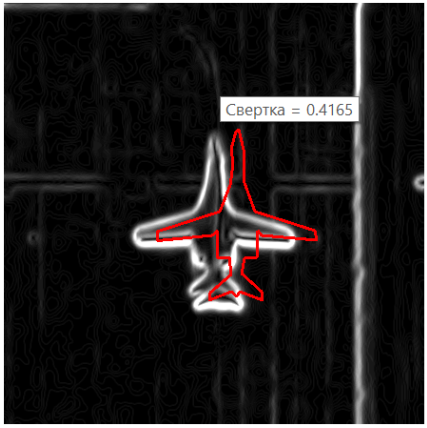

Рис. 6. Частичное соответствие.

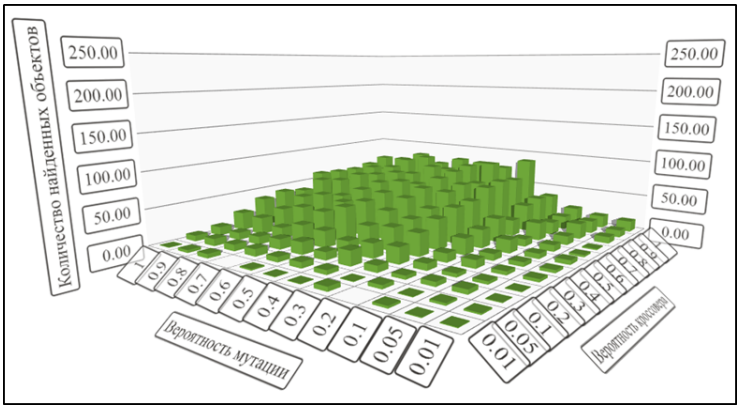

Рис. 7. Статистика успешно найденных объектов.

- достижение необходимого порога по значению свертки (необходимый порог является дискретной величиной).

После окончания работы алгоритма делается вывод о том, есть ли объект искомой формы на данном изображении.

4. Проведение вычислительного эксперимента

Эксперимент заключался в проверке, как сказываются параметры вероятности мутации и кроссовера на качестве работы разработанного алгоритма.

Для проведения вычислительного эксперимента был использован фрагмент с аэросъемки с изображенным на нем самолетом (рис. 1). Для каждой пары значений вероятности мутации и кроссовера проводилось 250 запусков алгоритма распознавания формы объекта на изображении. Сравнение производилось по числу успешно найденных объектов (рис. 7) и среднему времени работы алгоритма (рис. 8). На основе полученных результатов был сделан сравнительный анализ эффективности работы алгоритма с установленными параметрами в единицу времени (рис. 9).

По итогам вычислений были получены следующие результаты:

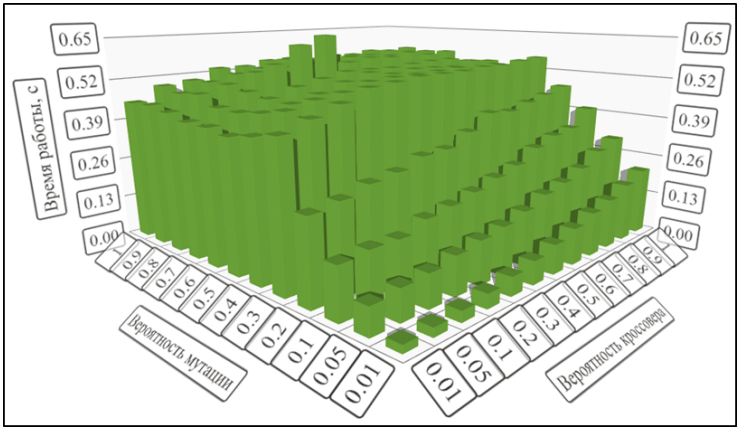

Рис. 8. Временные затраты.

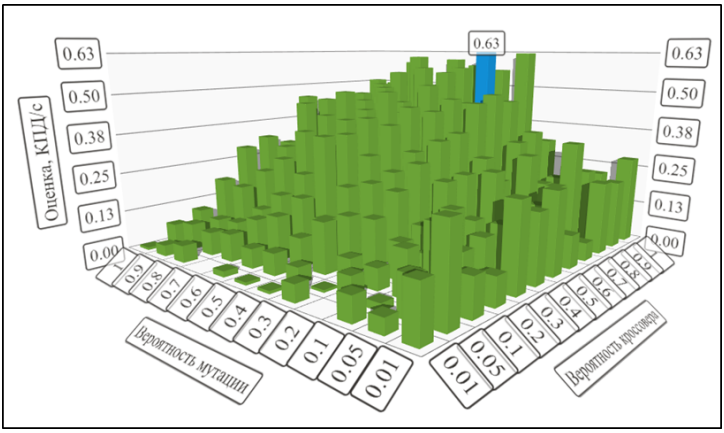

Рис. 9. Оценка качества работы алгоритма.

- наибольшее число распознанных объектов (88) получилось при вероятности мутации 0.3 и вероятности кроссовера 1, что составляет $88/250 \cdot 100\% = 35{,}2\%$ от количества тестов;

- наименьшее затраченное время было получено для варианта с наименьшими показателями вероятности $(0.01, 0.01)$, но при быстрой работе удалось найти объект лишь один раз;

- самой эффективной парой значений стала пара $(0.4, 0.9)$. Объект был распознан 85 раз за самое оптимальное время.

Заключение

В рамках статьи был представлен метод распознавания формы объектов на изображении при использовании генетического алгоритма. Разработанная технология позволяет делать вывод о наличии заданного предмета на изображении в ситуации неизвестности касательно его размеров и ориентации. Также был проведен эксперимент по поиску наиболее эффективной пары значений параметров вероятности мутации и кроссовера.

Дальнейшая работа в рамках данной темы предполагает реализацию механизма нахождения всех объектов указанной формы на изображении. Также вызывают интерес направления, связанные с динамическим изменением параметров генетического алгоритма и стратегией подбора оптимальных параметров с упором на уменьшение скорости работы или на повышения качества работы разработанного алгоритма.

Список литературы

[1] Гонсалес, Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс ; пер. с англ. Л. И. Рубанова, П. А. Чочиа. — Изд. 3-е, испр. и доп. — М. : Техносфера, 2012. — 1104 с.

[2] Саймон, Д. Алгоритмы эволюционной оптимизации / пер. с англ. А. В. Логунова. — М. : ДМК Пресс, 2020. — 1002 с.

Библиографическая ссылка

Барыкин, М. М. Метод распознавания формы объектов на основе генетического алгоритма // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 34–44.

Сведения об авторах

Барыкин Максим Михайлович
Студент магистратуры
Направление «Фундаментальная информатика и информационные технологии»

УДК 004.93 + 004.932

Применение аппарата опорных векторов в распознавании образов

Дорофеев Н. В.

Тверской государственный университет

Аннотация. В статье рассмотрена модификация алгоритма распознавания образов на основе аппарата опорных векторов и диаграмм Воронова. Предложенный метод позволяет снизить ошибку классического метода и существенно повысить скорость работы.

Введение

Задача распознавания образов возникает во многих сферах деятельности, включая оцифровку рукописей, сканирование штрихкодов, распознавание лиц, отпечатков пальцев, голоса и многое другое. Несмотря на наличие разнообразных методов распознавания, актуально решение следующих проблем:

1) большинство систем носят общий характер, что их делает их более гибкими, но менее точными;
2) временные и трудовые затраты на использование универсальных систем не позволяют применять их повсеместно;
3) проблема выделения лексических единиц сложно поддается формализации и существует множество разнородных подходов к ее определению и решению.

В рамках данной статьи рассматривается задача модификации аппарата опорных векторов (SVM) для распознавания отсканированных изображений рукописных текстовых и цифровых символов. Базовый метод опорных векторов — это алгоритм бинарной классификации единичного объекта на основе вектора его признаков. Для связанных образов его необходимо дополнять алгоритмами сепарации символов, их объединения в лексические единицы, такие как слова и строки и модифицировать базовый метод для возможности множественной классификации.

В нашей работе ставятся следующие цели:

1) разработать и реализовать алгоритм распознавания связных образов на основе аппарата опорных векторов и диаграмм Вороного с дополнительными авторскими модификациями;
2) провести сравнительный анализ результатов авторской реализации и классического метода.

Основные отличия авторского метода от классической реализации заключаются в следующем:

1) алгоритм распознавания адаптирован под конкретную предметную область, что должно повысить точность и снизить вероятность ошибки;
2) разработан новый подход к определению некоторых лексических структур, более гибкий и устойчивый к незначительным изменениям в основных характеристиках текста;
3) реализован алгоритм множественного распознавания и подобран оптимальный метод нахождения гиперплоскости в новых условиях.

1. Постановка задачи и краткое описание классических методов ее решения

Разобьем постановку задачи на два этапа.

1) Получение и разбиение исходного текста на лексемы с помощью диаграмм Вороного. Для этого нужно
   - разбить входящий текст на символы;
   - разработать и применить алгоритм объединения символов в слова;
   - разработать и применить алгоритм объединения символов в строки.
2) Распознавание каждого символа с помощью алгоритма SVM на обучающей выборке из набора MNIST. Для этого необходимо
   - очистить входящее изображение от помех;

- выделить вектор признаков полученного изображения;
- классифицировать символы на основе выборки из набора MNIST.

На первом этапе заменим символы центрами их масс, которые вычисляются по формуле

$$c_i = \frac{1}{|S_i|} \sum_{s \in S_i} s, \tag{1}$$

где $|S_i|$ — количество точек, образующих символ, а суммирование ведется по координатам этих точек.

Если S — конечное множество точек, то диаграмма Вороного $V(S)$ представляет собой разбиение плоскости на подмножества

$$V(s_i) = \{p : \forall j \neq i, d(p, s_i) < d(p, s_j)\}, \ s_i \in S,$$
$$V(S) = \bigcup_{i=1}^{N} V(s_i), \tag{2}$$

где $d(p, s)$ — евклидова метрика. Доказано, что ячейка Вороного — это выпуклый многоугольник [1].

Существует множество методов различной сложности для построения такого разбиения. В работе использован алгоритм сложности $O\left(n^2 \log n\right)$, где n — количество центров масс, который позволяет построить локус (ячейку Вороного) для одного сайта (точки) следующим образом:

1) строим $n-1$ серединный перпендикуляр, для прямой, проходящей через выбранный сайт, и каждый из оставшихся генераторов. Эти прямые будут образующими полуплоскостей;
2) каждая из полученных полуплоскостей задается образующей из п. 1 а также своим положением относительно прямой. Оставляем только те полуплоскости, которые лежат по ту же сторону от образующей, что и выбранный сайт;
3) пересекаем все оставшиеся полуплоскости и получаем искомый локус для сайта;

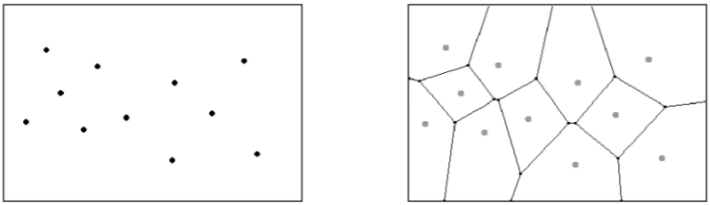

Рис. 1. Множество генераторов и соответствующие ему локусы.

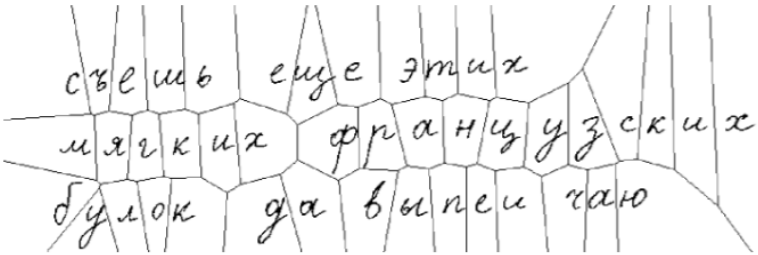

Рис. 2. Диаграмма Вороного множества центров масс.

4) повторяя предыдущие шаги для оставшихся генераторов, получаем полное разбиение плоскости на ячейки Вороного.

Для описания процесса выделения лексических элементов в тексте введем следующие обозначения:

1) $md_s(c, p)$ — расстояние от двух смежных точек генераторов c и p до общей границы ячеек Вороного $V(c)$ и $V(p)$. По построению ячеек диаграммы $md_s(c, p)$ совпадает с половиной евклидова расстояния между точками;

2) пусть c — точка генератор диаграммы Вороного, а соответствующая ей ячейка $V(c)$ ограничена множеством отрезков прямых
$$E = \{e_1, e_2, \ldots, e_n\}.$$
Обозначим наименьшее расстояние от точки c до границы

ячейки $V(c)$ как

$$md_c(c) = \min_{c \in E} d(c, e), \tag{3}$$

где $d(c, e)$ — евклидово расстояние от точки c до отрезка e;

3) пусть имеются два генератора $c = (c_x, c_y)$ и $p = (p_x, p_y)$. Условие принадлежности символов одной строке имеет вид

$$|p_y - c_y| < \alpha \min\{md_c(p), md_c(c)\}. \tag{4}$$

Условие принадлежности символов одному слову задается формулой

$$md_s(p, c) < \beta \min\{md_c(p), md_c(c)\}. \tag{5}$$

В данных условиях $\alpha > 0, \beta > 0$ — некоторые константы. Было установлено что для сепарации большинства текстов оптимально брать $\alpha = 1.3$, $\beta = 1.5$.

Оптимальным методом [1] выделения признаков для SVM служит метод гистограмм ориентированных градиентов (HOG), сущность которого заключается в следующем:

1) для каждого элемента (ij) матрицы исходных данных вычисляются градиенты по вертикальному и горизонтальному направлению через дискретные дифференциальные операторы;

2) для каждого элемента (ij) вычисляются величина и направление градиента:

$$G_{i,j} = \sqrt{\left(G^x_{i,j}\right)^2 + \left(G^y_{i,j}\right)^2},$$
$$\Theta_{i,j} = \operatorname{arctg}\left(\frac{G^y_{i,j}}{G^x_{i,j}}\right); \tag{6}$$

3) матрица разбивается на ячейки $C_{m,n}$, каждая из которых состоит из $w \times v$ значений. Далее для каждой из непересекающихся ячеек вычисляют гистограммы по t направлениям:

$$H_{m,n} = \left\{ \sum_{|\Theta_{i,j} - 2\frac{\pi k}{t}| \leqslant \frac{\pi}{t}} |G_{i,j}| : \forall (i,j) \in C_{m,n}, k = \overline{1,t} \right\}; \tag{7}$$

4) ячейки $C_{m,n}$ объединяются в блоки $B_{p,q}$, внутри которых гистограммы ячеек нормализуют:

$$H'_{m,n} = \frac{H_{m,n}}{\left\| \bigcup\limits_{H_{m,n} \subset B_{p,q}} H_{m,n} \right\|}; \tag{8}$$

5) после чего формируется вектор признаков

$$HOG = \left[H'_{0,0} \dots H'_{0,n} \dots H'_{m,n} \right], \tag{9}$$

который и подается на вход SVM.

SVM — это метод бинарной классификации, сущность которого заключается в следующем. Пусть существует два класса объектов — A и B с конечным числом элементов в каждом классе, а также два множества X и Y, где $X = \{x_1, \dots, x_n\}$ — обучающая выборка, а $Y = \{y_1, \dots, y_n\}$ — набор меток:

$$y_i = \begin{cases} 1, & \text{при } x_i \in A, \\ -1, & \text{при } x_i \in B. \end{cases}$$

Цель алгоритма — на основе обучающей выборки представить такую линейную разрешающую функцию, что

$$\begin{aligned} f(x_i) > 0 \quad & \forall x_i \in A, \\ f(x_i) < 0 \quad & \forall x_i \in B. \end{aligned} \tag{10}$$

Можно показать что нахождение функции сводится к поиску седловой точки лагранжиана [3]

$$L(w, b, \lambda) = \frac{1}{2} \langle w, w \rangle - \sum_{i=1}^{N} \lambda_i \left(y_i (\langle w, x_i \rangle + b) - 1 \right)$$

при условиях

$$\lambda_i \left(y_i (\langle w, x_i \rangle + b) - 1 \right) = 0, \quad \lambda_i \geqslant 0, \quad i \in \overline{1, N},$$

что в свою очередь сводится к задаче оптимизации:

$$\varphi(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \left\| \sum_{i=1}^{N} \lambda_i y_i x_i \right\|^2$$

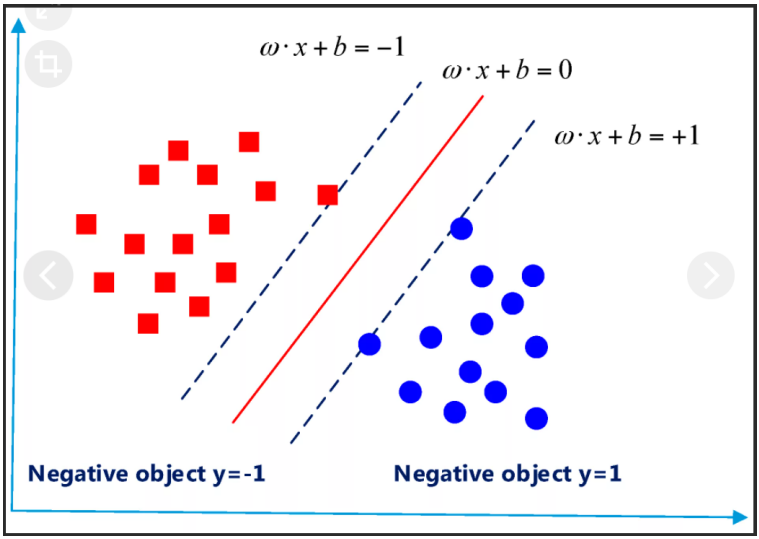

Рис. 3. Разделяющая плоскость.

$$\lambda_i \geqslant 0, \quad i \in \overline{1,N} \quad \sum_{i=1}^{N} \lambda_i y_i = 0 \tag{11}$$

После нахождения критической точки строим разделяющую плоскость

$$w = \sum_{i=1}^{N} \lambda_i y_i x_i'; \quad b = \frac{1}{y_n} - \langle w, x_n \rangle; \quad f(x) = \langle w, x \rangle + b, \tag{12}$$

имеющую максимальный отступ от объектов обоих классов (рис. 3).

2. Авторская модификация

Внесем несколько модификаций в классическую реализацию.

Будем считать, что на вход подаются только изображения символов.

Классификатор на случай $k > 2$ классов. Метод «один–против–одного» заключается в построении $0.5n(n-1)$ бинарных классификаторов для каждой пары векторов обучающей выборки и классификации вектора признаков x_{n+1} с помощью каждого из них. Таким образом можно строить разделяющую плоскость всего по двум точкам, но множественное число раз.

Исходя из этого, вектор весов будет найден как

$$w = \lambda x_1 - \lambda x_2 = \lambda(x_1 - x_2). \tag{13}$$

Следовательно, решение о принадлежности вектора X_3 будет выносится исходя из значения результирующей функции f

$$f = \left(\sum_{i=1}^{n} \lambda\left(x_{1,i} - x_{2,i}\right)x_{3,i}\right) + 1 - \sum_{i=1}^{n} \lambda\left(x_{1,i} - x_{2,i}\right)x_{1,i}, \tag{14}$$

что менее затратно по сравнению с классической реализацией. А поиск $0.5n(n-1)$ бинарных классификаторов легко поддается распараллеливанию.

Алгоритм поиска соседних символа в рамках одного слова. Дано: константы — α, β, точка генератора — c, для которой необходимо найти соседний символ, U — множество точек генераторов, которые допускаются в решении.

1) Перебором ребер, ограничивающих ячейку Вороного, порожденную точкой $c = (c_x, c_y)$, найти точку $p = (p_x, p_y)$ — смежную с ней, такую что выполняются следующие условия:

$$\begin{cases} p \in U, \\ |p_y - c_y| < \alpha \min(md_c(p), md_c(c)), \\ md_s(p, c) < \beta \min(md_c(p), md_c(c)). \end{cases} \tag{15}$$

2) Если точка p не найдена, вернуть $\varnothing$.
3) Обновить множество генераторов $U = U \setminus \{p\}$.
4) Вернуть p.

Алгоритм выделения слова. Дано: точка генератора — c, соответствующая начальному символу слова, U — множество точек генераторов, которые допускаются в решении.

1) Инициализировать список w, задающий порядок букв слова.
2) Положить точку c в список w.
3) Положить $p = c$.
4) Найти соседний символ (p, U) по указанному ранее алгоритму и присвоить его в c.
5) Если $c \neq \varnothing$, добавить p в конец списка.
6) Если $c = \varnothing$, вернуть список w.
7) Перейти на шаг 3.

Алгоритм выделения строк. Дано: список слов, полученных по алгоритму 2 считыванием слева направо.

1) Выбираем левое слово расположенное в самом верху — w_1.
2) Инициализируем L_i — список слов строки и добавляем в него слово w_1.
3) Инициализируем переменную $Ln[i]$ — средняя всех координат y входящих в первое слово списка L_i.
4) Двигаясь слева направо, находим новое слово w_2 и вычисляем для него $Ln'[i]$.
5) Если хотя бы для одного списка L_i выполнено
$$|Ln'[i] - Ln'[i]| < \gamma \min\{md_c(w_1), md_c(w_2)\}, \qquad (16)$$
то добавляем новое слово в список L_i. Здесь $\min\{md_c(w_1), md_c(w_2)\}$ есть минимальная из констант md_c среди символов, принадлежащих слову.
6) Если для всех списков L_i выполнено
$$|w_1 - w_2| > \gamma \min(md_c(w_1), md_c(w_2)), \qquad (17)$$
то инициализируем новый список L_i и добавляем в него новое слово. Так же рассчитываем для него переменную $Ln[i]$.
7) Если существует слово, не отнесенное к списку L_i, то перейти на шаг 4, иначе — на шаг 8.
8) Упорядочить списки L_i по значению средних y.
9) Вернуть списки L_i — списки слов, принадлежащих строке с номером i.

Оптимально полагать $\gamma = \alpha$.

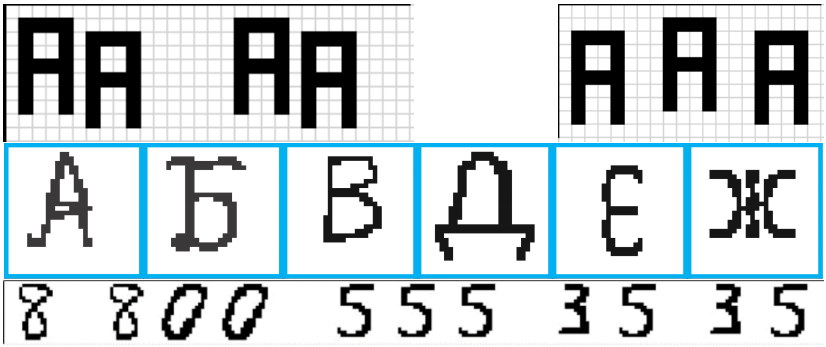

Рис. 4. Примеры идентифицированных образов.

3. Результаты численных исследований

Нашей реализацией были успешно разделены и идентифицированы образы, приведенные на рис. 4.

Классический SVM с линейным классификатором дает среднестатистическую ошибку в 7.52% [1]. Наша реализация на основе той же обучающей выборки дает ошибку ∼4%. Однако стоит отметить, что при смене типа распознаваемых объектов количество ошибок варьируется от 6% до 12%, что в среднем выше, чем стандартная ошибка SVM на новом наборе.

Сепарация образов на основе диаграмм Вороного производилась корректно в 100% испытуемых случаев.

Заключение

Предложенные модификации классического алгоритма SVM, позволили практически вдвое снизить количество ошибок в распознавании. Ценой увеличения точности стало снижение универсальности распознаваемых образов. Однако существенное сокращение времени работы модифицированной версии SVM является ее очевидным плюсом. Заметим, что в случае параллельного программирования время работы метода можно сократить еще больше, что немаловажно при распознавании сложных образов.

Помимо этого в работе был реализован новый подход в определении лексем с помощью диаграмм Вороного, позволяющий за счет строго фиксированного порядка обхода корректно находить и определять строки с содержащимися в них словами.

Список литературы

[1] Лисицын, С. О. Распознавание дорожных знаков с помощью метода опорных векторов и гистограмм ориентированных градиентов / С. О. Лисицын, О. А. Байда // Компьютерная оптика. — 2012. — Т. 36, № 2. — С. 289–294.

[2] Препарата, Ф. Вычислительная геометрия: Введение / Ф. Препарата, М. Шеймос ; пер. с англ. С. А. Вичеса, М. М. Комарова под ред. Ю. М. Баяковского. — М. : Мир, 1989. — 478 с.

[3] Katiyar, G. Off-Line Handwritten Character Recognition System Using Support Vector Machine / G. Katiyar, A. Katiyar, S. Mehfuz // American Journal of Neural Networks and Applications. — 2017. — Vol. 3, № 2. — P. 22–28.

Библиографическая ссылка

Дорофеев, Н. В. Применение аппарата опорных векторов в распознавании образов // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 45–55.

Сведения об авторах

Дорофеев Никита Владиславович
Студент магистратуры
Направление «Прикладная математика и информатика»

УДК 004.9

Атрибуция текста на основе N-грамм

Дуганов И. А.

Тверской государственный университет

Аннотация. В статье рассматривается метод решения задачи атрибуции текста на основе N-грамм. При решении использовались два критерия определения авторства текста, расстояние от профилей и критерий, основанный на непараметрической корреляции Спирмена.

Результаты работы показали, что данный метод хорошо себя показывает, правильно определяя большинство авторов при наличии достаточного количества данных. Наилучшие результаты метод показывает при использовании 2-, 3-, 4-грамм.

Введение

Задача определения авторства текста в настоящее время является достаточно актуальной. Достаточно важным примером, с которым сталкивается каждый студент или научный сотрудник является проверка курсовых или выпускных работ, а также статей на антиплагиат. Вторым интересным примером служит вопрос о причастности заявленных авторов к написанию некоторых произведений. До сих пор не умолкают споры об авторстве М. А. Шолохова по отношению к произведению «Тихий Дон», также ломаются копья в попытках решить «Шекспировский вопрос» и однозначно ответить на вопрос о том писал ли У. Шекспир приписываемые ему произведения или они вышли из-под пера совершенно других людей [1].

Атрибуция текста — это исследование и анализ текста с целью установление подлинного автора.

Методы атрибуции теста, которыми оперируют эксперты в области литературы, основаны на изучении орфографических, пунктуационных, фразеологических стилевых и прочих характеристик присущих конкретному автору. Принятое в этом случае решение зачастую не лишено субъективизма, порожденного опытом конкретного эксперта.

биграммы	триграммы
«те»	«тек»
«ек»	«екс»
«кс»	«ксе»
«ст»	

Таблица 1. N-граммы для слова «текст».

Поэтому востребованы математические модели атрибуции, способные быстро выносить объективное решение на основе статистического анализа больших объемов информации. Такие системы могут стать хорошими помощниками при проведении автороведческой экспертизы.

Одной из таких моделей является метод, основанный на анализе N-грамм [2], поскольку они вполне могут отображать стиль автора, включая частоту использования определенных слов и их связок, речевых оборотов, частей речи и многого другого.

1. Теоретические аспекты атрибуции текста на основе N-грамм

N-грамма — это последовательность из N элементов. Для задачи атрибуции теста за элементы будут приняты символы слов в тексте.

В табл. 1 представлены примеры N-грамм для слова «текст».

Весь текст разбивается на подобные N-граммы, после чего необходимо рассчитать частоты встречаемости каждой из них. По итогам будет получен набор пар (n_i, f_i), где n_i — значение N-граммы, f_i — частота ее встречаемости.

Всю совокупность таких пар для произведений одного автора будем называть профилем автора [2]. С ними в дальнейшем будет производиться сравнение поданного на вход неизвестного текста.

Сравнение будет производиться на основе расстояния между профилями автора и поданного на вход текста, вычисляемого по формуле

$$f = \sum_{i=1}^{n} |f_{i,1} - f_{i,2}|, \tag{1}$$

где $f_{i,1}$ — вектор частот N-грамм профиля автора, $f_{i,2}$ — вектор частот N-грамм текста.

Автор, до которого полученное расстояние будет минимальным, признается наиболее вероятным автором текста.

2. Авторская реализация метода

2.1. Создание базы профилей

Для решения задачи определения авторства необходима информация о профилях авторов, на основе которой в дальнейшем будет производится атрибуция текста неизвестного автора. Для этого была создана база данных, в которую вошли произведения 13 авторов, написанные в прозе на русском языке. Количество произведений для одного автора варьировалось от 10 до 25. Длина произведений при отборе не учитывалась.

Предварительно тексты произведений были обработаны следующим образом:

- убраны примечания, заголовки, теги и иллюстрации;
- убраны лишние пробелы, знаки пунктуации и управляющие символы.

Для каждого произведения автора были рассчитаны N-граммы, частоты и сохранены в виде таблиц в базе данных, которые содержат два поля — значение N-граммы и ее частота. В дальнейшем с ними будут сравниваться профиль поданного на вход текста.

2.2. Расчет расстояния до профилей

Автором данной статьи была создана программа, проводящая сравнение профилей. На вход ей подается неизвестный текст, который преобразуется вышеупомянутым образом. Рассчитываются N-граммы и их частоты.

По формуле (1) вычисляется расстояние до каждого из профилей автора, содержащихся в базе. В том случае если необходимой N-граммы не содержится в профиле автора или профиле тексте, частота принимает значение 0.

		Автор					
		A	B	C	D	E	F
Произведение	1	0.229	0.226	0.229	0.237	0.219	0.259
	2	0.167	0.181	0.168	0.206	0.176	0.225
	3	0.198	0.154	0.192	**0.077**	0.183	0.165
		G	H	I	J	K	L
	1	0.224	**0.166**	0.217	0.352	0.213	0.246
	2	**0.131**	0.213	0.191	0.328	0.160	0.179
	3	0.161	0.165	0.143	0.271	0.145	0.238

Таблица 2. Результаты работы метода для 2-грамм.

2.3. Результаты работы метода

Для апробирования метода на вход программе подавались произведения авторов, которые содержатся в базе данных. При этом сами эти произведения не содержались в профиле автора. Использовались тексты, написанные в прозе на русском языке. Длина произведений варьировалась от 5 тысяч до 50 тысяч символов. В ходе работы было установлено, что для получения наилучших результатов длина текста должна составлять более 17 тысяч символов. Произведения меньшей длины плохо поддаются анализу.

Профиль автора должен состоять из более чем 10–15 произведений общей длиной символов не менее 100 тысяч. Например, в профиле Д. И. Фонвизин присутствовало только 10 произведений, что не всегда позволяло определить принадлежащие ему произведения.

В табл. 2–4 приведены результаты работы алгоритма для трех произведений, которые сравнивались с профилями авторов содержащихся в базе. Наименьшим результатам соответствуют истинные авторы произведений.

Произведения:

- 1 — Война и мир. Том 1. Л. Н. Толстой
- 2 — Пикник на обочине. Братья Стругацкие
- 3 — Преступление и наказание. Ф. М. Достоевский

Авторы:

		Автор					
		A	B	C	D	E	F
Произведение	1	0.474	0.438	0.466	0.446	0.446	0.483
	2	0.396	0.411	0.389	0.435	0.394	0.464
	3	0.417	0.353	0.417	**0.186**	0.395	0.367
		G	H	I	J	K	L
	1	0.462	**0.306**	0.421	0.683	0.411	0.505
	2	**0.301**	0.462	0.427	0.691	0.388	0.426
	3	0.373	0.371	0.323	0.594	0.333	0.492

Таблица 3. Результаты работы метода для 3-грамм.

		Автор					
		A	B	C	D	E	F
Произведение	1	0.749	0.682	0.721	0.672	0.698	0.736
	2	0.699	0.732	0.685	0.738	0.683	0.782
	3	0.666	0.592	0.658	**0.329**	0.633	0.601
		G	H	I	J	K	L
	1	0.731	**0.473**	0.644	1.015	0.649	0.786
	2	**0.546**	0.769	0.738	1.062	0.695	0.746
	3	0.624	0.591	0.535	0.931	0.556	0.784

Таблица 4. Результаты работы метода для 4-грамм.

- A — М. А. Булгаков
- B — Н. В. Гоголь
- C — М. Горький
- D — Ф. М. Достоевский
- E — В. В. Набоков
- F — М. Е. Салтыков-Щедрин
- G — Братья Стругацкие
- H — Л. Н. Толстой
- I — И. В. Тургенев
- J — Д. И. Фонвизин
- K — А. П. Чехов
- L — М. А. Шолохов

2.4. Критерий непараметрической корреляции Спирмена

Заметим, что исходная идея метода N-грамм не позволяет проверить статистическую значимость полученных оценок для меры близости f, поскольку распределение этой статистики неизвестно.

Поэтому в дополнение к расчету расстояния между профилями была произведена оценка непараметрического коэффициента корреляции Спирмена и проверена его статистическая значимость на основании аппроксимаций Имана – Конновера и Фишера – Боннета – Райта.

Алгоритм проверки заключался в следующем.

1) Были взяты n наиболее часто встречающихся N-грамм поданного на вход текста (выборка Y), в профилях всех авторов найдены аналогичные N-граммы (выборка X).
2) По каждому профилю проделаны следующие расчеты.
   а) Выборки X и Y были расположены в порядке возрастания и проранжированы. R_i — ранг наблюдения X_i в ряду X, S_i — ранг наблюдения Y_i в ряду Y.
   б) Вычислена статистика с учетом наличия связей:

$$p_s = \frac{n^3 - n - 6 \cdot \left(\sum\limits_{i=1}^{n} d_i^2 + T_x + T_y \right)}{\sqrt{(n^3 - n - 12 \cdot T_x) + (n^3 - n - 12 \cdot T_y)}}, \quad (2)$$

$$T_x = \frac{1}{12} \sum_{i=1}^{g} (t_i^3 - t_i), \quad (3)$$

$$T_y = \frac{1}{12} \sum_{i=1}^{h} (u_i^3 - u_i), \quad (4)$$

где n — число парных наблюдений, $d_i = R_i - S_i$ — разности рангов для i-й пары наблюдений, g и h — число связанных групп по X и Y, t_i и u_j — объем i-й связанной группы по X и j-й связанной группы по Y, для несвязанных наблюдений $t_i = u_j = 1$.

в) Применена аппроксимация Имана – Канновера:

$$J_p = \frac{p_s}{2} \cdot \left(\sqrt{n-1} + \sqrt{\frac{n-2}{1-p_s^2}} \right), \tag{5}$$

$$|J_p| > J_{\alpha/2} \Rightarrow H_1 : |p_s| \neq 0, \tag{6}$$

где z_α и $t_\alpha(n)$ — верхние $\alpha\%$ точки нормального закона и распределения Стьюдента с n степенями свободы.

г) Применена аппроксимация Фишера – Боннета – Райта:

$$p_s^* = 0.5 \cdot \ln \left(\frac{1+p_s}{1-p_s} \right), \tag{7}$$

$$\sigma^2_{p_S^*} = \frac{1 + 0.5 \cdot (p_S^*)^2}{n-3}, \tag{8}$$

$$z^* = \frac{p_S^*}{\sqrt{\sigma^2_{p_s^*}}}, \tag{9}$$

$$|z^*| > z_{\alpha/2} \Rightarrow H_1 : |p_s| \neq 0, \tag{10}$$

где z_α — верхняя $\alpha\%$ точка нормального закона.

Итоги расчетов представлены в табл. 5–7.

Соответствующие данные были получены на выборках объемов: $m = 2\,500$ для 2-грамм, $m = 1\,200$ для 3-грамм и $m = 500$ для 4-грамм.

Гипотезы H_1 на уровне $\alpha = 0.05$ принимаются на всем наборе профилей, что не позволяет однозначно определить автора. Но по наблюдениям у профиля истинного автора произведения значения критерия всегда значительно больше остальных.

Заключение

По итогам алгоритм показал неплохой результат, правильно распознав большинство авторов в тестовом наборе данных. Для создания профиля автора не важна длина отдельно взятого произведения, важно только общее количество символов.

Для анализируемого произведения длина должна быть от 17 тысяч символов. Чем больше произведений содержит профиль автора

	2-граммы		3-граммы		4-граммы	
Автор	J_p	z^*	J_p	z^*	J_p	z^*
A	58.288	28.687	33.645	28.316	15.592	14.271
B	54.897	28.344	32.115	27.447	14.031	13.065
C	59.593	28.808	33.396	28.178	16.133	14.671
D	52.658	28.089	31.818	27.273	15.962	14.546
E	60.001	28.844	33.991	28.505	14.947	13.781
F	53.323	28.167	29.744	26.003	12.252	11.605
G	**70.667**	**29.629**	**40.857**	**31.781**	**20.495**	**17.559**
H	51.318	27.924	30.316	26.362	13.714	12.811
I	52.662	28.090	32.316	27.564	15.991	14.567
J	37.993	25.548	24.673	22.516	10.919	10.459
K	56.958	28.558	33.484	28.226	15.612	14.286
L	53.446	28.181	32.380	27.601	13.653	12.762

Таблица 5. Значения критерия Спирмена для произведения «Пикник на обочине».

	2-граммы		3-граммы		4-граммы	
Автор	J_p	z^*	J_p	z^*	J_p	z^*
A	33.816	25.021	30.277	26.338	11.355	10.839
B	35.615	25.613	31.268	26.945	11.532	10.992
C	33.568	24.935	29.491	25.842	11.518	10.981
D	42.272	27.359	30.784	26.651	11.262	10.759
E	37.634	26.211	29.625	25.928	11.714	11.149
F	45.058	27.926	27.719	24.676	9.512	9.206
G	33.421	24.883	29.497	25.846	11.333	10.821
H	**61.051**	**30.078**	**42.593**	**32.474**	**16.468**	**14.915**
I	45.261	27.964	31.752	27.234	12.004	11.396
J	37.093	26.057	23.948	21.975	9.806	9.471
K	36.539	25.895	31.344	26.991	12.734	12.009
L	33.249	24.822	28.537	25.222	11.813	11.233

Таблица 6. Значения критерия Спирмена для произведения «Война и мир. Том 1».

	2-граммы		3-граммы		4-граммы	
Автор	J_p	z^*	J_p	z^*	J_p	z^*
A	12.881	12.131	35.566	29.338	60.113	29.753
B	14.929	13.767	37.823	30.445	67.307	30.361
C	12.506	11.819	34.576	28.821	67.791	30.396
D	**24.371**	**19.608**	**56.983**	**36.642**	**82.544**	**31.263**
E	14.287	13.267	35.516	29.312	63.604	30.068
F	13.867	12.934	36.117	29.617	57.812	29.522
G	14.215	13.211	37.994	30.524	61.541	29.887
H	13.954	13.004	36.768	29.941	50.898	28.681
I	16.261	14.764	40.823	31.767	61.847	29.914
J	10.027	9.669	27.547	24.559	41.837	27.088
K	14.752	13.631	39.115	31.034	69.167	30.495
L	9.828	9.491	30.713	26.608	55.978	29.322

Таблица 7. Значения критерия Спирмена для произведения «Преступление и наказание».

и чем больше длина анализируемого текста, тем результат более точен.

Алгоритм показывает наиболее хорошие результаты при использовании 2-, 3-, 4-грамм.

Список литературы

[1] Орехов, Б. Атрибуция текста: теория и практика : [сайт]. — 2019. — 6 июн. — URL: https://postnauka.ru/faq/99046 (дата обращения: 08.05.2022)

[2] Леонова, А. В.Определение авторства текстов на основе подхода N-грамм / А. В. Леонова, И. В. Леонова // Научное обозрение. Технические науки. — 2018. — № 6. — С. 37–40.

Библиографическая ссылка

Дуганов, И. А. Атрибуция текста на основе N-грамм // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ,

2022. — С. 56–65.

Сведения об авторах

Дуганов Илья Алексеевич
Студент магистратуры
Направление «Прикладная математика и информатика»

УДК 519.254

Построение модели ценообразования для реализации подержанных ноутбуков на веб-сайте

Князев Р. Д.

Тверской государственный университет

Аннотация. В данной статье проводится анализ зависимости компонентов и бренда ноутбука от его цены, интерпретация полученной регрессионной модели для подержанных ноутбуков и последующее сравнение полученных моделей. На примере цен ноутбуков собранных из открытых источников за период 2019–2022 года. Приведен пример использования модели на веб-сайте и возможности ее улучшения.

Введение

В современном мире тенденция покупки бытовой техники и электроники склоняется к замене старых моделей на более современные. Многие электроприборы заменяются, не достигнув полный срок эксплуатации. В том числе и ноутбуки. Спрос потребителей и коммерческих фирм на портативную технику только увеличивается в последние годы. Возникает проблема в большом количестве старой устаревшей морально, но все еще рабочей техники. Крупные фирмы решают проблему путем продажи техники в ближайшие учебные учреждения или сферам малого бизнеса, где скорость и производительность техники играет меньшую роль. Но как поступать обычным потребителям? Продать старое устройство им проблематичнее.

В настоящее время существует множество сайтов по скупке техники или сайты-объявления о продаже. Но у всех есть общая проблема в полном отсутствии либо очень поверхностной оценки стоимости техники вне зависимости от параметров.

Таким образом сайт, способный предоставить пользователю информацию о его устройстве и провести качественную оценку цены по параметрам востребован рынком. Важно отметить, что сфера

Asus	59
Lenovo	50
HP	36
Huawei	6
Acer	29
Dell	37
Xiaomi	35
MSI	20
Apple	21

Таблица 1. Структура собранных данных по брендам.

применения модели не заканчивается только сайтом, но и позволит проводить сравнительный анализ одинаковых по цене новых устройств разных производителей и выявлять заметные завышения или занижения цен на ноутбуки.

В рамках данной статьи будет проведено построение регрессионной модели цен на ноутбуки, собранных по данным из открытых источников с помощью пакета анализа данных R и описаны методы получения наиболее точной модели путем построения регрессионных моделей для групп и очистки исходных данных. Такой метод обычно называется гедонической регрессией и проводится для оценки недвижимости или спроса на товар [1]. Для примера использования модели описана процедура интеграции с веб-сайтом.

1. Сбор данных для построения модели

Данные для построения модели взяты из открытых источников путем слепого сбора. В случае, если цены одной модели не совпадали использовалось среднее арифметическое между ценами из разных источников.

Собранные данных загружены в базу данных PostgreSQL. Всего для анализа используются данные о 293 моделях ноутбуков 10 различных брендов (Asus, Lenovo, HP, Huawei, Acer, Dell, Xiaomi, MSI, Apple). Распределение ноутбуков по брендам представлено в табл. 1.

Для прогнозирования на основе собранных данных выбрана модель множественной линейной регрессии вида представленного в

формуле (1), где a_i — регрессионные коэффициенты, b_0 — свободный член, e — член, содержащий ошибку:

$$Y = \sum_{i=1}^{n} a_i x_i + b_0 + e. \tag{1}$$

В данном случае $n = 12$ по числу собранных количественных объясняющих переменных. Используются следующие численные характеристики ноутбуков: диагональ экрана, ширина экрана (в пикселях), длина экрана (в пикселях), частота обновления матрицы (в герцах), объем оперативной памяти (в мегабайтах), частота оперативной памяти (в мегагерцах), объем жесткого диска (в гигабайтах, 0 если отсутствует), объем SSD диска (в гигабайтах, 0 если отсутствует), вес ноутбука (в килограммах), год производства ноутбука. И следующие логические переменные принимающие значения 0 или 1: наличие матового покрытия, наличие дискретной видеокарты.

2. Построение модели

Первичное построение модели сделано в пакете статистики R при помощи метода *lm* [2].

Данные были предварительно подготовлены, очищены от всплесков и убраны записи, по результатам которых после прогнозирования стьюдентизированные остатки были больше 2 по модулю. Оценка остатка рассчитывалась по формуле (2), где e_i — остатки регрессии, h_{ii} — рычаг проекционной матрицы H, $\sigma_{(i)}$ — оценка стандартного отклонения (корень из дисперсии) ошибок модели с исключенным i-м наблюдением.

Оценка дисперсии ошибок модели в таком случае вычислялось по формуле (3) и для $i = 1$ равна 14991.32, где n — количество наблюдений, e_j — ошибка, m — количество оцениваемых параметров модели:

$$r_i = \frac{e_i}{\sigma_{(i)}\sqrt{1 - h_{ii}}}, \tag{2}$$

$$\widehat{\sigma^2_{(i)}} = \frac{1}{n - m - 1} \sum_{j=1,\ j\neq i}^{n} e_j^2. \tag{3}$$

Получили модель, представленную на рис. 1 и формуле (4), где x_1–x_{12} — объясняющие переменные диагональ экрана, ширина экрана, длина экрана, наличие матового покрытия, частота обновления матрицы, объем оперативной памяти, частота оперативной памяти, наличие дискретной видеокарты, объем жесткого диска, объем SSD диска, вес ноутбука, год производства ноутбука соответственно. Неиспользуемые переменные были удалены после проверки их значимости:

$$\begin{aligned} price\,\widehat{db} = 4861854 - 9689.188x_1 - 161.281x_2 + \\ + 299.554x_3 + 3.228x_6 + 8.93x_7 - 16322.24x_8 + 4.893x_9 + \\ + 21.692x_{10} + 13872.17x_{11} - 2351.176x_{12}. \end{aligned} \quad (4)$$

Чтобы убедиться в возможности применения линейной модели, был построен график квантиль-квантиль на рис. 2. И чтобы убедиться в нормальности остатков модели, построена гистограмма остатков на рис. 3. Из которых следует, что линейная модель применима, хотя на концах графика наблюдаются отклонения, а остатки имеют визуально похожее на нормальное распределение.

Рассмотрим статистические показатели модели чтобы убедиться в возможности ее применения на практике. Средняя абсолютная ошибка составляет 11178.03 что в процентах составляет 21.33%. Для модели были вычислены коэффициенты детерминации $R^2 = 0.7852$ и скорректированный $R^2_{adj} = 0.7755$, рассчитанные по формуле (5) и (6) соответственно, где y_i — фактическое значение цены, $\widehat{y}_i$ — расчетное по модели значение цены, $\overline{y}$ — выборочное среднее цен, n — количество наблюдений, k — количество зависимых переменных:

$$R^2 = \frac{\sum\limits_{i=1}^{n}(y_i - \widehat{y}_i)^2}{\sum\limits_{i=1}^{n}(y_i - \overline{y})^2}, \quad (5)$$

$$R^2_{adj} = 1 - (1 - R^2)\frac{n-1}{n-k}. \quad (6)$$

Данный результат не является достаточно точным для использования в прогнозировании цен по ноутбукам. Чтобы получить наиболее точный результат, были удалены неиспользуемые переменные.

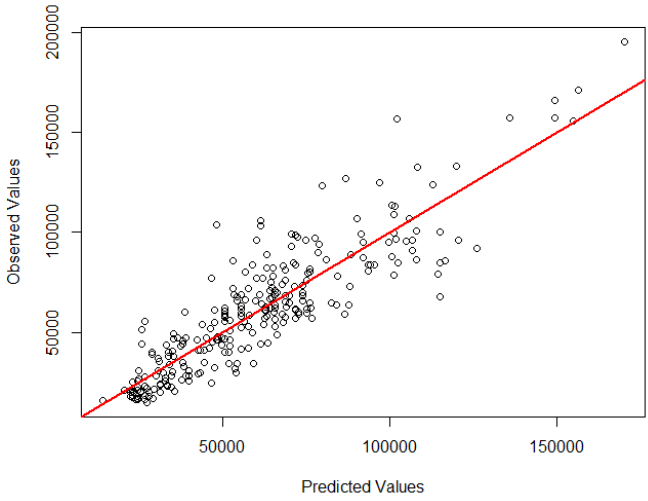

Рис. 1. График «Наблюдаемые-предсказанные значения».

После удаления неиспользуемых переменных их осталось 9 (удалено наличие матового покрытия, частота обновления матрицы, объем жесткого диска) статистика модели практически не изменилась. Средняя абсолютная ошибка теперь составляет 11232.61 что в процентах 21.58%. Коэффициенты детерминации равны $R^2 = 0.7829$ и $R^2_{adj} = 0.7756$. Так как скорректированный коэффициент детерминации стал незначительно ближе к 1, то можно говорить, что модель с очищенными переменными дает результат незначительно лучше.

С практической точки зрения удаление неиспользуемых переменных в модели позволит пользователю задавать меньше параметров устройства. Но заметим, что модель является недостаточно точной и все еще дает ошибку около 20% в цене, что недопустимо.

3. Построение моделей для категорий

Для корректирования полученной модели было произведено разбиение данных по категориям. Первая логичная фильтрация – по брендам устройств. Разбив данные на разные бренды, заметим, что

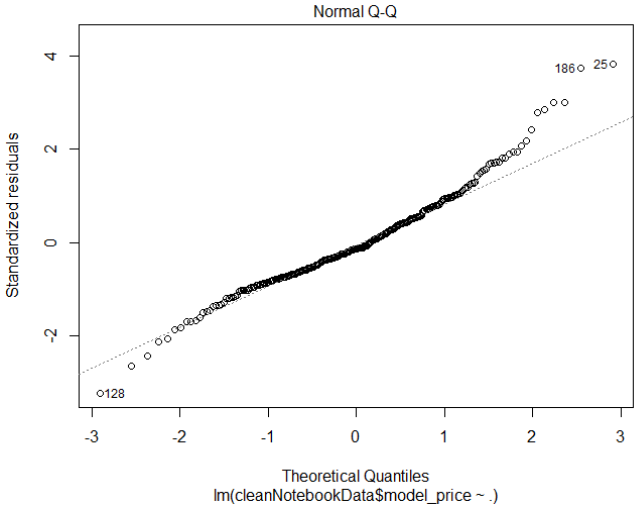

Рис. 2. График QQ-plot для проверки нормальности остатков.

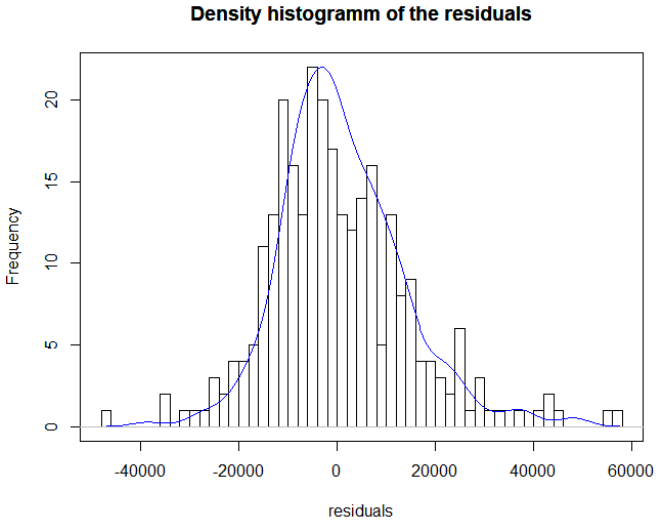

Рис. 3. Гистограмма остатков и эмпирической функции плотности.

Бренд	MAE	MAPE	R^2	R^2_{adj}
Asus	21390.12	26.86%	0.7159	0.6992
Lenovo	4610.52	10.42%	0.9216	0.9106
HP	7662.31	10.77%	0.9124	0.8978
Huawei, Xiaomi	13164.87	12.42%	0.7351	0.6962
Acer	3109.89	10.75%	0.8626	0.8461
Dell	13215.51	14.77%	0.5517	0.5385
MSI	6370.18	8.82%	0.6385	0.5872
Apple	10325.73	11.35%	0.8623	0.8323

Таблица 2. Статистика моделей по брендам.

имеется бренд Huawei с количеством данных 6. Для него модель построить не получится ввиду того, что количество данных меньше объясняющих переменных. Он будет объединен с брендом Xiaomi. Ввиду технического сходства моделей и географического расположения производство такое соединение весьма корректно и может положительно сказаться на результатах модели.

После построения моделей были получены результаты, представленные в табл. 2. Для каждой модели были вычислены коэффициенты R^2 и R^2_{adj} по формулам (5) и (6) соответственно.

Таким образом, для большинства брендов разбиение значительно увеличило точность модели и использование брендовой модели будет предпочтительнее использованию общей. Заметим, что обратная ситуация для брендов Asus, Dell и MSI. У первого огромная ошибка, которая сильно повлияет на прогноз и будет выдавать некорректную цену. Для MSI и Dell наблюдается низкий процент ошибки, но коэффициент детерминации достаточно далек от 1. В случае прогнозирования новых устройств возможна большая ошибка. Это может быть объяснено сравнительно маленьким набором данных, но предпочтительнее использовать общую модель.

4. Использование на веб-сайте

Для тестирования модели разработан сайт купли-продажи подержанных устройств. Особенностью сайта является при желании

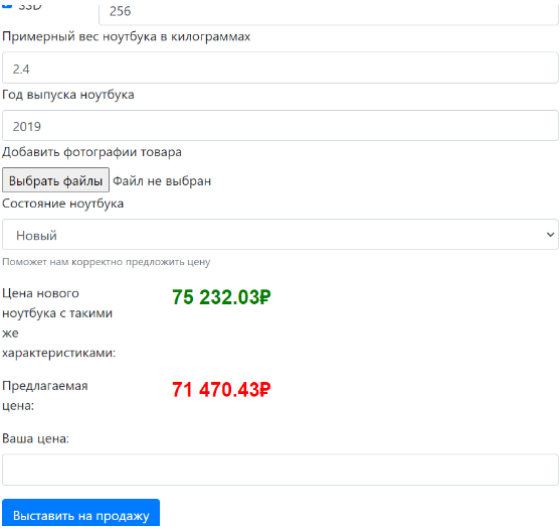

Рис. 4. Пример использования модели на веб-сайте.

пользователя продать устройство выполнять прогнозирование цены товара, как если бы ноутбук был новым.

На основе введенных параметров производится интерпретация и выбор модели регрессии для товара. Например, по бренду либо общая для всех устройств модель если бренд новый или ранее неизвестный. После ввода параметров пользователь получает для информации цену устройства как нового и скорректированную цену подержанного. Корректировка совершается в процентном соотношении в зависимости от состояния. Пример визуализации предсказанной цены для пользователя показан на рис. 4.

В дальнейшем после получения информации об успешных сделках станет возможно по аналогии с новыми ноутбуками провести регрессионный анализ для построения корректной модели подержанных устройств.

С технологической стороны сайт написан на стандартном стеке веб-технологий для клиентской части и с использованием языка Java для серверной части. Интегрирование с моделью происходит по средствам REST. Создана точка для изменения, пересчета и

обновления модели на языке Python, которая при работе вызывает скрипт пакета R.

Заключение

Анализ показывает, что построение модели, способной корректно произвести оценку ноутбука как нового перед продажей подержанного устройства для последующего использования на веб-сайте возможно с приемлемой точностью. Путем фильтрации и построения нескольких моделей, которые при вводе в эксплуатацию могут дополняться данными и корректироваться, возможно описывать цену ноутбука через набор всех или части его характеристик.

Список литературы

[1] Şentürk, İ. Factors Affecting the Notebook Computer Prices in Turkey: A Hedonic Analysis / İ. Şentürk, C. Erdem // The Empirical Economics Letters. — 2010. — Vol. 9, № 6. — P. 545–553.

[2] Faraway, J. J. Linear Models with R. — 2nd ed. — New York : Chapman and Hall/CRC, 2014. — 286 p.

Библиографическая ссылка

Князев, Р. Д. Построение модели ценообразования для реализации подержанных ноутбуков на веб-сайте // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 66–74.

Сведения об авторах

Князев Роман Дмитриевич
Студент магистратуры
Направление «Прикладная информатика»

УДК 004.032.26 + 004.932.4

Заливка с учетом содержимого

Коваленко А. Л.

Тверской государственный университет

Аннотация. Рассматривается задача удаления объектов на изображении так, чтобы это было незаметно для пользователя. В описанной задаче мы знаем какие объекты будем удалять, ввиду обученности нейронной сети на поиск этих объектов заранее. Используется алгоритм заливки изображения на основе его содержимого, который не требует обучения заранее, а обучается в рамках поданного изображения. Проиллюстрированы результаты на различных архитектурах сверточных нейронных сетей. Проведено сравнение результатов по основным показателям.

Введение

Иногда на фотографиях появляются лишние элементы, которые выбиваются из общей картины. Случайно попавшие или сразу незамеченные лишние объекты на фотографии, посторонние люди в кадре, различные дефекты и шумы — все это является объектом негодований обычных пользователей. Чтобы маскировать на изображении дефекты часто нужны навыки и опыт, которых у обычного человека может и не быть. Более того, далеко не все хотят или умеют пользоваться редакторами изображений такими как Adobe Photoshop и подобными, а затрачивать время на обучение не видят возможным. Это является поводом для того, чтобы исследовать методы и развивать инструментарий для решения таких задач. В данной статье предлагается комплексное решение одной из подпроблем, а именно удаления людей с изображения.

1. Сверточные нейронные сети

Наиболее часто изображения являются многоканальными, где каждый канал представляет матрицу, описывающую монохромную картинку модели RGB. При использовании таких входов в нейросеть

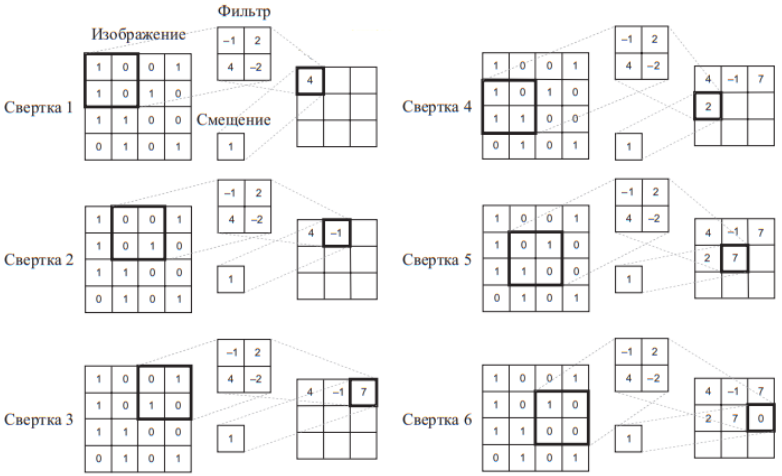

Рис. 1. Свертка.

с прямой связью может возникнуть проблема длительной работы из-за большого количества вычислительных действий.

Чтобы упростить вычисления в задачах с обработкой изображений были придуманы сверточные нейронные сети. Сверточные нейронные сети — это модели, которые значительно уменьшают количество параметров в глубокой нейронной сети. В математическом смысле сверточные сети обеспечивают инструменты для эффективного использования локальной структуры данных [1]. При этом модель практически не теряет в качестве, а работа и обучение сети заметно ускоряются. СНС прозвали так именно из-за операции свертки, которая сокращает количество входных параметров. Слой с операцией свертки называется сверточным слоем. Сверточный фильтр скользит (рис.1) по изображению и умножается на выделенную часть изображения. Это позволяет выделять признаки, формируя новый вход для следующего слоя.

Подвыборка или пулинг (рис. 2) — это вторая разновидность слоя в данном типе нейронной сети. Ее действие схоже со сверткой ввиду того, что также происходит скольжение по изображению. Только здесь не используется умножение на фильтр, а выбирается

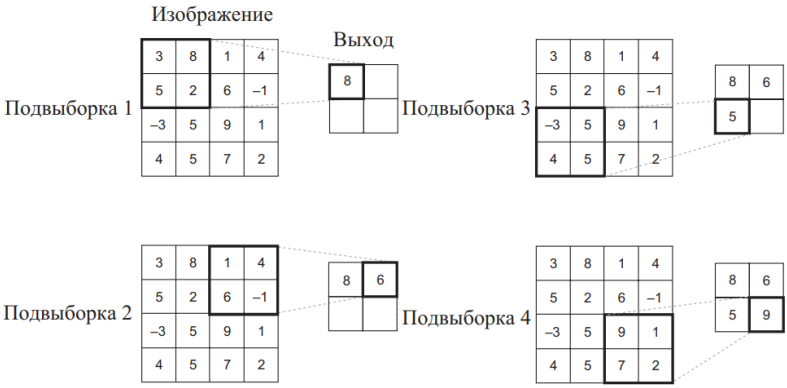

Рис. 2. Подвыборки.

из выделенной части изображения максимальное значение (иногда считается среднее).

За счет комбинирования этих двух типов слоев, а также фильтров и используемых функций, строятся сверточные нейронные сети различных архитектур.

2. Распознавание объектов на изображении с помощью YOLO

Первая часть приложения состоит из обнаружения объектов и получения ограничивающего прямоугольника его позиции. Для этого была использована одна из самых быстрых нейронные сетей, а именно YOLO [2]. Явным преимуществом над другими методами распознавания является то, что в других методах процесс подачи изображения через сеть происходит многократно, а здесь требуется «один раз взглянуть». Изображение разбивается на сетку размера $N \times N$.

Важно упомянуть, что такое ограничивающая рамка. Ограничивающая рамка — это прямоугольная рамка вокруг изображения, которая выделяет часть, где есть требуемый объект.

За каждой из клеток зафиксирован ряд параметров (рис. 4). Каждая клетка отвечает за прогнозирование количества ограничи-

Рис. 3. Разбиение изображения.

Рис. 4. Параметры ячейки.

вающих рамок и значений, показывающих с какой вероятностью тот или иной объект присутствует в рамке.

- t_x, t_y — координаты центра ограничивающей рамки, относительно той ячейки, в которой она находится;
- t_w, t_h — ширина и высота ограничивающей рамки;
- p_0 — индекс объектности, показывает вероятность успешного нахождения объекта в рамке;
- $p_1, p_2, \ldots, p_c$ — вероятность принадлежности объекта к классу предполагаемых.

Именно этот набор параметров помогает алгоритму точно обнаружить объект.

3. Вписывание части изображения

Восстановление изображения — это процесс замены поврежденной части изображения на реалистичные фрагменты. Deep Image Prior [3] — метод восстановления изображения использующий глубокое обучение и не требующий данных для тренировки. Сеть учится извлекать полезную информацию только с поданного на вход изображения. Сформулируем задачу как задачу минимизации [3]:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\ E(f_\theta(z), x_0), \quad x^* = f_{\theta^*}(z)$$

где $E(x, x_0)$ — это функция потерь, зависящая от решаемой задачи, а $f_\theta(z)$ — некоторая сверточная сеть.

Алгоритм решения выглядит следующим образом.

1) Инициализируем θ случайными весами.
2) На каждой итерации:
   а) сеть f с текущими весами θ получает на вход фиксированный тензор z и возвращает восстановленное изображение x;
   б) с помощью сгенерированного изображения x и исходного изображения x_0 вычисляется функция потерь $E(x, x0)$;
   в) веса θ обновляются так, чтобы минимизировать g:

$$g = \theta^* - \underset{\theta}{\operatorname{argmin}}\ E(f_\theta(z), x_0).$$

На рис. 5 показан пример работы данного алгоритма.

4. Полученные результаты

Объеденив все вышеописанные пункты, мы получаем систему, способную обнаруживать и маскировать обнаруженные объекты на

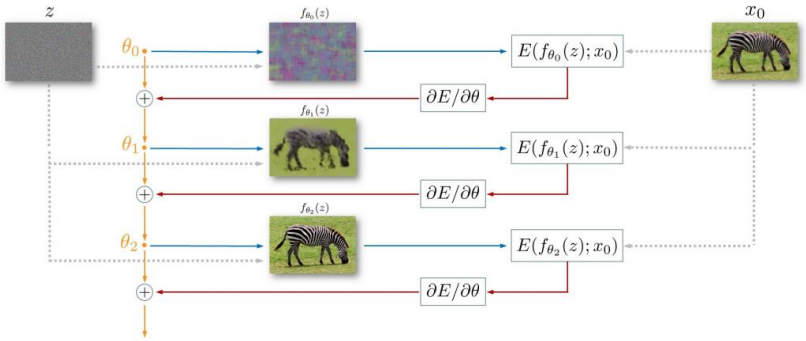

Рис. 5. Визуализация работы алгоритма [3].

Рис. 6. Оригинальное и входное изображения.

изображении. Она не требует большого объема данных для обучения заливке, а следовательно может работать над любым изображением.

Следующие архитектуры сверточных нейронных сетей были использованы в сравнении: ResNet, SkipNet, U-Net. На вход сети получали изображения (рис. 6) одинакового размера 720×480 с уже найденным и залитым объектом. Количество эпох в экспериментах с различными сетями было равно 3000. Во время обучения использовался оптимизатор Adam с коэффициентом обучения 0.001.

ResNet. На выходном изображении получилось высокое качество изображения, однако заметны артефакты в виде лоскута из текстур входной фотографии (рис. 7, 8). Обработка изображения заняла 32 минуты, что является плохим результатом и определенно требует

Рис. 7. Процесс работы.

Рис. 8. Сравнение оригинала и результата при работе с сетью ResNet.

доработки. Сеть с задачей не справилась.

SkipNet. На выходе имеем четкое окружение, однако в месте заливаемого объекта заметны артефакты в виде полос (рис. 9, 10). Большой прирост скорости вычисления в сравнении с ResNet. Итоговое время работы 8 минут и 11 секунд. Можно сказать, что сеть с задачей не справилась.

U-Net. Невооруженным глазом сложно отличить оригинал от выходного изображения (рис. 11, 12). Сеть с задачей справилась. Итоговое время работы 7 минут.

Стоит отметить, что рассматриваемые модели не являются оптимальными, но проведеные эксперименты позволяют продемонстрировать результаты применения тех или иных архитектур на качество

Рис. 9. Процесс работы.

Рис. 10. Сравнение оригинала и результата при работе с сетью SkipNet.

работы системы.

Заключение

В процессе работы были затронуты нейронные сети, один из наиболее быстрых методов обнаружения объектов на изображении и протестированы различные архитектуры для заливки с учетом содержимого на изображении.

Созданная система способна находить и заливать объекты с разной долей качества. Видно, что такие сети как ResNet и SkipNet создают артефакты, однако окружение выглядит более четко. В

Рис. 11. Процесс работы.

Рис. 12. Сравнение оригинала и результата при работе с сетью U-Net.

свою очередь, U-Net справляется с непосредственно поставленной задачей лучше, однако окружение имеет чуть более низкую резкость. Дальнейшая работа в этой области предполагает уменьшение показателей времени, минимизацию появления артефактов и улучшение качества выходного изображения. Исследования в поиске и подборе оптимальных архитектур, моделей и параметров продолжаются.

Список литературы

[1] Рамсундар, Б. TensorFlow для глубокого обучения / Б. Рамсундар, Р. Б. Заде ; пер. с англ. А. В. Логунова — СПб. : БХВ-Петербург, 2020. — 256 с.: ил.

[2] Redmon, J. YOLOv3: An Incremental Improvement / J. Redmon, A. Farhadi. — 2018. — URL: https://arxiv.org/abs/1804.02767 — Загл. с титул. экрана.

[3] Ulyanov, D. Deep Image Prior / D. Ulyanov, A. Vedaldi, V. Lempitsky // International Journal of Computer Vision. — 2020. — Vol. 128. — P. 1867–1888.

Библиографическая ссылка

Коваленко, А. Л. Заливка с учетом содержимого // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 75–84.

Сведения об авторах

Коваленко Антон Леонидович
Студент магистратуры
Направление «Фундаментальная информатика и информационные технологии»

УДК 330.322 + 519.832.3

Модель диверсификации многопериодных инвестиций по достигаемому уровню дохода

Колпакова Д. В.

Тверской государственный университет

Аннотация. Исследуется задача диверсификации инвестиций в многопериодные проекты по достигаемому уровню дохода в условиях неопределенности процентных ставок. Актуальность темы обуславливается слабо развитым инструментарием для работы с многопериодными инвестиционными проектами, ориентированностью известных портфельных моделей в основном на фондовый рынок, а также тем, что используемые методы описания неопределенности в известных моделях в общем случае неадекватны реальности. Предложен подход к решению задачи диверсификации инвестиций на основе ее представления моделью антагонистической игры. Приведен алгоритм решения задачи в предлагаемой постановке. Применение модели иллюстрируется на примере решения тестовой задачи.

Введение

В работе рассматривается задача диверсификации инвестиционных вложений в многопериодные инвестиционные проекты в условиях неопределенности процентных ставок по дополняющему инвестированию и дополняющему заимствованию. В качестве показателя эффективности проектов используется достигаемый уровень изъятий (на текущее потребление).

Актуальность работы обусловлена следующими аспектами: во-первых, инструментарий для работы с многопериодными инвестиционными проектами в рассматриваемых условиях недостаточно развит; во-вторых, в известных методах используемые модели неопределенности в общем случае неадекватны реальности; в-третьих, известные портфельные модели ориентированы, как правило, на фондовый рынок.

Целью работы является повышение обоснованности формирования портфеля многопериодных инвестиций по показателю достигаемого уровня дохода на основе представления данной задачи моделью антагонистической игры.

В работе обосновывается возможность представления рассматриваемой задачи как конфликта между инвестором и рынком капитала, который моделируется бесконечной антагонистической игрой, а также предложены алгоритмы формирования компонент игровой модели и приводятся соотношения для сведения игры к паре двойственных задач линейного программирования. При этом для описания неопределенности используются интервальные оценки прогнозных процентных ставок по инвестированию и заимствованию в плановые периоды реализации проектов. Диапазоны указанных прогнозных оценок определяют стратегии рынка капитала.

Теоретическая значимость полученных результатов определяется развитием моделей портфельного анализа многопериодных инвестиций в части использования более адекватной для практики модели неопределенности и определения диверсификации инвестиций на основе решения антагонистической игры.

1. Математическая постановка задачи

Пусть $I = \{1, 2, \ldots, N\}$ — множество многопериодных инвестиционных проектов.

Рассмотрим задачу формирования портфеля

$$x = (x_1, x_2, \ldots, x_N),$$

$$x \in X = \left\{ x \in R^N \mid \sum_{i=1}^{N} x_i = 1, \quad x_i \geqslant 0, \quad i = 1, \ldots, N \right\}$$

многопериодных инвестиционных проектов, которая сводится к определению долей x_i вложений инвестиционного капитала в анализируемые проекты $i \in I$. Вектор $x = (x_1, x_2, \ldots, x_N)$ определяет диверсификацию инвестиционных вложений между проектами из множества I [1].

Исходными данными, одинаковыми для всех проектов, являются [1]:

- плановый инвестиционный горизонт T с периодами $t = 0, \ldots, T$ реализации проектов;
- требуемая остаточная стоимость C_T^*;
- вектор $f = (f_0, f_1, \ldots, f_T)$, определяющий структуру изъятий на потребление по периодам;
- прогнозные диапазоны $[h_t^{\text{н}}, h_t^{\text{в}}], [s_t^{\text{н}}, s_t^{\text{в}}]$ возможных значений h_t и s_t процентных ставок соответственно по дополняющему инвестированию и дополняющему заимствованию $t = 1, \ldots, T$;
- базовые платежи M_t инвестора, $t = 0, 1, \ldots, T$.

Индивидуальными данными для каждого проекта $i \in I$ являются потоки инвестиционных платежей $z_t(i), t = 0, 1, \ldots, T$.

Для соблюдения условий несовершенного рынка капитала полагаем, что $s_t^{\text{н}} > h_t^{\text{в}}$, $t = 1, \ldots, T$.

Взаимодействие инвестора и рынка капитала будем рассматривать как антагонистический конфликт, в котором возможные действия инвестора — выбор инвестиционного проекта $i \in I$, а возможные действия рынка капитала — выбор пары $q = \{h, s\}$, где

$$h = \{h_1, h_2, \ldots, h_T\},$$
$$s = \{s_1, s_2, \ldots, s_T\},$$
$$h_t \in [h_t^{\text{н}}, h_t^{\text{в}}],\ s_t \in [s_t^{\text{н}}, s_t^{\text{в}}],\ t = 1, \ldots, T.$$

Обозначим множество возможных действий рынка капитала через K. Полагаем, что выбор своих действий стороны осуществляют независимо друг от друга. При этом полезность ситуации $(i, q) \in I \times K$ для инвестора будем оценивать достигаемым уровнем $Y(i, q)$ изъятия средств (из денежного потока проекта) на текущее потребление.

Рассматриваемый конфликт может моделироваться бесконечной антагонистической игрой [3]

$$G = \{I, K, Y\}, \tag{1}$$

где первым игроком является инвестор с конечным множеством I чистых стратегий, вторым игроком — рынок капитала с бесконечным

множеством K чистых стратегий [3], $Y : I \times K \to R^1$ — функция выигрыша инвестора, определяющая его выигрыши в ситуациях $(i, q) \in I \times K$.

Пусть $\{\alpha^*, \beta^*, v\}$ — решение смешанного расширения игры G [3], где $\alpha^* \in A$, $\beta^* \in B$ — оптимальные стратегии игроков 1, 2 соответственно, A, B — множества смешанных стратегий игроков, v — значение игры. Смешанные стратегии инвестора (игрока 1) допускают содержательную интерпретацию в терминах компонент вектора $x \in X$, определяющего некоторую диверсификацию инвестиционных вложений, а именно вероятности α_i выбора проекта $i \in I$ согласно стратегии $\alpha = (\alpha_1, \ldots, \alpha_N) \in A$ инвестора могут трактоваться как доли x_i инвестиционных вложений в проекты $i \in I$. Последнее означает, что оптимальная стратегия $\alpha^* = (\alpha_1^*, \ldots, \alpha_N^*)$ инвестора в смешанном расширении игры G определяет оптимальную диверсификацию $x^* = (x_1^*, \ldots, x_N^*) \in X$ инвестиционных вложений в рассматриваемой задаче формирования портфеля многопериодных инвестиционных проектов.

2. Метод аппроксимации множества стратегий рынка капитала

В качестве метода решения игры (1) используем ее аппроксимацию конечной антагонистической игрой. Для перехода к конечному множеству стратегий рынка капитала применим упрощенный метод, который назовем процедурой пропорционального сдвига. Компоненты первой стратегии ($k = 1$) на t-м периоде формируются как пары $(s_t^{\text{н}}, h_t^{\text{н}})$ нижних значений ставок по дополняющему заимствованию и дополняющему инвестированию, а компоненты стратегий $k = 2, \ldots, m$ определяются с помощью сдвига ставок в периоды t на величины $\Delta_t(s)$ и $\Delta_t(h)$ соответственно:

$$\Delta_t(s) = (k-1)\frac{s_t^{\text{в}} - s_t^{\text{н}}}{m-1},\ k = 2, \ldots, m, \tag{2}$$

$$\Delta_t(h) = (k-1)\frac{h_t^{\text{в}} - h_t^{\text{н}}}{m-1},\ k = 2, \ldots, m. \tag{3}$$

Результатом приведенной аппроксимации является конечная антагонистическая игра, в которой функция выигрыша представляется матрицей $Y = \|Y(i,k)\|$, $i = 1, \ldots, N$; $k = 1, \ldots, m$.

3. Формирование матрицы выигрышей

Обозначим через $Y \in [0, C_T^*]$ некоторое возможное значение функции выигрыша, а через $C_t(i)$ значение остаточной стоимости при таком Y, достигаемое проектом $i \in I$ к моменту времени $t = 0, 1, \ldots, T$. Алгоритм оценки остаточной стоимости $C_T(i)$ на момент завершения реализации проекта определяется следующими соотношениями [1]:

при $t = 0$

$$C_t(i) = M_t - f_t Y + z_t(i),$$

при $t \geqslant 1$

$$C_t(i) = \begin{cases} M_t - f_t Y + z_t(i) + (1 + h_t)\, C_{t-1}(i) & \text{при } C_{t-1}(i) > 0, \\ M_t - f_t Y + z_t(i) + (1 + s_t)\, C_{t-1}(i) & \text{при } C_{t-1}(i) < 0. \end{cases} \quad (4)$$

В первом из соотношений (4) осуществляется дополняющее инвестирование по ставке h_t, во втором — дополняющее заимствование по ставке s_t. Указанные ставки определяются конкретной стратегией рынка капитала. Величина $f_t Y$ в (4) определяет изъятия (доход) на текущее потребление в момент $t = 0, 1, 2, \ldots, T$. При $t = T$ получаем искомую величину $C_T(i)$.

Задача формирования матрицы выигрышей заключается в оценке для каждой возможной ситуации (i, k) такого уровня изъятий $Y(i, k)$, который обеспечивает достижение с требуемой точностью $\varepsilon > 0$ значения C_T^* остаточной стоимости. С учетом убывания остаточной стоимости $C_T(i)$ по аргументу Y, требуемое значение $Y(i, k)$ может быть найдено с использованием метода деления интервала $[0, C_T^*]$ пополам.

Пусть заданы ситуация (i, k) и точность $\varepsilon > 0$ достижения уровня C_T^* остаточной стоимости. Приведем алгоритм оценки элемента $Y(i, k)$ матрицы выигрышей.

1) Положить $a = 0$, $b = C_T^*$.
2) Положить $Y = \dfrac{a + b}{2}$.
3) Вычислить $C_T(i)$ с использованием соотношений (4).
4) Если $C_T(i) < C_T^*$, то положить $b = Y$ и перейти на шаг 5.
   Если $C_T(i) > C_T^*$, то положить $a = Y$ и перейти на шаг 5.
   Если $C_T(i) = C_T^*$, то перейти на шаг 6.

Момент времени t	0	1	2	3
Диапазоны прогнозных ставок h_t		$[0.04, 0.08]$	$[0.03, 0.07]$	$[0.05, 0.09]$
Диапазоны прогнозных ставок s_t		$[0.09, 0.13]$	$[0.08, 0.12]$	$[0.10, 0.14]$
Инвестиционные платежи:				
Проект 1, $z_t(1)$	-500	-400	800	400
Проект 2, $z_t(2)$	-320	-800	1100	200
Проект 3, $z_t(2)$	-900	800	360	-10
Базовые платежи M_t	600	100	-200	800
Структура изъятий f_t	1	1	1.2	1.5

Таблица 1.

5) Если $|C_T(i) - C_T^*| \leqslant \varepsilon$, то перейти на шаг 6, иначе — на шаг 2.

6) Положить $Y(i,k) = Y$. Завершить работу алгоритма.

По матрице выигрышей

$$Y = \begin{pmatrix} Y(1,1) & \dots & Y(1,m) \\ \dots\dots\dots\dots\dots\dots\dots \\ Y(N,1) & \dots & Y(N,m) \end{pmatrix} \tag{5}$$

решение игры находим путем ее сведения к паре двойственных задач линейного программирования [2].

4. Результаты применения модели

Для демонстрации применения предложенной модели рассмотрим задачу диверсификации многопериодных инвестиций на множестве $I = \{1, 2, 3\}$ трехпериодных проектов (см. табл. 1).

Требуемый уровень остаточной стоимости $C_T^* = 500$, $m = 3$, $\varepsilon = 30$.

Процедура пропорционального сдвига для $m = 3$ приводит к следующему аппроксимирующему множеству стратегий рынка капитала:

- стратегия №1: [0.04, 0.09], [0.03, 0.08], [0.05, 0.10];
- стратегия №2: [0.06, 0.11], [0.05, 0.10], [0.07, 0.12];
- стратегия №3: [0.08, 0.13], [0.07, 0.12], [0.09, 0.14].

Результатом решения игры являются оптимальные стратегии:

$$\alpha^* = (0.00, 0.59, 0.41); \quad \beta^* = (0.28, 0.00, 0.72).$$

Таким образом, оптимальная диверсификация инвестиционных вложений для данного примера формирования инвестиционного портфеля определяется вектором $x^* = (0, 0.59, 0.41)$, то есть для инвестора выгоднее всего вложить 59% своего инвестиционного фонда в Проект 2 и 41% инвестиционного фонда — в Проект 3. Для конфликтующей стороны (рынка капитала) стратегия β^* указывает на то, что рациональнее всего установить процентные ставки по инвестированию и заимствованию, следуя стратегии 3 (с вероятностью 0.72), и с вероятностью 0.28, следуя стратегии 1. Стратегия 2 установления процентных ставок для рынка капитала нерациональна.

Заключение

Предложенная модель обеспечивает повышение обоснованности диверсификации инвестиций в многопериодные инвестиционные проекты за счет использования интервальных экспертных оценок поперiодных процентных ставок по инвестированию и заимствованию. Предложенная теоретико-игровая модель задачи диверсификации позволяет найти (при возникающей неопределенности по процентным ставкам) равновесную диверсификацию в смысле невыгодности отклоняться от соответствующих стратегий ни инвестору, ни рынку капитала. Перспективным развитием предложенной модели является исследование возможности использования модели бескоалиционной (неантагонистической) игры для решения задачи портфельного анализа многопериодных инвестиций.

Список литературы

[1] Крушвиц, Л. Инвестиционные расчеты : учебник для вузов. — СПб. : Питер, 2001. — 414 с.

[2] Лубенцова, В. С. Применение линейного программирования в теории игр: Методические указания для самостоятельной работы. — Самара : Самарский государственный технический университет, 2011. — 27 с.

[3] Мазалов, В. В. Математическая теория игр и приложения. — СПб. : Издательство Лань, 2010. — 446 с.

Библиографическая ссылка

Колпакова, Д. В. Модель диверсификации многопериодных инвестиций по достигаемому уровню дохода // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 85–92.

Сведения об авторах

Колпакова Дарья Васильевна
Студентка магистратуры
Направление «Прикладная математика и информатика»

УДК 330.322 + 519.685

Композиционная модель выбора многопериодных инвестиций при неопределенности

Крутелева Е. А.

Тверской государственный университет

Аннотация. В статье рассматривается задача выбора лучшего многопериодного проекта в условиях неопределенности процентных ставок по дополняющим заимствованиям и по дополняющим инвестициям. Приводится обоснование и алгоритм построения композиционной модели, состоящей из модели максимума ожидаемой остаточной стоимости и максимума гарантированного значения остаточной стоимости. Применение модели демонстрируется на тестовом примере.

Введение

Одной из основных проблем в задачах анализа многопериодных инвестиционных проектов является неопределенность прогнозных процентных ставок по инвестированию и по заимствованию.

Целью работы является выбор лучшего многопериодного инвестиционного проекта в условиях неопределенности относительно процентных ставок путем составления композиционной модели, основанной на моделях максимума ожидаемой остаточной стоимости и максимума гарантированного значения остаточной стоимости.

В качестве показателя эффективности проектов используется остаточная стоимость. В основе работы лежит генерирование процентных ставок по инвестированию и заимствованию в каждом периоде.

Выбор предпочтительного проекта определяется максимумом композиционной модели.

1. Постановка задачи

Рассматривается множество $I = \{1, 2, \ldots, N\}$ многопериодных инвестиционных проектов с инвестиционным горизонтом T и плановыми периодами $t = 0, 1, \ldots, T$ при несовершенном рынке капитала.

Исходные данные по проектам определяются следующей информацией:

1) прогнозными диапазонами $s_t \in [\underline{s}_t, \overline{s}_t]$, $h_t \in [\underline{h}_t, \overline{h}_t]$ процентных ставок соответственно по дополняющим заимствованиям и по дополняющим инвестициям, $t = 1, 2, \ldots, T$. Процентные ставки являются случайными величинами;

2) M_t — базовыми платежами в моменты $t = 0, 1, \ldots, T$;

3) заданным уровнем Y изъятий (дохода) с определенной структурой $f = (f_0, f_1, \ldots, f_T)$ изъятий, при которой изымаемый доход в момент времени t равен $f_t Y$;

4) лимитом заимствования G (при неограниченном рынке капитала $G = \infty$);

5) потоком платежей $z_t(i)$, порождаемых инвестиционными проектами $i \in I$ в моменты времени $t = 0, 1, \ldots, T$.

Данные по пунктам 1)–4) одинаковые для всех инвестиционных проектов. Данные по пункту 5) — это индивидуальная информация для каждого отдельного проекта.

В качестве показателя эффективности проектов будем использовать остаточную стоимость $\widetilde{C}_T(i)$ на конец периода T. Случайность процентных ставок обуславливает случайность остаточной стоимости. При этом полагаем, что распределение остаточной стоимости (для любого проекта $i \in I$) не известно. Более того, статистической информации для достоверной идентификации распределения недостаточно. Такая ситуация соответствует неполной информированности о распределении последствий реализации проектов, которые (последствия) измеряются остаточной стоимостью, достигаемой проектами. Множество возможных условий реализации проектов определяется множеством Г сценариев поведения рынка капитала по установке процентных ставок по инвестированию и

по заимствованию в плановые периоды реализации проектов. Отдельный сценарий $\gamma = \{(h_1, s_1), (h_2, s_2), \ldots, (h_T, s_T)\}$ представляет собой возможную реализацию последовательности процентных ставок, каждый элемент которой — это возможное значение пары $(h_t, s_t) \in \left[\underline{h}_t, \overline{h}_t\right] \times [\underline{s}_t, \overline{s}_t]$, $t = 1, 2, \ldots, T$. Будем обозначать далее реализацию остаточной стоимости для проекта $i \in I$ при конкретном сценарии $\gamma \in \Gamma$ через $C_T(i, \gamma)$.

При перечисленных положениях и исходных данных в качестве модели выбора лучшего многопериодного проекта примем модель, являющуюся композицией модели максимума ожидаемой остаточной стоимости и максимума гарантированного значения остаточной стоимости. Целевая функция в такой модели имеет вид

$$\varphi(i) = \lambda\varphi_1(i) + (1 - \lambda)\varphi_2(i), \tag{1}$$

где $\varphi_1(i) = M\{C_T(i, \gamma), \gamma \in \Gamma\}$ — математическое ожидание остаточной стоимости (ожидаемая остаточная стоимость) по проекту $i \in I$ в предположении, что оно с вероятностью $\lambda \in [0, 1]$ является параметром конкретного распределения остаточной стоимости:

$$\varphi_2(i) = \min_{\gamma \in \Gamma} C_T(i, \gamma).$$

Задача заключается в выборе проекта, доставляющего максимум функции (1):

$$\varphi(i) \to \max_{i \in I}. \tag{2}$$

Решением задачи будет являться инвестиционный проект

$$i^* = \arg\max_{i \in I} \varphi(i). \tag{3}$$

Заметим, что при $\lambda = 1$ в (1) получим целевую функцию модели максимума ожидаемой остаточной стоимости, а при $\lambda = 0$ — целевую функцию модели Вальда [1].

Для решения задачи (1)–(3) необходимо:

1) конкретизировать вероятностное распределение процентных ставок;
2) конкретизировать метод оценки реализаций остаточной стоимости $C_T(i, \gamma)$ и функций $\varphi_1(i)$, $\varphi_2(i)$ в (1);

3) определить методы оценки параметра λ;

4) провести численную аттестацию комплексного алгоритма решения задачи.

2. Комплексный алгоритм выбора многопериодных инвестиционных проектов

В силу отсутствия исходной информации о вероятностном распределении процентных ставок, целесообразно принять распределение, обладающее максимальной энтропией [2].

Известно, что таким распределением является равномерное распределение вероятностей [2].

При этом для соблюдения условия несовершенного рынка капитала полагаем $\underline{s}_t > \overline{h}_t$, $t = 0, 1, \ldots, T$.

Получение реализаций остаточной стоимости $\widetilde{C}_T^{(l)}(i)$ по каждому проекту, основано на генерировании реализаций процентных ставок по инвестированию и заимствованию в каждом периоде $t = 0, 1, \ldots, T$.

Отдельная реализация процентных ставок определяет сценарий $\gamma \in \Gamma$. Для конкретного сценария γ, алгоритм вычисления остаточной стоимости определяется следующими соотношениями [1].

При $t = 0$ остаточная стоимость равна

$$C(i, 0) = M_0 - f_0 Y + z_0(i).$$

Далее, для $t > 0$ остаточная стоимость определяется формулой (4):

$$\begin{cases} C_t(i) = M_t - f_t Y + z_t(i) + (1 + h_t)\, C_{t-1}(i) & \text{при } C_{t-1}(i) > 0, \\ C_t(i) = M_t - f_t Y + z_t(i) + (1 + s_t)\, C_{t-1}(i) & \text{при } C_{t-1}(i) < 0. \end{cases} \tag{4}$$

Первое соотношение соответствует дополняющему инвестированию по ставке h_t. Второе соотношение — дополняющему заимствованию по ставке s_t.

Если возникает ситуация, при которой в какой-то момент времени планового периода необходимо осуществить дополняющее заимствование, которое превышает лимит $-\widetilde{C}_t(i) > G$, то проект неосуществим.

Оценка ожидаемой остаточной стоимости (то есть $\varphi_1(i)$) осуществляется с использованием метода математической статистики [2]:

$$\varphi_1(i) = M\left\{C_T(i,\gamma), \gamma \in \Gamma\right\} = \frac{1}{L}\sum_{l=1}^{L} C_T^{(l)}(i). \tag{5}$$

Требуемое число реализаций в (5) определяется соотношением [1]:

$$L \geqslant \frac{t_\varepsilon \sigma^2}{\Delta^2},$$

где

- t_ε — пороговое значение случайной величины с распределение Стьюдента с $(L-1)$степенью свободы при допустимой вероятности $\varepsilon > 0$ ошибки;
- Δ — точность оценки математического ожидания;
- $\sigma^2(i) = \dfrac{1}{L}\sum\limits_{l=1}^{L}\left(C_T^{(l)}(i) - M\left\{C_T(i,\gamma)\right\}\right)^2$.

Нетрудно показать, что

$$\varphi_2(i) = \min_{\gamma\in\Gamma} C_T(i,\gamma) = C_T\left(i,\gamma^0\right),$$

где сценарий γ^0 определяется процентными ставками по инвестированию равными $\underline{h}_1,\ldots,\underline{h}_T$ и процентными ставками по заимствованию $\overline{s}_1,\ldots,\overline{s}_T$.

Для определения параметра λ по сформированной выборке, проверяем гипотезу о равномерности распределения с помощью критерия χ^2.

3. Применение модели

Рассмотрим задачу выбора многопериодного инвестиционного проекта из множества $I = \{1,2,3\}$ исходные данные, по которым представлены в табл. 1. Плановый горизонт $T = 3$. Следовательно,

$$i^* = \arg\max_{i\in I} \varphi(i) = 1.$$

Момент времени t	0	1	2	3
Диапазоны процентных ставок по заимствованию s_t		$[0.05; 0.20]$	$[0.03; 0.18]$	$[0.03; 0.18]$
Диапазоны процентных ставок по инвестированию h_t		$[0.02; 0.08]$	$[0.03; 0.12]$	$[0.03; 0.12]$
Проект A, $z_{t,A}$	-500	-400	800	400
Проект B, $z_{t,B}$	-300	-800	1200	200
Проект C, $z_{t,C}$	-900	800	360	-10
Базовые платежи M_t	600	100	-200	800
Изъятия $f_t Y$	20	22	24	26

Таблица 1. Исходные данные.

Параметр / Проект	$\varphi_1(i)$	$\varphi_2(i)$	λ	$\varphi(i)$
1	1511.51	1475.1	0.6	1496.94
2	1517.12	1448.88	0.6	1489.82
3	1502.09	1428.16	0.55	1468.82

Таблица 2. Результаты.

Результаты применения рассмотренного комплексного алгоритма приведены в табл. 2.

По результатам лучшим инвестиционным проектом является проект 1.

Заключение

Рассмотренная в статье модель является композицией модели максимума ожидаемой остаточной стоимости и максимума гарантированного значения остаточной стоимости. Данная модель более адекватная для реальных условий.

Перспективным направлением развития данной модели является использование композиции модели максимума ожидаемой остаточной стоимости с другими априорными моделями. Например, модель

чрезмерного оптимизма, Сэвиджа и др.

Список литературы

[1] Крушвиц, Л. Инвестиционные расчеты : учебник для вузов. — СПб. : Питер, 2001. — 414 с.

[2] Чисар, И. Теория информации / И. Чисар, Я. Кернер. — М. : Мир, 1985. — 400 с.

Библиографическая ссылка

Крутелева, Е. А. Композиционная модель выбора многопериодных инвестиций при неопределенности // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 93–99.

Сведения об авторах

Крутелева Елена Алексеевна
Студентка магистратуры
Направление «Прикладная информатика»

УДК 004.032.26 + 004.94

Применение нейронных сетей в моделировании искусственной жизни

Логинов Д. А.

Тверской государственный университет

Аннотация. В рамках данной статьи рассматривается разработка и программная реализация алгоритма, моделирующего поведение живых организмов в ограниченной среде и их взаимодействие между собой, и вычисляющего оптимальные параметры нейронной сети, отвечающей за поведение живого организма, при заранее заданных закономерностях среды. В статье представляются результаты экспериментов с помощью разработанного алгоритма, вычисляются оптимальные параметры для нейронной сети интеллектуального агента, моделирующего поведение живого организма, и рассматриваются основные особенности взаимодействия интеллектуальных агентов в заданной искусственной среде.

Введение

В настоящее время все большую значимость принимает проблема конкурирующих интеллектуальных агентов, начиная с устройств умного дома и заканчивая автомобилями с автопилотом и роботами-курьерами. Вместе с человеком они образуют единую среду, в которой у каждого из них есть определенная цель. В рамках данной статьи предусматривается моделирование поведения организмов в замкнутой искусственной среде (далее — среда) с элементами естественного отбора. В качестве моделей таких организмов используются интеллектуальные агенты.

В качестве понятия агента (или интеллектуального агента) берется следующее определение.

Интеллектуальный агент (далее — агент) — самостоятельная сущность, получающая информацию об окружении из внешней среды и информацию о собственном состоянии, принимающая на основе этой информации решение о воздействии на среду в целом.

Ввиду ограниченности информации для обучения и большого количества случайных факторов возникает актуальная проблема: нельзя точно определить оптимальное решение для поставленной цели.

В рамках данной статьи рассматривается реализация алгоритма, позволяющего найти оптимальные параметры для нейронной сети интеллектуального агента в среде с заданными правилами, то есть позволяющего достичь приспособленности агента в ней при большом количестве неизвестных.

1. Общее описание модели среды с агентами

Имеется ограниченное поле заданного размера. На поле расположены агенты, моделирующие поведение живых организмов и имеющие свое состояние, отражаемое в параметрах:

- сытость;
- жажда;
- здоровье;
- энергия;
- зона видимости;
- позиция на поле;
- класс;
- угол поворота;
- зона взаимодействия;
- скорость и направление движения.

Также на поле в случайном порядке располагаются объекты типов, отличных от агента:

- еда;
- вода;

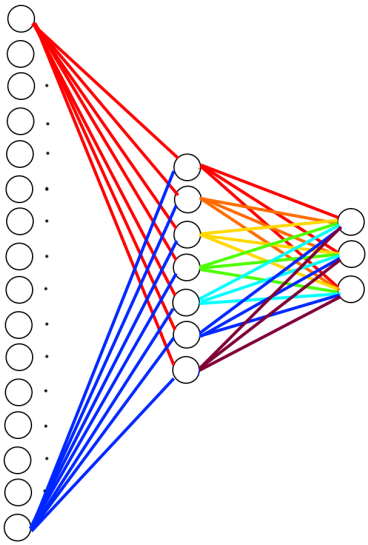

Рис. 1. Схематическое представление нейронной сети агента.

- сухая еда.

Задача агента — принимать решения на основе текущего состояния среды:

- взаимодействовать с объектом (съесть);
- изменить скорость;
- повернуться на выбранный угол.

Цель агента — приспособление к заданным условиям среды.

Для принятия решений у каждого агента присутствует собственная нейронная сеть прямого распространения (см. рис. 1 — схематическое представление нейронной сети агента)

В качестве входных параметров нейронной сети агента подаются его собственные параметры, а также параметры трех ближайших объектов в зоне его видимости. В качестве выходных параметров

нейронная сеть используется три нейрона, имеющих значение в диапазоне от 0 до 1 и отображающих решение о еде, изменение скорости и угла отклонения агента.

В качестве способа обучения применяется два подхода. Первый подход заключается в предварительном обучении нейронной сети агента на заранее известных данных (обучение с учителем). Второй подход подразумевает собой совмещение обучения с подкреплением и обучение через эволюцию.

2. Первый подход

Первым шагом является формирование обучающей выборки.

В качестве алгоритма для формирования обучающей выборки для обучения с учителем выступает следующий.

1) Случайным образом генерируются параметры агента.
2) Случайным образом генерируются три объекта в зоне видимости агента из п. 1.
3) Вычисляются расстояния и углы от агента из п. 1 до объектов из п. 2.
4) Для каждого видимого объекта:
   - 4.1) если здоровье меньше 30:
     - 4.1.1) если рассматриваемый объект — агент, то остановиться и повернуть максимально вправо;
     - 4.1.2) иначе поменять угол по направлению к объекту, увеличить скорость до максимума и съесть объект;
   - 4.2) иначе поменять угол по направлению к объекту, увеличить скорость до максимума и съесть объект.
5) Если ничего не было съедено, то проверяем наличие стены перед агентом и выносим решение об изменении скорости и угла.

Обучающая выборка делится на две части, в соотношении 85% к 15%, где 85% — выборка, используемая в обучении, 15% — выборка, используемая в проверке точности обучения. Далее проводится обучение нейронной сети агента и добавление его в среду.

3. Второй подход

Создается модель среды с агентами, в которой заданы правила изменения состояния агентов и правила их взаимодействия со средой.

В рассматриваемой модели вводится стимулирующий фактор для агентов, то есть изменение его параметров с течением времени. В данном случае в модели реализованы зависимости параметров агента от различных факторов, таких как:

- время;
- внутренние факторы;
- внешние факторы.

При инициализации среды объекты на поле расположены в случайном порядке, но ни одна пара объектов не расположена в одной точке.

За один шаг среды принимается один такт работы основного алгоритма.

1) Проверяется признак наличия/отсутствия агента на поле и выполняется удаление лишних агентов с поля.
2) Вычисляются функции приспособленности агентов и проводится один шаг генетического алгоритма. Если агентов на поле меньше заданного предела, то случайным образом генерируются новые агенты.
3) Проводится один шаг обучения агентов.
4) Если количество еды меньше заданного предела, то генерируется новая еда.
5) Проводится операция съедения объектов агентами, изменения состояния объектов, после чего выполняется отрисовка всех объектов в графическом интерфейсе. Стоит отметить, что при удалении агента с поля он все равно используется при скрещивании до следующего выполнения селекции.

В качестве функции приспособленности для каждого агента берется следующая формула:

$$fitFunc = \frac{isAlive}{4} \cdot \left(\frac{eatenMeal}{maxEatenMeal} + \frac{agtGenNum}{maxGenNum} + \right.$$
$$\left. + \frac{lifeTime}{maxLifeTime} + \frac{hunger + thirst + health + energy}{400} \right),$$

где $eatenMeal$ — количество съеденных агентом объектов, $maxEatenMeal$ — максимальное количество съеденных одним агентом объектов, $agtGenNum$ — количество поколений агента, $maxGenNum$ — максимальное количество поколений, $lifetime$ — время жизни агента, $maxLifeTime$ — максимальное время жизни агентов, $hunger$ — сытость/голод агента, $thirst$ — жажда агента, $health$ — здоровье агента, $energy$ — энергия агента.

В качестве методов для использования в генетическом алгоритме рассматриваются следующие.

- Методы селекции: турнирная селекция, метод рулетки, метод ранжирования.
- Методы подбора родителей: панмиксия, инбридинг, аутбридинг.

Параметры обучения методом обратного распространения ошибки выбираются таким образом, чтобы обучение не происходило слишком долго и достигалась приемлемая точность. Количество эпох для обучения берется равным 10, а целевая точность обучения — 0.01.

При этом скорость выполнения шагов среды обратно пропорциональна количеству допустимых агентов среды и количеству эпох обучения.

4. Программная реализация

Программная реализация модели содержит в себе следующий набор классов: NNLayer — класс для реализации одного слоя нейронной сети агента; NeuralNetwork — класс для реализации нейронной сети агента; Agent — класс для реализации непосредственно агента; GeneticAlgorithmNN — класс для реализации генетического алгоритма, необходимого для подбора оптимальных параметров нейронной сети и предварительного обучения агента; GeneticAlgorithm — класс для реализации генетического алгоритма, используемого в среде; EnvManager — класс для реализации среды с агентами и ее правил;

EnvGui — класс для графического отображения среды с агентами, а также окон статистики и графика функции приспособленности

5. Графический интерфейс

Графический интерфейс представляет собой отображение основного поля среды с объектами в ней (см. рис. 2 — графическое отображение среды с агентами), окна статистики агентов и графика функции приспособленности для агента с самым высоким ее значением (см. рис. 3 — графическое отображение окна статистики и графика функции).

На поле с объектами отображаются агенты, а также объекты, представляющие еду. Каждый объект на поле отмечен кругом, имеет свой размер и цвет.

Агенты на поле обозначены числом, соответствующим их порядковому номеру. У каждого агента обозначен вектор направления движения, область видимости и область взаимодействия (область съедения).

Объекты, представляющие еду, обозначаются символами m, d, w в зависимости от их параметров.

В окне статистики отображаются все параметры агента с наибольшей функцией приспособленности, а также среднее значение функции приспособленности и среднее значение количества поколений для всех агентов на поле.

В окне графика отображается информация об агенте с самым высоким значением функции приспособленности, для чего отдельно сохраняются значения функции приспособленности агента на протяжении 50 шагов работы среды.

6. Проведение экспериментов

Эксперимент с первым подходом. Вычисляются оптимальные параметры нейронной сети для обучения с учителем по переданной обучающей выборке. Для вычисления используется класс GeneticAlgorithmNN.

Предварительно формируется обучающая выборка для обучения с учителем по ранее описанному алгоритму.

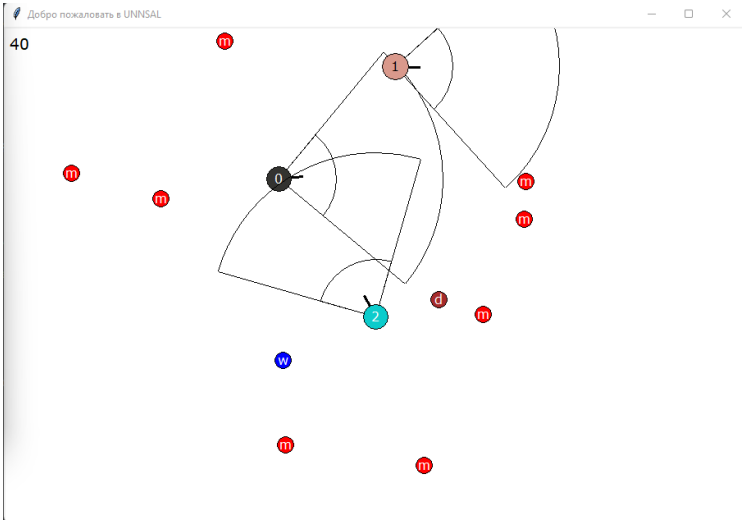

Рис. 2. Графическое отображение среды с агентами.

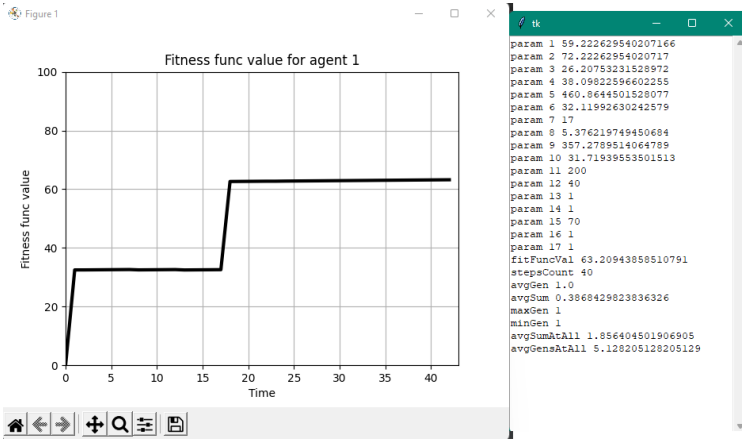

Рис. 3. Графическое отображение окна статистики и графика функции.

После генерации обучающей выборки выполняется поиск оптимальных параметров нейронной сети с использованием двух методов ga_instance.run() и ga_instance.best_solution() вспомогательной библиотеки pygad.

В результате вычисления получены параметры нейронной сети, при которых ее обучение происходит с максимальным значением функции приспособленности, то есть с наибольшей точностью обучения для количества эпох, равного 10 и целевой точности обучения, равной 0.01:

- размер нейронной сети — $[14, 8, 6]$;
- функция активации — softmax;
- оптимизатор — Adadelta;
- функция потерь — log_cosh;
- количество скрытых слоев — 3.

Полученные параметры применяются во втором вычислительном эксперименте.

Эксперимент со вторым подходом. Проводится вычисление оптимальных параметров нейронной сети для достижения наибольшего значения функции приспособленности (обучение с подкреплением) осуществляется следующим образом.

1) Создается объект класса NeuralNetwork с ранее вычисленными параметрами и сгенерированной тестовой выборкой.
2) Созданная нейронная сеть обучается на сгенерированной выборке.
3) Создается объект класса EnvManager с параметрами agtCount — 10, selectionThreshold — 20, в который в качестве параметра добавляется нейронная сеть из п. 2).

С помощью метода mainLoop с параметрами $[1, 20, 40, 2, 10]$ запускается среда с агентами, в которой вычисляются параметры нейронной сети агента с наибольшей функцией приспособленности.

После прохождения алгоритма нахождения параметров на количестве шагов среды, равном 1 800, были получены следующие результаты.

С помощью подхода совмещения генетического алгоритма и алгоритма обратного распространения ошибки к нейронным сетям, а также применения генетического алгоритма к вектору весов и смещений наибольшее значение функции приспособленности агента достигает значения 0.829 или 82.9%. Обозначим его как агент-2. Функция приспособленности агента, обученного с помощью первого подхода, достигает значения 0.676 или 67.6%. Обозначим его как агент-1.

Количество пройденных агентом-2 шагов среды равно 640, номер его поколения равен 7, а количество съеденных объектов равно 15.

Количество пройденных агентом-1 шагов среды равно 480, а количество съеденных объектов равно 12.

Параметры нейронной сети агента-2 имеют значения:

- размер нейронной сети — $[8, 5, 1, 1]$;
- функция активации — softsign;
- оптимизатор — Adadelta;
- функция потерь — log_cosh;
- количество скрытых слоев — 4.

Стоит отметить, что половина параметров агента-2 совпадает с параметрами агента-1. При этом размер нейронной сети агента-1 больше по количеству нейронов в слоях, но меньше по количеству скрытых слоев. Это показывает, что в данной области исследования предварительно обученный агент имеет излишнее количество нейронов в своей нейронной сети и, как следствие, тратит больше ресурсов на ее поддержание, что порождает меньшее значение функции приспособленности данного агента по отношению к функции приспособленности агента, созданного в рамках комбинированного алгоритма, и доказывает ограниченность тестовой выборки и низкую эффективность обучения с учителем перед случайными событиями.

Все вышеописанные результаты достигаются при условии, что агент может есть в том числе агентов своего типа. В противном случае на 300 шаге среды количество типов агентов на поле стремится к 1. Это ограничение напрямую влияет на структуру нейронной сети агентов таким образом, что количество нейронов в слоях нейронной

сети агента стремится к 1 с каждым новым агентом. Значение функции приспособленности таких агентов стремится к 0 и вычисление оптимальных параметров для заданной среды невозможно.

Заключение

В рамках данной статьи была рассмотрена модель искусственной среды с интеллектуальными агентами и программная реализация алгоритма, моделирующего эту среду и вычисляющего оптимальные параметры нейронной сети для существования организма при заданных правилах среды; представлены результаты экспериментов, проведенных с помощью рассмотренного алгоритма; вычислены оптимальные параметры для нейронной сети интеллектуального агента, моделирующего поведение живого организма; а также рассмотрены основные особенности взаимодействия интеллектуальных агентов в заданной искусственной среде.

Данную задачу возможно расширить для применения в ситуациях, когда существует набор конкурирующих сущностей, имеющих различные цели. Например, рассмотренный алгоритм может быть применен для оптимизации работы роботизированных погрузчиков в складской логистике.

В качестве дополнительных возможностей для расширения разработанного алгоритма предлагается хранение истории ключевых ситуаций для использования их в процессе обучения, а также использование многопоточности для уменьшения временных затрат при работе алгоритма.

Список литературы

[1] Джонс, М. Т. Программирование искусственного интеллекта в приложениях / пер. с англ. А. И. Осипова. — М. : ДМК Пресс, 2004. — 312 с. : ил.

[2] Prateek, J. Artificial Intelligence with Python. — Birmingham : Packt Publishing, 2017. — 437 p.

Библиографическая ссылка

Логинов, Д. А. Применение нейронных сетей в моделировании искусственной жизни // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 100–111.

Сведения об авторах

Логинов Денис Александрович
Студент магистратуры
Направление «Прикладная математика и информатика»

УДК 519.854.2

Применение методов имитационного моделирования и интеллектуальной оптимизации для решения задачи организации доставки товаров посредством дронов

Лосев Я. С.

Тверской государственный университет

Аннотация. Задача маршрутизации (VRP) с различными ограничениями является обобщением NP-полной задачи коммивояжера. В статье предложен трехфазный алгоритм, позволяющий сократить вычислительное время решения путем разбиения исходной задачи на подзадачи меньшей размерности. По результатам анализа работы алгоритма на основе открытых данных соревнования Google Hash Code, предложенный алгоритм входит в ТОП-5 лучших решений.

Введение

Рынок электронной коммерции, подкрепленный онлайн рекламой в социальных сетях, достаточно сильно вырос за последние несколько лет. Например, один из крупнейших игроков на российском рынке — Озон — отчитался о более чем 220 млн. совершенных заказов, а общий оборот продаж (GMV) достиг 445 млрд. рублей.

Задача маршрутизации транспортных средств (англ. Vehicle Routing Problem, VRP) впервые упоминается в работах Данцига и Рамстера в 1959 году [1]. Задача представляет собой проблему комбинаторной оптимизации. На естественном языке она формулируется достаточно лаконично: «Какими должны быть маршруты некоторого набора транспортных средств, чтобы посетить каждого из покупателей, понеся при этом минимальные затраты». Задача VRP является обобщением известной задачи коммивояжера, с некоторым набором дополнительных ограничений и несколькими коммивояжерами.

Задача VRP имеет большое количество модификаций. В их числе:

- Vehicle Routing Problem with Profits (VRPP). Посещение всех локаций необязательно. Цель — максимизировать некоторую выгоду, полученную с каждого из заказов;
- Capacitated Vehicle Routing Problem (CVRP). Вводится ограничение на грузоподъемность транспортного средства;
- Vehicle Routing Problem with Multiple Trips (VRPMT). Транспортное средство может иметь несколько маршрутов, например посетить несколько покупателей, вернуться на склад и снова отправиться производить доставку;
- Open Vehicle Routing Problem (OVRP). Транспортное средство не обязано возвращаться в исходную точку маршрута;
- Multi-Depot Vehicle Routing Problem (MDVRP). Существует несколько начальных точек отправления;
- Multi-Product Vehicle Routing Problem (MPVRP). Существует несколько различных видов товаров;
- Split Delivery VRP (SDVRP). Каждый покупатель может обслуживаться одновременно несколькими транспортными средствами.

Добавление любого из ограничений или даже нескольких одновременно значительно увеличивает количество возможных комбинаций, а значит, делает и без того NP-полную задачу еще более вычислительно сложной в практическом применении.

Существующие подходы чаще всего описывают методы решения одной из модификаций задачи VRP, но не описывают случаи, когда все эти ограничения присутствуют одновременно.

Из условия исходной задачи мы видим [2], что наша задача является комбинацией одновременно нескольких модификаций VRP, описанных выше: VRPP, CVRP, VRPMT, OVRP, MDVRP, MPVRP, SDVRP.

Попробуем разбить нашу подзадачу на более гранулярные подзадачи, решения которых уже были описаны в том или ином виде и предложим ряд оптимизаций для уменьшения времени выполнения алгоритма. После этого оценим полученные результаты.

1. Назначение складов

Первым шагом будет являться определение, с какого из складов повезется тот или иной товар для конкретного покупателя. Сформулируем эту задачу в виде P независимых транспортных задач, где P — количество различных видов товаров [3]. Для оптимизации времени исполнения положим, что каждая из подзадач может вычисляться в отдельном потоке.

Каждая из подзадач формулируется следующим образом. Допустим, у нас имеется m складов, на которых содержится некоторый товар, и n потребителей, заказавших данный товар.

Для каждого склада i и для каждого заказа j заданы следующие величины:

- a_i — объем товара на складе i;
- b_j — количество товара в заказе j;
- c_{ij} — расстояние от склада i до заказа j.

Суммарное количество товара на всех складах равно суммарной потребности для удовлетворения всех заказов:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j.$$

Требуется назначить каждому заказу склад, с которого будет отгружен некоторый товар, удовлетворив спрос каждого из покупателей, при этом затратить как можно меньше времени на транспортировку (рис. 1):

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \to \min,$$

$$\sum_{j=1}^{n} x_{ij} = a_i,\ i = 1,\dots,m,$$

$$\sum_{i=1}^{m} x_{ij} = b_j,\ j = 1,\dots,n.$$

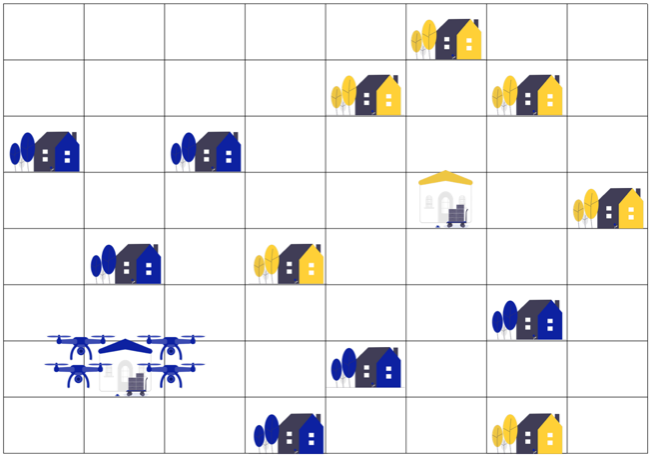

Рис. 1. Пример назначения складов.

Задачу можно решать любым из известных методов, например симплекс-методом. В результате получаем некоторую таблицу вида «заказ-товар-склад».

2. Ранжирование заказов

В постановке задачи каждый заказ оценивается только относительно времени доставки, поэтому имеет смысл отсортировать заказы для нас по выгоде. Будем считать заказ тем менее выгодным, чем больше следующая метрика:

$$WD = \sum_{i \in O} w_i \times \sum_{p \in O} d_p,$$

где O — множество заказов, w_i — вес товара, а d_p — расстояние от назначенного склада до заказа.

3. Кластеризация заказов

Следующим шагом разобьем заказы на некоторые оптимальные кластеры, в которых пройденное внутри кластера расстояние будет

Рис. 2. Пример разбиения на регионы.

минимальным, а вес товаров — наоборот, максимальным, но не превышающим грузоподъемность ТС.

Для этого разобьем карту на N регионов примерно равных размеров, а каждый из этих регионов — еще на K пачек (рис. 2). Данное разбиение является своего рода эвристикой, позволяющей сократить размерности решаемых далее подзадач и дающей возможность делать это параллельно и независимо друг от друга.

Анализ данных показал, что грузоподъемность дрона всего в 3–4 раза больше, чем средний вес товара. Соответственно, получаемые кластеры явно не будут включать большое количество вершин. На этом заключении и основывается наша эвристика.

Для кластеризации решим задачу OCVRP, полагая, что количество ТС условно бесконечно, одновременно полагая, что мы должны использовать минимально возможное количество ТС.

Задача OCVRP имеет следующую математическую формулировку.

Пусть $G = (V, H, c)$ — полный ориентированный граф с n вершинами V и ребрами H. Вершина с номером 0 является некоторым складским помещением, где расположено p транспортных средств с одинаковой грузоподъемностью Q. Оставшиеся вершины — покупатели.

Каждый покупатель $i \in V$, $i \neq 0$ имеет некоторую потребность в продукте $d_i \leqslant Q$. Каждое ребро $(i, j) \in H$ имеет некоторую неот-

рицательную длину пути c_{ij}. Дронов достаточно, чтобы обслужить все заказы с учетом их веса.

Объявим переменные $x_{rij} \in \{0, 1\}$, которые означают, что транспортное средство p проходит по ребру (i, j) в ходе оптимального решения. Тогда задача целочисленного линейного программирования может быть сформулирована следующим образом:

минимизировать

$$\sum_{r=1}^{p} \sum_{i=0}^{n} \sum_{j=0, j\neq i}^{n} c_{ij} x_{rij}$$

учитывая

$$\sum_{r=1}^{p} \sum_{i=0, i\neq j}^{n} x_{rij} = 1, \ \forall j \in \{1, \ldots, n\},$$

$$\sum_{j=1}^{n} x_{r0j} = 1, \ \forall r \in \{1, \ldots, p\},$$

$$\sum_{i=0, i\neq j}^{n} x_{rij} = \sum_{i=0}^{n} x_{rji}, \ \forall j \in \{0, \ldots, n\}, r \in \{1, \ldots, p\},$$

$$\sum_{i=0}^{n} \sum_{j=1, j\neq i}^{n} d_j x_{rij} \leqslant Q, \ \forall r \in \{1, \ldots, p\},$$

$$\sum_{r=1}^{p} \sum_{i\in S} \sum_{j\in S, i\neq j} x_{rij} \leqslant |S| - 1, \ \forall S \subseteq \{1, \ldots, n\},$$

$$x_{rij} \in \{0, 1\}, \ \forall r \in \{1, \ldots, p\}, i, j \in \{0, \ldots, n\}, i \neq j.$$

Решая несколько независимых подзадач OVRP, мы получим необходимые нам кластеры (рис. 3). Для решения можно использовать любой из известных генетических алгоритмов, например алгоритм имитации отжига.

4. Маршрутизация кластеров

Помним, что в предыдущем шаге мы получили кластеры, которые являются в некотором смысле «максимальными», то есть добавить

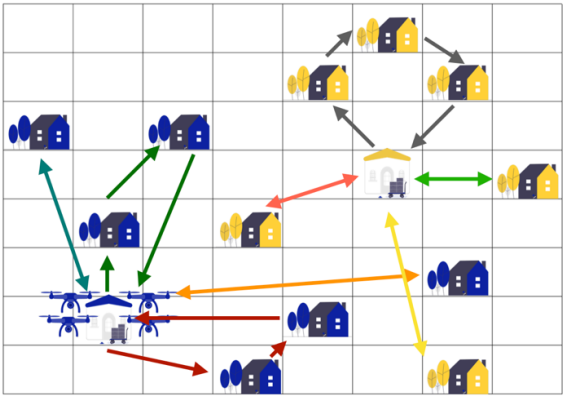

Рис. 3. Пример построенных кластеров.

еще один заказ в них не является возможным из-за ограничения по весу. Кроме того, задача решалась из предположения, что каждый дрон может совершать только один маршрут. Но в нашей задаче это ограничение отсутствует: дроны могут совершать сколь угодно много маршрутов до тех пор, пока не иссякнет время симуляции.

Поэтому следующим шагом мы попытаемся определить оптимальный порядок, в котором дроны будут обходить полученные кластеры. Кластеры уже учитывают ограничение на максимальную грузоподъемность, поэтому достаточно решить классическую задачу VRP, где в качестве вершин будут выступать наши полученные кластеры.

Кластеры отсортируем по среднему значению метрики WD заказов, попавших в этот кластер. А затем разобьем кластеры на N равных пачек. И решим для каждой из них задачу VRP (получается из задачи, приведенной выше, путем удаления ограничения на грузоподъемность), полагая, что расстояние от кластера i до кластера j — длина пути от последней вершины кластера i до первой вершины кластера j плюс длина маршрута внутри кластера j (рис. 4).

Далее приводим все маршруты в соответствие с требованием к формату выходных данных.

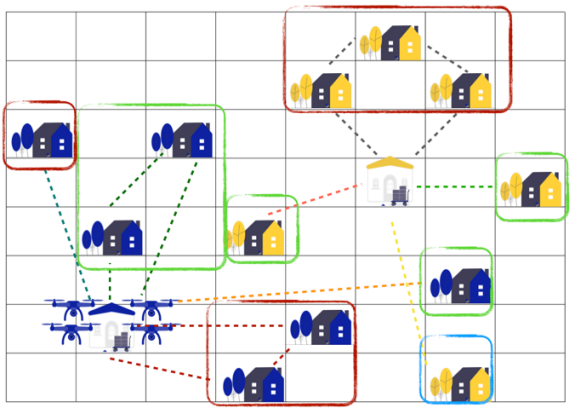

Рис. 4. Пример разбиения кластера на пачки.

5. Анализ результатов

Вычислительные эксперименты проводились на достаточно мощном вычислительном устройстве со следующими характеристиками: Intel Ice Lake 3.7 GHz, 24 потока, 72 Гб ОЗУ.

На первом шаге, при назначении складов, получаем следующие результаты:

- симплекс-метод показал наилучший результат;
- жадный алгоритм в среднем проигрывает в 2 раза (длина маршрута 550 545 ед. vs 745 411 ед. для набора данных из условиях задачи);
- параллельность дала ускорение в 4 раза, время выполнения — 37 сек.

Кластеризация заказов:

- разбиение карты на области и пачки дает потерю в средней загруженности дронов 1–2%;
- разбиение дает ускорение в 6 раз — 10 сек vs 60 сек;
- средняя загрузка — 92% от максимальной.

Планирование доставок и итоговый результат:

- разбиение на пачки позволяет решить исходную задачу за время около 1 часа. При отсутствии разбиения результат выполнения программы неизвестен из-за ограничения по времени;
- итоговый результат — 114 412 очков, что дает нам попадание примерно в TOP-5 (максимальный результат — 115 399).

В целом, исходя из вышеизложенного, мы видим, что наш алгоритм дает достаточно хороший результат и благодаря предложенным оптимизациям имеет приемлемое время выполнения.

Заключение

В ходе работы рассмотрены основные шаги для решения поставленной задачи и предложен ряд оптимизаций. Введенные оптимизации значительно сокращают время выполнения алгоритма, что критично при задачах большой размерности. Одновременно, показано, что введенные оптимизации не дают значительного ухудшения решения, при этом давая приемлемое время выполнения алгоритма.

Дальнейшие улучшения работы алгоритма могут быть связаны со сравнением эффективности различных методов решения на каждом из шагов, например, различные эвристики для построения первого решения оптимизационных алгоритмов. Кроме того, улучшение может дать добавление условия, что дроны могут перевозить товары между складами, тем самым ускоряя время доставки в ряде случаев.

Список литературы

[1] Dantzig, G. B. The Truck Dispatching Problem / G. B. Dantzig, J. H. Ramser // Management Science. — 1959. — Vol. 6, № 1. — P. 80–92.

[2] Hash Code Archive — Drone Delivery : [сайт]. — URL: https://www.kaggle.com/c/hashcode-drone-delivery (дата обращения: 23.04.2022).

[3] Корбут, А. А. Дискретное программирование / А. А. Корбут, Ю. Ю. Финкельштейн. — М. : Наука, 1969. — 368 с.

Библиографическая ссылка

Лосев, Я. С. Применение методов имитационного моделирования и интеллектуальной оптимизации для решения задачи организации доставки товаров посредством дронов // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 112–121.

Сведения об авторах

Лосев Ярослав Станиславович
Студент магистратуры
Направление «Фундаментальная информатика и информационные технологии»

УДК 330.322

Модель диверсификации многопериодных инвестиций при смешанной неопределенности

Опарина Т. Ю.

Тверской государственный университет

Аннотация. Статья посвящена модели формирования портфеля многопериодных инвестиций с учетом комбинированной неопределенности. Приведены математическая постановка задачи и ее представление моделью конечной антагонистической игры. Для нахождения оптимального портфеля используются задачи линейного программирования. Применение модели демонстрируется на тестовом примере.

Введение

Рассматривается задача портфельного анализа многопериодных инвестиций в условиях, когда проект может остановиться в любой непредвиденный момент реализации и имеет неопределенность по процентным ставкам.

Используемые в настоящее время модели портфельного анализа в общем случае не предусматривают совокупный учет возможности прекращения реализации проекта в любой период времени и не рассматривают возможность вариации процентных ставок на интервалах. Данное обстоятельство обуславливает актуальность развития портфельного анализа.

Целью работы является повышение обоснованности формирования инвестиционных портфелей с учетом рассматриваемых условий на основе теоретико-игровой модели антагонистического конфликта. Формирование компонентов данной модели основано на использовании показателя остаточной стоимости по периодам реализации проектов, из которого формируется матрица капитализации, используемая в дальнейшем как матрица выигрышей [3].

В основе решения задачи лежит интерпретация вектора диверсификации инвестиционных вложений с использованием оптимальных смешанных стратегий игры.

Применение представляемой в статье модели иллюстрируется на тестовом примере.

1. Математическая постановка задачи

Пусть инвестор имеет единичный капитал, который предполагает полностью вложить в финансовые активы.

Рассмотрим множество многопериодных инвестиционных проектов $I = \{1, 2, \ldots, N\}$ [3]. Для обозначения периодов реализации проектов примем символ $t = 0, 1, \ldots, T$. Здесь T — плановый инвестиционный горизонт.

Ставится задача формирования портфеля $x = (x_1, x_2, \ldots, x_N)$ многопериодных инвестиционных проектов, где $\sum_{i=1}^{N} x_i = 1$ и $x_i \geqslant 0$, $i = 1, 2, \ldots, N$. Вектор x определяет диверсификацию многопериодных инвестиций между проектами из множества I.

Полагается, что процентные ставки по заимствованию и инвестированию являются случайными величинами с равномерным распределением. При этом условия реализации проектов могут привести к их остановке в любой период, то есть имеется полная неопределенность по времени.

Реальные условия реализации проектов удовлетворяют возможностям представления конфликта между неопределенностью и инвестором конечной антагонистической игрой [2]. Компоненты такой игры $\Gamma = \langle X, Y, \varphi \rangle$ имеют следующую интерпретацию: инвестор является первым игроком; сторона, определяющая условия прекращения реализации проектов — второй игрок; возможные действия инвестора определены через множество чистых стратегий $X = I$; возможные действия другой стороны по прекращению реализации проектов определены через $Y = \{0, 1, \ldots, T\}$; функция выигрыша $\varphi(i, t) = C(i, t)$ — остаточная стоимость, получаемая на период реализации t для проекта i, $(i, t) \in X \times Y$.

Исходными данными для вычисления показателя φ являются одинаковые для всех проектов уровень изъятий Y со структурой

$f = (f_0, f_1, \ldots, f_T)$, базовые платежи $M = (M_0, M_1, \ldots, M_T)$, процентные ставки по инвестированию $h = (h_1, h_2, \ldots, h_T)$ и по заимствованию $s = (s_1, s_2, \ldots, s_T)$. Индивидуальными для каждого проекта i являются платежи $z(i) = (z_0(i), z_1(i), \ldots, z_T(i))$, связанные с его реализацией.

Обозначим далее в смешанном расширении игры через $A = (\alpha_1, \alpha_2, \ldots, \alpha_N)$ и $B = (\beta_0, \beta_1, \ldots, \beta_T)$ — множества смешанных стратегий первого и второго игроков соответственно, где α_i — вероятность, с которой применяется стратегия i; $i = 1, 2, \ldots, N$, а β_t — вероятность прерывания проекта в момент t; $t = 0, 1, \ldots, T$.

При заданных условиях целевая функция игры определяется по формуле

$$H(\alpha, \beta) = \sum_{i=1}^{N} \sum_{t=0}^{T} \varphi(i, t) \cdot \alpha_i \cdot \beta_t. \qquad (1)$$

Формула (1) определяет математическое ожидание выигрыша инвестора в ситуации (α, β).

Известно, что решение такой игры всегда существует [2]. При этом оптимальная смешанная стратегия инвестора допускает следующую содержательную интерпретацию: вероятностьα_i^* применения стратегии $i \in I$ равна доле x_i^* вложения инвестиционного капитала в проект i. То есть любая смешанная стратегия может трактоваться как распределение, определяющее доли вложения капитала инвестора. Содержательную интерпретацию оптимальной смешанной стратегии β^* второго игрока можно сформулировать следующим образом: вероятность прекращения реализации проекта в момент времени $t \in T$ равна β_t^*.

Рассмотренные положения определяют математическую постановку задачи, а решение игры обеспечивает формирование оптимального портфеля. Функция выигрыша ориентирована на максимизацию математического ожидания остаточной стоимости.

На базе матрицы остаточной стоимости может быть сформирована матрица рисков.

2. Модель формирования инвестиционного проекта

Рассматривается несовершенный неограниченный рынок капитала. Формирование матрицы выигрышей основано на использовании алгоритма вычисления остаточной стоимости в любом периоде, предложенного (алгоритма) в [3].

Для $t = 0$ остаточная стоимость равна

$$C(i, 0) = M_0 - f_0 Y + z_0(i).$$

Далее, для $t > 0$ остаточная стоимость определяется как

$$C_t(i) = M_t - f_t Y + z_t(i) + (1 + h_t) \cdot C_{t-1}(i) \text{ при } C_{t-1}(i) > 0,$$
$$C_t(i) = M_t - f_t Y + z_t(i) + (1 + s_t) \cdot C_{t-1}(i) \text{ при } C_{t-1}(i) < 0.$$

Для учета вероятностных неопределенностей по процентным ставкам по каждому проекту используются оценки математических ожиданий остаточных стоимостей в каждом периоде.

На основании рассчитанной капитализации для всех проектов по всем периодам времени формируем матрицу капитализации $C = \|C_t(i)\|$, $i = 1, 2, \ldots, N$, $t = 0, 1, \ldots, T$. В этой матрице каждая строка соответствует проекту, а каждый столбец — возможному времени остановки проекта.

Так как при расчете есть вероятность появления отрицательных элементов в матрице и выигрыш инвестора может принять отрицательное значение, то для исключения такой ситуации, согласно [2], можно добавить достаточно большое положительное число к матрице выигрыша. При этом стратегии игроков не изменятся.

Матрица остаточных стоимостей $C' = \|C'_t(i)\|$, $i = 1, 2, \ldots, N$, $t = 0, 1, \ldots, T$, является матрицей выигрышей при решении задачи максимизации ожидаемой капитализации проектов для любого момента прерывания реализации.

Целесообразно использовать метод сведения игры к паре двойственных задач линейного программирования для нахождения решения игры [1].

Прямая задача линейного программирования формируется сле-

дующим образом:

$$\begin{cases} \sum\limits_{j=1}^{T} u_j \to \max\limits_{u}, \\ \sum\limits_{j=1}^{T} h_{ij} \cdot u_j \leqslant 1, \ i = 1, 2, \ldots, N, \\ u_j \geqslant 0, \ j = 0, 1, \ldots, T. \end{cases} \tag{2}$$

Если $u^* = (u_0^*, \ldots, u_T^*)$ — решение прямой задачи линейного программирования, то оптимальная стратегия второго игрока равна вектору $\beta^* = (\beta_0^*, \beta_1^*, \ldots, \beta_T^*)$, который определяется по формуле (3):

$$\beta_j^* = \frac{u_j^*}{\sum\limits_{j=1}^{T} u_j^*}, \ j = 0, 1, \ldots, T. \tag{3}$$

Двойственная задача линейного программирования имеет вид:

$$\begin{cases} \sum\limits_{i=1}^{N} t_i \to \min\limits_{t}, \\ \sum\limits_{i=1}^{N} h_{ij} \cdot t_i \geqslant 1, \ j = 0, 1, \ldots, T, \\ t_i \geqslant 0, \ i = 1, 2, \ldots, N. \end{cases} \tag{4}$$

Если $t^* = (t_1^*, t_2^*, \ldots, t_N^*)$ — решение двойственной задачи линейного программирования, то оптимальная стратегия первого игрока равна вектору $\alpha^* = (\alpha_1^*, \ldots, \alpha_N^*)$, который определяется по формуле (5):

$$\alpha_i^* = x_i^* = \frac{t_i^*}{\sum\limits_{i=1}^{N} t_i^*}, \ i = 1, \ldots, N. \tag{5}$$

Основным методом решения сформулированных задач линейного программирования является симплекс-метод, положения которого подробно описаны в источнике [1].

Момент времени, t	0	1	2	3
Проект 1, $z_t(1)$	-500	-400	800	400
Проект 2, $z_t(2)$	-300	-800	1200	200
Проект 3, $z_t(3)$	-900	800	360	-10
Диапазоны ставок по заимствованию, s_t	–	$[0.10, 0.14]$	$[0.09, 0.12]$	$[0.09, 0.12]$
Диапазоны ставок по инвестированию, h_t	–	$[0.03, 0.06]$	$[0.05, 0.08]$	$[0.04, 0.07]$
Базовые платежи, M_t	600	100	-200	800
Изъятия, $f_t Y$	20	22	24	26

Таблица 1. Исходные данные.

Период / Проект	0	1	2	3
1	502.4	186	733.6	1935.6
2	702.4	0	925.3	1944.6
3	102.4	935.6	1128	1955.5

Таблица 2. Матрица капитализации.

Рассмотренный подход может быть применен при использовании в качестве матрицы выигрышей матрицы рисков, в которой каждый элемент определяется по формуле (6):

$$r(i,t) = C_t^{\max} - C_t(i), \tag{6}$$

где $C_t^{\max}$ — максимально возможная капитализация в момент времени t.

3. Применение модели

Рассматриваются результаты применения моделей при исходных данных, представленных в табл. 1. На основе этих данных рассчитана матрица капитализации (табл. 2).

При решении задачи, была найдена оптимальная диверсификация $x^* = (0, 0.54, 0.46)$ инвестиционных вложений, обеспечивающая

максимизацию ожидаемого уровня капитализации проектов. Согласно решению задачи инвестору рекомендуется вложить 54% средств во второй проект и 46% средств в третий проект. При этом, оптимальная стратегия конфликтующей стороны (игрока 2) определяется вектором $y^* = (0.61, 0.39, 0, 0)$, согласно которому рекомендуется остановка реализации проектов в начальный момент с вероятностью 0.61 и в первый период с вероятностью 0.39.

Заключение

Рассмотренная в статье модель портфельного анализа инвестиций является обобщением известных моделей на многопериодный случай. Основным результатом исследования является обоснование и реализация теоретико-игровой модели для определения оптимальной диверсификации инвестиционных вложений в условиях неопределенности по времени и по процентным ставкам. Предложенная модель отличается простотой и обоснованностью исходных данных, а также простотой применения.

Список литературы

[1] Банди, Б. Основы линейного программирования / Пер. с англ. О. В. Шихеевой. — М. : Радио и связь, 1989. — 176 с.

[2] Дюбин, Г. Н. Введение в прикладную теорию игр / Г. Н. Дюбин, В. Г. Суздаль. — М. : Наука, 1981. — 336 с.

[3] Крушвиц, Л. Инвестиционные расчеты : учебник для вузов. — СПб. : Питер, 2001. — 414 с.

Библиографическая ссылка

Опарина, Т. Ю. Модель диверсификации многопериодных инвестиций при смешанной неопределенности // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 122–128.

Сведения об авторах

Опарина Татьяна Юрьевна
Студент магистратуры
Направление «Прикладная информатика»

УДК 004.934.2

Сравнительный анализ разных методов исследования текста

Осинский Р. Р.

Тверской государственный университет

Аннотация. Статья посвящена сравнительному анализу различных моделей для решения задачи сентимент-анализа. В статье приведены модели на основе словарей и правил и модели с использованием алгоритмов машинного обучения. Для оценки и валидации моделей использованы данные из открытых источников. Сравнение проведено на основе точности классификации документов.

Введение

С распространением интернета и, в частности, социальных сетей, растет количество данных, анализ которых традиционными методами становится затруднителен или невозможен. Пользователи интернета генерируют огромное количество данных: посты в социальных сетях, отзывы о товарах, видео, выражают свое мнение социальных проблемах, пишут рецензии на фильмы и книги. Такие данные можно назвать неструктурированными, так как они не имеют четкой структуры, которую можно анализировать традиционными средствами.

Для возможности анализа таких данных крупные IT-компании перекладывают на пользователя ответственность за структурирование данных. Самым очевидным примером здесь можно назвать отзывы о некотором товаре в интернет магазине. Пользователь в своем отзыве зачастую указывают рейтинг продукта по некоторой шкале, перечисляют в специальном поле его достоинства и недостатки, дополняя отзыв информацией, более пригодной к анализу стандартными инструментами. Однако, данный подход ограничивает объем данных для анализа — пользователи могут публиковать отзывы на своих страницах в социальных сетях, форумах и других площадках.

Именно поэтому запрос на инструменты для анализа неструктурированных данных растет с каждым годом, появляются и методы для такого анализа: системы распознавания изображений, рукописного текста, перевода речи в текст и так далее.

Задачи анализа текста решается в рамках направления NLP (Natural Language Processing) — обработки естественного языка. В рамках данного направления исследователи занимаются проблемами компьютерного анализа и синтеза текстов на естественных языках. Одним из направлений такого анализа является так называемый сентимент-анализ текста.

Сентимент-анализ, называемый так же анализом тональности — это класс задач анализа текста применительно к эмоциональной окраске, в том числе для выделения отношения автора к некоторому субъекту, называется задачами сентимент-анализа текста. Существуют различные методы проведения такого анализа, в основном их можно поделить на два множества — алгоритмические методы и методы с использованием машинного обучения. Тем не менее, нельзя однозначно сказать, что тот или иной подход является более точным в плане получаемых с его помощью результатов.

Данное исследование посвящено сравнительному анализу двух подходов решения задачи сентимент-анализа: анализа, основанного на словарях и правилах, и анализа, основанного на машинном обучении (в том числе нейросетях).

Актуальность данного исследования обусловлена растущим запросом компаний на анализ неструктурированных массивов данных, в том числе текста с точки зрения эмотивной составляющей; малым количеством работ, посвященных сравнительному анализу текста; развитием инструментария сентимент-анализа и возможностью применения сентимент-анализа в рамках рекомендательных систем и различных систем мониторинга.

1. Математическая постановка задачи сентимент-анализа

Дадим математическую постановку задачи анализа тональности текста по непрерывной шкале от -1 до 1, где -1 — негативная окраска текста, а 1 — позитивная.

Пусть X — документ (текст), предварительно подготовленный к использованию в модели (например, все слова приведены к их начальным формам). Тогда оценку текста X можно представить в виде результата применения функции $f(X)$:

$$X' = f(X). \tag{1}$$

В описанном выше случае $f(X) \in [-1, 1]$, в более сложных моделях, например, если оценка производится не по бинарной шкале «отрицательная-положительная», а на основе принадлежности текста различным классам, постановка усложняется.

Рассмотрим постановку задачи в случае определения принадлежности текста разным классам, описывающим эмоциональную окраску.

Пусть $C = \{c_1, c_2, \ldots, c_n\}$ — множество классов эмоциональной окраски, $X = \{x_1, x_2, \ldots, x_m\}$ — множество документов, а $F : C \times X \to \{0, 1\}$ — неизвестная целевая функция, описывающая принадлежность документа x_i классу c_i. Таким образом, задача сводится к построению классификатора F', максимально близкого по оценке к функции F. Для оценки точности полученного классификатора используют выборку размеченных документов $R \subset C \times D$, для которых известно значение F. Далее выборку делят на две части: обучающую и проверочную. Первую используют для построения классификатора, вторую — для проверки полученного классификатора.

2. Модели на основе словарей и правил

В основе моделей, использующих словари и правила лежит идея о том, что различные слова в различных доменных областях текста могут иметь разную оценку. Поэтому для каждой доменной области, такой как обзоры кинофильмов, отзывы о смартфонах, финансовые сводки, составляется отдельный словарь (тезаурус), представляющий набор данных в формате ключ-значение.

Правила же обычно не изменяются в случае различий между доменными областями. Правила могут представлять из себя регулярные выражения, конечные автоматы или деревья, не ограничиваясь данными структурами. В самом простом случае правила

могут управлять тональной оценкой суждения в зависимости от так называемых модификаторов. Это могут быть наречия степени, изменяющие степень оценки («самый», «очень» и другие); частица «не», меняющая оценку высказывания на противоположную и так далее. Более сложные правила могут распознавать идиомы или устойчивые выражения: фраза «но самое интересное, что...» не обязательно означает, что автор действительно испытывает интерес, а выражение «оставляет желать лучшего» имеет отрицательный окрас несмотря на то, что слова «желать» и «лучший» имеют в общем положительную оценку.

В ходе данного исследования для реализации моделей этого типа использовался словарь Linis Crowd, детально разобранный в ходе основной магистерской работы. Также были описаны базовые правила изменения оценки в зависимости от модификаторов.

3. Модель с использованием мешка слов, словаря и правил

Алгоритм перевода текста в векторное пространство выглядит следующим образом: для исходного документа строится последовательность n-грамм, в рамках каждой n-граммы происходит применение правил и оценка тональности с помощью словаря. На основе полученных результатов строится так называемый мешок слов (bag of words), который используется как входные данные классификатора на основе метода k-ближайших соседей.

Для каждой n-граммы из документа применяются правила и для каждого оценочного суждения в n-грамме строится словарь по следующему принципу: ключом является нормальная форма слова из основного словаря (в данном случае Linis Crowd), а значением — агрегированная тональность суждения на основе правил.

Валидация данной модели на двух разных корпусах документов с оптимизацией параметров классификатора дала следующие результаты:

	Корпус 1	Корпус 2
Средняя ошибка	0.61	0.33
Процент точных совпадений	46%	51%

4. Классификация с помощью многослойного перцептрона

Многослойный перцептрон (MLP) — класс нейронных сетей, состоящий минимум из трех слоев: входного, скрытого и выходного. При этом размерность входного слоя совпадает с размерностью входных данных, а размерность выходного слоя — с числом классов (в случае классификатора). MLP является одним из самых распространенных типов нейросетей, используемых для решения задач классификации.

Используя алгоритм преобразования текста из предыдущей модели и многослойный перцептрон, были получены следующие результаты:

	Linis Dataset	Kinopoisk Dataset
Процент точных совпадений	49%	58%

5. Модели на основе машинного обучения

Модели машинного обучения имеют широкое распространение в области обработки естественного языка. С их помощью решаются задачи машинного перевода, генерации текста и сентимент-анализа.

Основным способом обработки текста с помощью методов машинного обучения является перевод текста в некоторое векторное пространство заданной размерности. Существуют различные способы преобразования документа в численный вектор: простой мешок слов, мешок слов с использованием TF-IDF меры, токенизация текста с помощью заданного словаря и другие.

Модели на основе алгоритмов машинного обучения показывают высокую точность при решении задач сентимент-анализа. Среди таких методов наиболее подходящими являются рекуррентные и сверточные нейронные сети.

6. Классификация с помощью рекуррентных нейронных сетей

В силу своей архитектуры рекуррентные нейронные сети показывают высокую точность при обработке связных последовательностей,

таких как временные ряды. Представление текста в виде связной последовательности через кодирование текста на основе множества слов в корпусе документов позволяет эффективно решать задачи NLP с помощью рекуррентных нейронных сетей.

Для кодирования текста необходимо задать два параметра: размер словаря (N) и максимальную длину вектора (размерность векторного пространства — M). Далее выполняется следующая последовательность действий.

1) По корпусу определяется N наиболее часто используемых слов.
2) Каждое слово из полученного списка кодируется своим индексом $x_i \in [1, N+1]$, $i \in [0, N]$.
3) Для каждого документа слово заменяется его индексом из словаря, полученного на прошлом шаге. Если слово отсутствует — оно кодируется индексом 0.
4) Каждый вектор x, размерность которого меньше M, дополняется нулями до тех пор, пока его размерность не станет равна M.

После обучения модели были получены следующие результаты:

	Linis Dataset	Kinopoisk Dataset
Процент точных совпадений	81%	86%

7. Классификация с помощью сверточных нейронных сетей

Сверточные нейронные сети (CNN, convolutional neural network) — архитектура нейронных сетей, изначально нацеленная на эффективное распознавание визуальных образов. Ключевой идеей этой архитектуры является операция свертки: каждый фрагмент входных данных умножается на ядро свертки (матрицу) поэлементно.

Для преобразования документов в случае сверточных нейросетей хорошо зарекомендовал себя алгоритм one-hot конвертации, применяемый после токенизации по словарю. Закодировав текст с помощью алгоритма, используемого для рекуррентной нейросети, необходимо провести дальнейшее преобразование:

1) Строка матрицы с номером i отвечает за i-й элемент исходного вектора
2) Столбец матрицы с номером j является индикатором вхождения слова с индексом j в документ
3) Элемент $a_{i,j}$ принимает значение равное 1, если по индексу i в исходном векторе находилось слово с индексом j

Закодировав таким образом документы, после обучения и валидации были получены следующие результаты:

	Linis Dataset	Kinopoisk Dataset
Процент точных совпадений	65%	89%

Можно сделать вывод о том, что полученная модель показывает высокую точность на документах, имеющих общую тему и во многом общую структуру, но имеет низкую точность на наборе данных, не объединенных общей доменной областью и темой.

Заключение

Модели анализа эмоциональной тональности текста с помощью машинного обучения показывают большую точность в сравнении с моделями на основе словарей и правил. Основное преимущество моделей машинного обучения в том, что данный подход не требует составления словарей для различных доменных областей, модель сама находит закономерности в документах и использует их при классификации.

Кроме того, выбор модели сильно зависит от предметной области документов, в рамках которой будет проводиться анализ. В отличии от сверточных нейросетей, рекуррентные нейросети показывают большую точность на документах, не объединенных общей темой (таких как Linis Crowd).

Список литературы

[1] Клековкина, М. В. Метод автоматической классификации текстов по тональности, основанный на словаре эмоциональной лексики / М. В. Клековкина, Е. В. Котельников // Электронные

библиотеки: перспективные методы и технологии, электронные коллекции : материалы XIV Всероссийской научной конференции (RCDL-2012), Переславль-Залесский, 15–18 октября 2012 года. — Переславль-Залесский: Университет города Переславля, 2012. — С. 118–123. — EDN STRDWL.

[2] Pang, B. Opinion mining and sentiment analysis / B. Pang, L. Lee // Foundations and Trends in Information Retrieval. — 2008. — Vol. 2, № 1–2. — P. 1–135.

Библиографическая ссылка

Осинский, Р. Р. Сравнительный анализ разных методов исследования текста // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 130–137.

Сведения об авторах

Осинский Роман Русланович
Студент магистратуры
Направление «Прикладная математика и информатика»

УДК 519.254

Анализ временных рядов гибридным методом на основе ARIMA и XGBoost

Порядочнов Ю. А.

Тверской государственный университет

Аннотация. Данная статья посвящена анализу временных рядов гибридным методом на основе методик ARIMA и XGBoost. В рамках исследования была разработана программа, реализующая гибридный метод на языке программирования R, и были проанализированы результаты работы написанной программы.
В работе содержатся теоретические основы и результаты анализа прогнозов ARIMA, XGBoost и гибридного метода на более чем трехстах временных рядах, которые отражают стоимости различных финансовых активов.

Введение

Суть анализа временных рядов заключается в определении математико-статистических методов для выявления структуры и прогнозирования конкретных временных рядов. Таким образом, при анализе временного ряда необходимо разработать такую модель, которая будет лучше всего описывать данный временной ряд.

Разработка модели позволяет выполнять прогнозирование будущих значений. Прогнозирование необходимо для эффективного принятия решения на основе собранных данных в течение определенного периода времени.

Задача прогнозирования является неотъемлемой частью планирования во многих экономических отраслях. Так, например, прогнозирование будущих значений на основе зафиксированных в прошлом значений важно в финансовых планированиях в сфере торговли, в управлении различными видами производств.

Так как качество планирования во многом зависит от точности сделанного прогноза, требования к точности моделей, описывающих определенные временные ряды и позволяющих предсказать будущие значения на их основе, постоянно растут. Вместе с тем

высокая востребованность в экономике делает анализ временных рядов актуальной задачей.

В качестве целей рассматриваемого исследования выступают разработка и исследование работоспособности программы, реализующей методику гибридного прогнозирования временных рядов, которая основана на алгоритмах ARIMA и XGBoost.

Таким образом, для достижения поставленных целей были выполнены следующие задачи.

1) Проведен обзор и изучены теоретические основы существующих моделей и методов прогнозирования временных рядов.
2) Разработана программная реализация гибридного метода анализа временных рядов на основе алгоритмов ARIMA и XGBoost на языке программирования R.
3) Проведен анализ результатов работы написанной программы.

Рассматриваемый гибридный метод основан на двух работающих в связке моделях: ARIMA и XGBoost.

1. ARIMA

ARIMA (autoregressive integrated moving average) модель — интегрированная модель авторегрессии — скользящего среднего, разработанная Боксом и Дженкинсом в 1970 году.

Модель ARIMA имеет три составляющие:

- авторегрессия (AR);
- процесс скользящего среднего (MA);
- интегрированная составляющая.

Авторегрессия означает, что значение временного ряда в конкретный момент времени может быть выражено в виде линейной комбинации предыдущих значений этого временного ряда и случайной ошибки со свойством «белого шума».

Авторегрессионным процессом порядка p (AR(p)) называется стохастический процесс X_t, имеющий следующий вид:

$$X_t = \alpha_0 + \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + \ldots + \alpha_p X_{t-p} + \varepsilon_t,$$

где ε_t — «белый шум» с математическим ожиданием равным 0; $\alpha_0, \ldots, \alpha_p$ — свободные члены.

Процесс скользящего среднего подразумевает, что текущее значение случайного процесса представляется в виде линейной комбинации прошлых значений ошибки, по своим свойствам соответствующей «белому шуму».

Процессом скользящего среднего порядка q (MA(q)) называется стохастический процесс X_t, имеющий вид

$$X_t = \varepsilon_t - \beta_1\varepsilon_{t-1} - \beta_2\varepsilon_{t-2} - \ldots - \beta_q\varepsilon_{t-q},$$

где ε_t — «белый шум» с математическим ожиданием равным 0; $\beta_1, \ldots, \beta_q$ — свободные члены.

Комбинация процессов авторегрессии и скользящего среднего порядков p и q соответственно называется авторегрессионным процессом скользящего среднего (ARMA(p, q)).

Модель ARMA(p, q) имеет вид

$$X_t = \alpha_0 + \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + \ldots + \alpha_p X_{t-p} + \varepsilon_t - \\ - \beta_1\varepsilon_{t-1} - \beta_2\varepsilon_{t-2} - \ldots - \beta_q\varepsilon_{t-q}.$$

Интегрированная составляющая отвечает за порядок дифференциации ряда при необходимости привести ряд к стационарному. Временной ряд преобразуется взятием разности соответствующего порядка и по полученной модели строится ARMA-модель.

Следует упомянуть, что стационарные временные ряды — это временные ряды, у которых среднее значение и дисперсия постоянны во времени, то есть свойства данного типа временных рядов не зависят от времени наблюдения ряда. В свою очередь, у нестационарных временных рядов либо среднее значение, либо дисперсия, либо оба эти свойства не являются постоянными во времени.

Пусть X_t — интегрируемый нестационарный процесс порядка d. Если процесс $Y_t = \Delta^d X_t$, составленный из первых разностей d-порядка исходного процесса, является процессом ARMA(p, q), то X_t называется процессом ARIMA(p, d, q).

2. XGBoost

XGBoost (Extreme Gradient Boosting) — это алгоритм машинного обучения, впервые представленный Тяньцзи Ченом и Карлосом Гастрином в 2016 году на конференции SIGKDD.

XGBoost используется для задач обучения с учителем, то есть используются данные обучения x_i для прогнозирования целевой переменной y_i с помощью модели в таких задачах, как классификация и регрессия.

Так, например, в качестве модели может использоваться линейная модель, прогноз в которой задается как $\widehat{y}_i = \sum_j \theta_j x_{ij}$ (линейная комбинация взвешенных входных функций). Коэффициенты θ являются неопределенной частью, которую необходимо извлечь из данных.

Задача обучения модели сводится к поиску наилучших параметров θ, которые лучше всего подходят для x_i и y_i. Для обучения модели необходимо определить целевую функцию, чтобы измерить, насколько хорошо модель соответствует обучающим данным.

Целевые функции состоят из двух частей:

- потери обучения;
- члена регуляризации.

Таким образом, целевая функция имеет вид

$$TF(\theta) = L(\theta) + \Omega(\theta),$$

где L — функция потерь при обучении, Ω — член регуляризации.

Потери при обучении демонстрируют, насколько модель предсказуема по отношению к обучающим данным. В свою очередь, регуляризация контролирует сложность модели, что помогает избежать переобучения.

В основе XGBoost лежит алгоритм градиентного бустинга деревьев решений, строящий модель в виде ансамбля деревьев решений.

Модель ансамбля деревьев состоит из набора деревьев классификации и регрессии (CART). Данные деревья имеют вид

$$f(x) = w_{q(x)},\ w \in R^T,\ q\colon R^d \to \{1, 2, \ldots, T\},$$

где w — вектор очков на листьях, q — функция, назначающая каждую точку данных соответствующему листу, T — количество листьев.

В отличие от деревьев решений, содержащие только значения решений, в CART реальная оценка связана с каждым из листьев.

На практике используется модель ансамбля, которая суммирует предсказания нескольких деревьев вместе. Оценки прогнозов каждого отдельного дерева суммируются, чтобы получить окончательную оценку. Таким образом, деревья пытаются дополнять друг друга. Данная модель имеет вид

$$\widehat{y}_i = \sum_{k=1}^{t} f_k\left(x_i\right), \;\; f_k \in \mathcal{F},$$

где t — количество деревьев, f — функция в функциональном пространстве $\mathcal{F}$, $\mathcal{F}$ — набор всех возможных CART.

Оптимизируемая целевая функция имеет вид

$$TF(\theta) = \sum_{i=1}^{n} L\left(y_i, \widehat{y}_i\right) + \sum_{k=1}^{t} \Omega\left(f_k\right).$$

Суть обучения ансамбля деревьев решений заключается в определении целевой функции и ее оптимизации.

Необходимо изучить функции f_k, содержащие соответствующие структуры деревьев и оценки листьев. Обучение проходит по аддитивной стратегии последовательно, то есть происходит исправление уже обученного и добавление по одному новому дереву за раз. Таким образом, записывая значение прогноза на шаге k в виде $\widehat{y}_i^{(k)}$, имеем

$$\begin{aligned}
&\widehat{y}_i^{(0)} = 0,\\
&\widehat{y}_i^{(1)} = f_1\left(x_i\right) = \widehat{y}_i^{(0)} + f_1(x_i),\\
&\widehat{y}_i^{(2)} = f_1\left(x_i\right) + f_2\left(x_i\right) = \widehat{y}_i^{(1)} + f_2(x_i),\\
&\ldots\\
&\widehat{y}_i^{(t)} = \sum_{k=1}^{t} f_k\left(x_i\right) = \widehat{y}_i^{(t-1)} + f_t(x_i).
\end{aligned}$$

На каждой итерации добавляется то дерево, которое оптимизирует целевую функцию. В свою очередь, цель оптимизации для нового дерева на этапе k имеет вид

$$TF^{(k)} = \sum_{i=1}^{n} \left[g_i f_k(x_i) + \frac{1}{2} h_i f_k^2(x_i) \right] + \Omega(f_k),$$

где $g_i = \partial_{\widehat{y}_i^{(k-1)}} L(y_i, \widehat{y}_i^{(k-1)})$, $h_i = \partial^2_{\widehat{y}_i^{(k-1)}} L(y_i, \widehat{y}_i^{(k-1)})$.

Таким образом, можно сказать, что при обучении на каждой итерации вычисляются отклонения предсказаний уже обученного ансамбля на обучающей выборке. Следующая модель, добавленная в ансамбль, будет предсказывать эти отклонения. При добавлении предсказания нового дерева к предсказаниям обученного ансамбля уменьшается среднее отклонение модели. Новые деревья добавляются в ансамбль до тех пор, пока ошибка уменьшается либо пока не выполняется одно из правил остановки.

Регуляризация функции f в XGBoost определяется так:

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2,$$

где T — количество листьев, w — вектор очков на листьях, γ и λ — параметры регуляризации.

3. Гибридный метод

Прогнозирование временных рядов по гибридной модели ARIMA-XGBoost проходит в несколько шагов.

В первую очередь происходит прогнозирование по линейной модели. Пусть модель ARIMA будет линейной составляющей и y_t^* — это прогнозное значение в момент времени t.

Остатки линейной модели содержат только нелинейную зависимость. Далее необходимо получить эти остатки. Пусть y_t — это исходное значение в момент времени t, e_t — остаток в момент времени t из линейной модели, тогда

$$e_t = y_t - y_t^*.$$

Следующим шагом будет прогнозирование по нелинейной модели. Любой значительный нелинейный паттерн в остатках ARIMA указывает на его ограничения. Таким образом, посредством моделирования остатков ARIMA с использованием XGBoost эти нелинейные отношения могут быть выявлены. Пусть $\widehat{e}_t$ — прогнозное значение остатка в момент времени t.

Прогнозирование по гибридной модели происходит путем комбинации моделей ARIMA и XGBoost. Данная комбинация приводит к модели гибридного леса. Пусть hm_t — это прогнозное значение по гибридной модели ARIMA-XGBoost в момент времени t, тогда

$$hm_t = y_t^* + \widehat{e}_t.$$

4. Результаты

В рамках рассматриваемого исследования были проанализированы результаты работы программы, реализующей методику гибридного прогнозирования ARIMA-XGBoost, на более чем трехстах временных рядах, отражающих состояние различных акций организаций, курс валют и тому подобное.

В основном гибридный метод не дает лучший результат точности прогнозирования таких временных рядов, график которых схож с графиком, представленным на рис. 1, по сравнению с ARIMA и XGBoost по отдельности. Значения средней абсолютной процентной ошибки MAPE для прогнозов ARIMA, XGBoost и гибридного метода данного временного ряда равны 0.242113%, 0.242115% и 0.242114% соответственно.

Таким образом, гибридный метод не дает преимущества в точности прогнозирования временных рядов с достаточно малым количеством выбросов по сравнению с другими рассматриваемыми методами.

Отметим, что гибридный метод наиболее эффективно выполняет прогнозирование временных рядов подобных временному ряду, представленному на рис. 2. Значения средней абсолютной ошибки в процентах прогнозов ARIMA, XGBoost и гибридного метода данного временного ряда равны 1.6927356%, 1.445148% и 1.068735% соответственно.

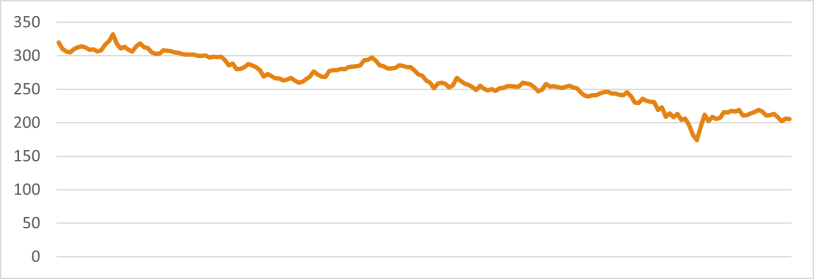

Рис. 1. Стоимость одной из акций, рассматриваемой в ходе исследования.

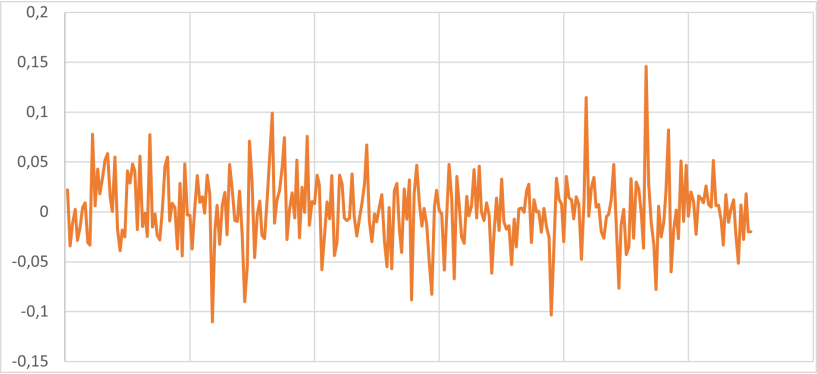

Рис. 2. Доходность одной из акций, рассматриваемой в ходе исследования.

Анализ временных рядов, графики которых имеют сходство с графиком, представленным на рис. 2, показывает, что в сравнении с методами ARIMA и XGBoost гибридный метод является более устойчивым к нерегулярным случайным достаточно большим выбросам и даёт лучший результат по точности прогнозирования при их наличии.

Заключение

В ходе исследования, рассматриваемого в данной статье, были выявлены закономерности, при которых точность прогнозирования того или иного метода будет лучше по сравнению с другими рассматриваемыми методами.

ARIMA лучше всего будет подходить для работы с временными рядами, значения которых изменяются плавно, то есть при отсутствии выбросов. XGBoost лучше всего прогнозирует временные ряды с регулярно возникающими продолжительными выбросами. В свою очередь, гибридный метод ARIMA-XGBoost показывает лучшую точность прогнозирования временных рядов с резкими нерегулярными непродолжительными выбросами.

Список литературы

[1] Агапова, Е. Г. Анализ временных рядов : учебное пособие / науч. ред. Т. М. Попова. — Хабаровск : Изд-во Тихоокеан. гос. ун-та, 2015. — 55 с.

[2] Джеймс, Г. Введение в статистическое обучение с примерами на языке R / Г. Джеймс, Д. Уиттон, Т. Хасти, Р. Тибширани ; пер. с англ. С. Э. Мастицкого. — М. : ДМК Пресс, 2016. — 450 с. : ил.

[3] Карпенко, Н. В. Эконометрика. Анализ и прогнозирование временного ряда : учебное пособие. — М. : РУТ (МИИТ), 2018. — 132 с.

[4] Подкорытова, О. А. Анализ временных рядов : учебное пособие для бакалавриата и магистратуры / О. А. Подкорытова, М. В. Соколов. — 2-е изд., перераб. и доп. — М. : Издательство Юрайт, 2017. — 267 с.

[5] ARIMA Modeling with R : [сайт]. — URL: https://www.listendata.com/2015/10/arima-modeling-with-r.html (дата обращения: 11.03.2022).

[6] Introduction to Boosted Trees : [сайт]. — URL: https://xgboost.readthedocs.io/en/latest/tutorials/model.html (дата обращения: 04.03.2022).

[7] Introduction to Forecasting with ARIMA in R : [сайт]. — URL: https://blogs.oracle.com/ai-and-datascience/post/

introduction-to-forecasting-with-arima-in-r (дата обращения: 11.03.2022).

[8] Time Series Analysis Using ARIMA Model In R : [сайт]. — URL: https://datascienceplus.com/time-series-analysis-using-arima-model-in-r/ (дата обращения: 04.03.2022).

[9] Understand your dataset with XGBoost : [сайт]. — URL: https://xgboost.readthedocs.io/en/latest/R-package/discoverYourData.html (дата обращения: 04.03.2022).

[10] XGBoost R Tutorial : [сайт]. — URL: https://xgboost.readthedocs.io/en/latest/R-package/xgboostPresentation.html (дата обращения: 04.03.2022).

Библиографическая ссылка

Порядочнов, Ю. А. Анализ временных рядов гибридным методом на основе ARIMA и XGBoost // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 138–147.

Сведения об авторах

Порядочнов Юрий Алексеевич
Студент магистратуры
Направление «Прикладная информатика»

УДК 004.855.5

Применение нейронных сетей в анализе временных рядов

Ромашко Р. Г.

Тверской государственный университет

Аннотация. В статье предложен подход к решению задачи декомпозиции временных рядов с использованием рекуррентных и сверточных нейронных сетей, обученных на искусственно сгенерированных данных. Анализ работы метода проведен в сравнении с методом STL. Выявлены как положительные результаты, так и недочеты. На данный момент метод требует доработок.

Введение

Явления, которые важно изучать с точки зрения их развития и изменения во времени, встречаются во многих областях. Это могут быть характеристики технических систем, показатели природных и социально-экономических процессов. Например, динамика курса валюты, динамика продаж компании, или же данные о потреблении природных ресурсов. Во всех этих прикладных задачах приходится иметь дело с временными рядами и проводить их анализ. Одной из задач такого анализа является задача декомпозиции временного ряда, то есть разложения его на составные компоненты [2]. Рассмотрим эти компоненты подробнее.

Тренд (T) — компонента, показывающая общую тенденцию изменения наблюдаемого значения. Визуально представляется гладкой линией, то есть различного рода колебания или случайные «скачки» в данных в тренд не входят.

Сезонность (S) — компонента, которая возникает тогда, когда наблюдаемое значение имеет зависимость от периодических факторов, например, от времени года или от дня недели. Всегда имеет фиксированную и известную частоту.

Цикличность — компонента, возникающая, когда в данных прослеживается повторение и роста, и снижения, не имеющее при

этом фиксированной частоты (в этом заключается отличие от сезонности). В качестве примера подходит экономический кризис. Обычно цикличность и тренд объединяют и считают это одной компонентой — трендом.

Случайная компонента (R) — сюда попадают все значения, которые в процессе анализа не удается отнести ни к одной из компонент выше.

Тогда, формально, очередное значение временного ряда Y_i может считаться композицией значений данных компонент.

Принято различать две модели композиции [2]: аддитивная и мультипликативная. Первая имеет место тогда, когда сезонность временного ряда не зависит от значения тренда и зависит лишь от времени. Она имеет вид

$$Y_i = T_i + S_i + R_i.$$

Мультипликативная же модель, напротив, возникает тогда, когда сезонность изменяется пропорционально тренду. Иными словами, является функцией не только от времени, но еще и от значения тренда. Для такой модели композиция будет следующая:

$$Y_i = T_i \cdot S_i \cdot R_i.$$

На текущий момент существует немало статистических методов решения задачи декомпозиции, среди которых классический метод, использующий скользящее среднее, производный от него метод X11, или, один из наиболее популярных — STL. Каждый метод может и применяться успешно, и быть неэффективным, в зависимости от разновидностей временных рядов. Например, X11 предназначен исключительно для декомпозиции данных, содержащих либо месячный, либо квартальный паттерн сезонности [2].

В данном случае, интерес представляет собой исследование альтернативного подхода для решения данной задачи — применение нейросетевых моделей. О существовании подобных методов на данный момент нет практически никакой информации. Также, использование статистических методов осложняется их количеством и особенностями — есть риск подобрать неподходящий метод. В то же время, подход с точки зрения нейросети может представлять собой более универсальное решение.

1. Подготовка данных

В первую очередь необходимо сформировать данные для обучения. Эти данные должны содержать в себе исходную последовательность (она будет являться входом для модели), а также ее отдельные компоненты, по которым модель и будет обучаться. Для формирования такого набора воспользуемся вспомогательным генератором временных рядов. Он искусственно генерирует отдельные компоненты и композирует их в итоговый временной ряд.

Для тренда возьмем за основу модель, описанную в книге [1]. Сезонность будет смоделирована через функцию синуса. Итоговая модель для генерации [1]:

$$y_t = \mu_t + \gamma_t + \varepsilon_t, \quad \varepsilon_t \sim N(0, 30), \quad t \geqslant 1.$$

Тренд:

$$\mu_t = \mu_{t-1} + \beta_{t-1}, \quad \mu_0 = 0, \quad t \geqslant 1.$$

Дрейф тренда:

$$\beta_t = \beta_{t-1} + \zeta_t, \quad \zeta_t \sim N(0, 0.15), \quad \beta_0 = 0, \quad t \geqslant 1.$$

Сезонность:

$$\gamma_t = 50 \sin\left(\frac{2\pi t}{T}\right), \quad T \in \{7, 30, 90\},$$
$$P(T = 7) = P(T = 30) = P(T = 90), \quad t \geqslant 1.$$

Также будем хранить отдельно последовательность, предварительно лишенную тренда (то есть сезонность + случайная компонента).

На рис. 1 приведен пример сгенерированных данных.

Следующим этапом необходимо нормализовать данные, приведя их значения в единый диапазон. Возьмем в качестве такого диапазона $[-0.8; 0.8]$. Тогда для каждой последовательности S, $|S| = n$, получаем нормализацию:

$$S_i = 1.6 \frac{S_i - S_{\min}}{S_{\max} - S_{\min}} - 0.8,$$
$$S_{min} = \min\{S_1, \ldots, S_n\},$$

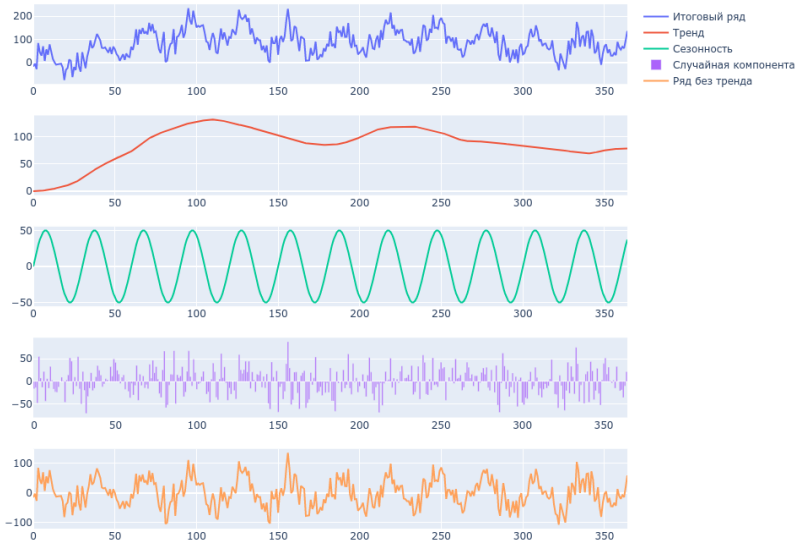

Рис. 1. Пример сгенерированного временного ряда.

$$S_{max} = \max\{S_1, \ldots, S_n\}.$$

В текущей реализации, для обучения использовалось порядка 15 000 сгенерированных примеров.

2. Архитектура нейросети

Работа с временными рядами — это работа с последовательностями. В первую очередь, для работы с последовательностями используется модель рекуррентной нейронной сети (RNN) [3]. Ее концепция заключается в последовательной обработке информации. В традиционных нейронных сетях подразумевается, что все входы и выходы независимы. Но для многих задач это не подходит. В частности, если требуется сделать вывод об очередном элементе временного ряда, нужно учитывать предшествующие ему значения.

RNN выполняют одну и ту же задачу для каждого элемента

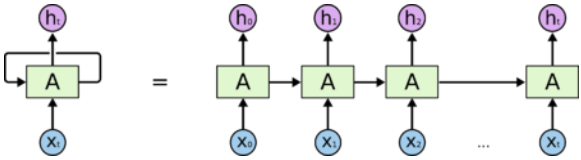

Рис. 2. Принцип работы рекуррентного слоя.

последовательности, причем выход зависит от предыдущих вычислений.

Наиболее часто используемым типом RNN являются LSTM, которые могут хранить долгосрочные зависимости, с чем у классических RNN возникают проблемы. LSTM имеет возможность хранить нужную информацию в специальной ячейке памяти, каждый раз решая, какую часть этой памяти нужно стереть, а какую сохранить. То есть, для очередного ответа, LSTM использует входные данные, ответ на предыдущем шаге и данные, хранящиеся в ячейке памяти.

Другим типом нейросетей, реже, но все же используемых для обработки последовательностей, являются сверточные сети (CNN) [3]. Изначально, такая архитектура создавалась для работы с изображениями, например, для распознавания образов. В ее основе лежит операция свертки, суть которой заключается в поэлементном умножении каждого фрагмента (то есть некоторой окрестности) входной матрицы на матрицу свертки (ядро), после чего результат суммируется и пишется в соответствующую позицию выходной матрицы (рис. 3). В частном случае, свертка может применяться и для последовательностей.

Архитектура сети в текущей работе следующая.

1) Слой LSTM. Для каждого элемента входной последовательности вычисляет следующую функцию:

$$
\begin{aligned}
i_t &= \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}),\\
f_t &= \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}),\\
o_t &= \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}),\\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{ic}x_t + b_{ic} + W_{hc}h_{t-1} + b_{hc}),\\
h_t &= o_t \odot \tanh(c_t),
\end{aligned}
$$

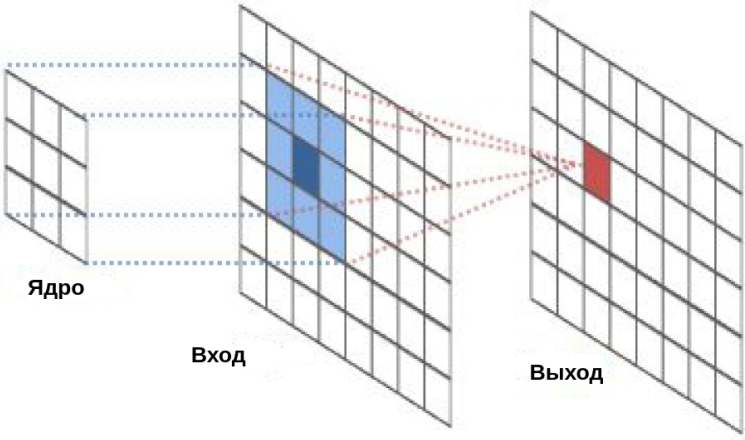

Рис. 3. Принцип работы свертки.

где c_t — значение в ячейке памяти в момент t, h_t — скрытое состояние (выход нейрона) в момент t, i_t, f_t, o_t — значения входного узла, узла забвения и выходного узла, соответственно, b — смещение (bias), σ — функция сигмоиды, $\odot$ — операция поэлементного умножения [4].

2) Выходная последовательность LSTM далее проходит через сверточный слой с размером ядра 30. Это способствует дополнительному «сглаживанию» данных.

Для обучения используется метод обратного распространения ошибки. Потери определяются при помощи среднеквадратической ошибки (MSE). Коэффициент скорости обучения — 0.0005.

На основе такой архитектуры обучено две модели. Первая обучена выделять тренд в исходном временном ряде, вторая — выделять сезонность в ряде, который предварительно был лишен тренда.

3. Процесс декомпозиции

Декомпозиция ряда будет проводиться последовательно: сначала выделим тренд, далее сезонность, и все, что останется — будем считать случайной компонентой. Важно отметить, что именно выделение тренда должно идти первым этапом, так как его наличие говорит о нестационарности временного ряда, и выделение сезонного паттерна в нестационарных данных было бы для нейросети нетривиальной задачей. Поэтому, отделив в первую очередь тренд, мы приведем данные к стационарному виду, тем самым упростив задачу определения сезонности.

Также допустим некоторые упрощения для текущей реализации метода. Во-первых, будем сами решать, какой тип декомпозиции использовать (аддитивную\мультипликативную). Во-вторых, будем сами определять наличие в данных сезонности и необходимость отделять ее.

По итогу, получается следующий алгоритм.

1) Вычислить минимум и максимум временного ряда.
2) Нормализовать ряд.
3) Выделить тренд (используя первую модель).
4) Отделить тренд от исходных данных, учитывая тип декомпозиции.
5) Выделить сезонность (используя вторую модель) — опционально.
6) Остаток считать за случайную компоненту.
7) Денормализовать компоненты — привести их значения в исходный диапазон значений, используя вычисленные до этого минимум и максимум.

4. Примеры работы. Анализ результатов

Рассмотрим пару примеров работы метода, сравнивая с декомпозицией при помощи STL.

Первый пример: периодические данные с линейно растущим трендом.

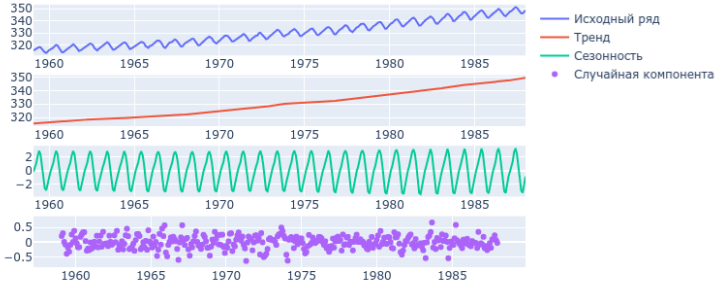

Рис. 4. Первый пример — декомпозиция STL.

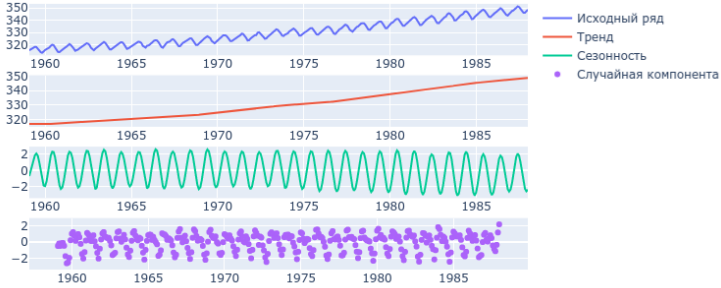

Рис. 5. Первый пример — декомпозиция нашим методом.

Результат работы STL приведен на рис. 4. Результат работы нашего метода приведен на рис. 5.

Средняя абсолютная ошибка для тренда: ∼0.21, что в масштабах значений весьма неплохо.

Средняя абсолютная ошибка для сезонности: ∼0.86, что в данном случае серьезная разница.

Второй пример: данные об измерениях объемов производства электрооборудования в ЕС.

Результат работы STL приведен на рис. 6. Результат работы нашего метода приведен на рис. 7.

Средняя абсолютная ошибка для тренда: ∼0.78 — опять же, в масштабах значений ошибка довольно мала. Визуально хорошо

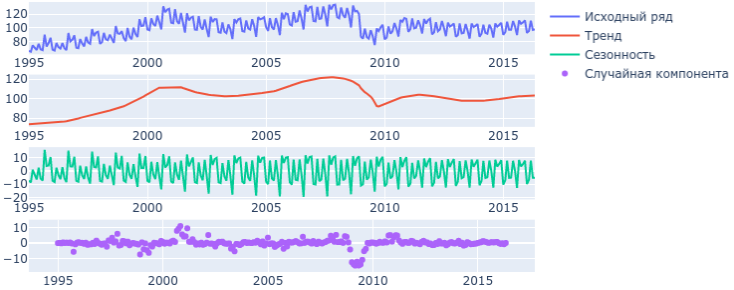

Рис. 6. Второй пример — декомпозиция STL.

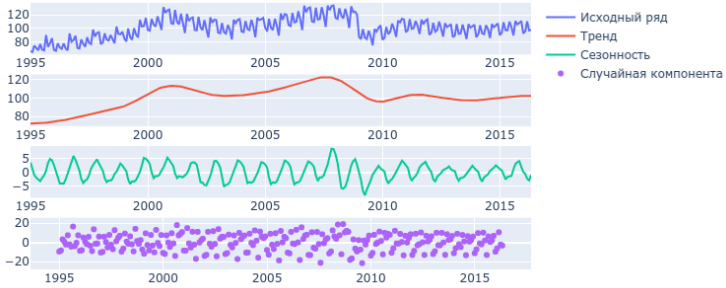

Рис. 7. Второй пример — декомпозиция нашим методом.

заметно сходство трендов.

Средняя абсолютная ошибка для сезонности: ∼6.85 — колоссальная разница, что опять же видно и по графикам.

Заключение

В ходе работы удалось достичь хороших результатов в извлечении тренда. По примерам видно, что модель, в целом, успешно подстраивается под любую форму линии тренда. Однако в декомпозиции сезонной компоненты были обнаружены серьезные проблемы. Скорее всего, здесь изначально ошибочен слишком упрощенный подход к обучению и обучающим данным: модель для сезонности

тренировалась лишь на функции синуса с дискретным набором периодов $(7, 30, 90)$ — это привело к переобучению. На втором примере наглядно видно, что сезонность в нашем методе очень напоминает тригонометрическую функцию, хотя в действительности, в этих данных сезонность имеет совершенно другой паттерн. Исследование в сторону того, как улучшить сезонную декомпозицию на данный момент является приоритетной задачей по модернизации метода. Также, большой интерес представляет автоматизация при помощи нейросетей этапа определения наличия сезонности в данных.

Список литературы

[1] Harvey, A. C. Forecasting, Structural Time Series Models and the Kalman Filter. — London : Cambridge University Press, 1990. — 574 p.

[2] Hyndman, R. J. Forecasting: principles and practice. — Melbourne : OTexts, 2018. — 382 p.

[3] Ian, G. Deep Learning / G. Ian, B. Yoshua, C. Arron. — Cambridge : The MIT Press, 2016. — 800 p.

[4] PyTorch — LSTM : [сайт]. — URL: https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html (дата обращения: 05.05.2022)

Библиографическая ссылка

Ромашко, Р. Г. Применение нейронных сетей в анализе временных рядов // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 148–157.

Сведения об авторах

Ромашко Роман Григорьевич
Студент магистратуры
Направление «Прикладная математика и информатика»

УДК 519.685

Модифицированная модель наведения типа «перехватчик-цель»

Седаков Н. М.

Тверской государственный университет

Аннотация. В работе решается задача формирования общей траектории наведения на воздушную цель, включающая горизонтальную и вертикальную проекции. Для этого траектория представляется в виде набора отдельных участков, состав и параметры которых определяются согласно выбранному методу наведения и профилю полета, а также приводится схема процесса наведения, связывающая проекции, в которой вводятся специальные участки корректировки курса движения. Для этих участков разработан алгоритм численной схемы для учета текущего курса движения при наведении прямолинейными методами, а также приведена реализация основного метода наведения — «Маневр с двумя разворотами».

Введение

Задача определения траектории перехвата воздушной цели заключается в формировании пространственной параметрической кривой, которую описывает центр масс летательного аппарата (ЛА), которая выводит перехватчик в заданное, относительно цели, положение в пространстве. Современные подходы решения подразумевают разделение задачи на две независимые проекции — горизонтальную (ГП) и вертикальную (ВП). В первой, задача решается относительно вывода перехватчика в область, удовлетворяющую условиям перехвата. Во второй, определяется профиль траектории выхода ЛА на подходящие для перехвата значения высоты и скорости движения. Выход на заданные значения производится с использованием базовых программ набора высоты (ПНВ), которые определяют профиль полета в вертикальной плоскости. В зависимости от конкретных условий используют предварительно рассчитанные ПНВ, которые являются исходными данными.

Текущие решения в ГП основываются на методах наведения, которые предполагают постоянную скорость движения и одинаковую высоту цели и перехватчика, в то время как в ВП значения скорости и высоты полета постоянно меняются. Из-за чего возникает задача совмещения двух решений. Целью данной работы является разработка схемы наведения, которая строит общую траекторию перехвата. Для достижения цели, в состав ПНВ вводятся участки корректировки курса движения, выводящие ЛА на наикратчайшую траекторию сближения с целью. Для этого приводится алгоритм расчета необходимого участка разворота.

1. Постановка задачи

В соответствии с закономерностями перемещения, описанными в работе [2], при которых взаимное расположение перехватчика и цели определяется курсовыми углами и дальностью, положения в пространстве воздушных средств представляются параметрами: $P = (x, y, z)$ — координаты положения в местной декартовой СК [м]; v — скорость движения [м/сек]; К — курс движения [рад]; ω — угловая скорость [рад/сек]; θ — угол вертикального наклона [рад]. Обозначим $\text{П}(t) = \{\text{Р}_\text{п}, v_\text{п}, \text{К}_\text{п}, \omega_\text{п}, \theta_\text{п}\}$ и $\text{Ц}(t) = \{\text{Р}_\text{ц}, v_\text{ц}, \text{К}_\text{ц}, \omega_\text{ц}, \theta_\text{ц}\}$ — соответственно характеристики положения перехватчика и цели в момент времени t.

Необходимо по имеющимся начальным характеристикам $\text{П}^{(0)}$ и $\text{Ц}^{(0)}$ построить траекторию перехвата, удовлетворяющую следующим требованиям:

- движение перехватчика и цели рассматривается с учетом сведения временного баланса, то есть время полета цели до точки перехвата совпадает со временем выполнения траектории перехвата;

- требуется выполнение координатного баланса, который заключается в том, что перехватчик в конечной точке наведения находится в заданном относительно цели положении в пространстве. Положение определяется множеством управляющих параметров $\text{У} = \{\text{Д}_\text{к}, p_\text{к}, v_\text{к}, \gamma\}$, где $\text{Д}_\text{к}$ — конечная дальность до цели [м]; $v_\text{к}$ — скорость в конце наведения[м/сек]; $p_\text{к}$ — угол

выхода на цель в конце наведения [рад]; γ — угол крена выполнения виража.

Рассматривая сближение перехватчика с целью в поставленной задаче наведения, будем исходить из некоторых предположений относительно движения ЛА:

- воздушная цель не маневрирует на протяжении всей траектории наведения, то есть в любой момент времени движется прямолинейно и с постоянной скоростью: $V_{\text{ц}} = const$, $\omega_{\text{ц}} = 0$;

- движение ЛА рассматривается как движение материальной точки, без учета действующих сил.

В изложенной формулировке задача определения параметров траектории движения перехватчика является задачей из области механики, именуемой кинематикой.

Траекторию будем представлять, как множество участков движения, которые ЛА способен выполнить последовательно, то есть значение $\Pi^{(t)}$ в момент окончания одного из участков совпадает со значением $\Pi^{(t+1)}$ — в момент начала следующего участка. Траектория может быть представлена участками следующих типов:

- участок прямолинейного движения представляется параметрами: $P_0 = (x_0, y_0, z_0)$ — координаты начала [(м, м, м)]; k — курс направления движения [рад]; l — длина участка [м];

- участок маневра представляется параметрами: R — радиус разворота [м]; $P_{\text{о}} = (x_{\text{о}}, y_{\text{о}}, z_{\text{о}})$ — координаты центра [(м, м, м)]; α — величина угла разворота [рад]; l — длина участка [м];

- участок ПНВ определяется параметрами: $P_0 = (x_0, y_0, z_0)$ — координаты начала [(м, м, м)], k — курс направления движения [рад]; l — длина участка [м]; h_1 — начальная высота; h_2 — конечная высота; v_1 — начальная скорость; v_2 — конечная скорость. Разделяют 4 разновидности участков ПНВ: участок набора высоты, разгона, снижения и торможения.

2. Наведение в горизонтальной плоскости

Основным методом наведения в горизонтальной плоскости считается «маневр с двумя разворотами» [3], поскольку данный метод позволяет вывести ЛА в наиболее выгодное для осуществления перехвата положение, учитывая все параметры из множества У. Траектория наведения данным методом состоит из участков:

- участок первого маневра для выхода на курс прямолинейного движения в точку упреждения;
- участок прямолинейного движения в точку начала маневра для выхода на цель;
- участок второго маневра для выхода на цель с заданного направления.

В работе используется точное решение задачи, изложенное в источнике [3]. Для вычисления искомых параметров составляется система уравнений (1), относительно угла наклона (y_1) и длины (C) прямолинейного участка, описывающая совместное изменение координат перехватчика и цели в ходе выполнения участков метода.

$$\begin{cases} \text{Б} - m_1 R_1 (\cos\gamma_1 - \cos\gamma_0) - C\sin\gamma_1 - \\ \qquad - m_2 R_2 (\cos p - \cos\gamma_1) - \text{Д}_\text{к} \sin p = 0, \\ \text{П} - m_1 R_1 (\sin\gamma_1 - \sin\gamma_0) + C\cos\gamma_1 - \\ \qquad - m_2 R_2 (\sin p - \sin\gamma_1) + \text{Д}_\text{к} \cos p = \\ \qquad = R_1 a_1 \alpha_1 + C a_1 + R_2 a_2 \alpha_2. \end{cases} \tag{1}$$

Здесь m_i — направление разворота i-го маневра; α_i — величина угла разворота i-го маневра; y_0 — угол наклона вектора скорости ЛА в начале наведения; $a_1 = \dfrac{v_\text{ц}}{v_\text{п}}$; $a_2 = \dfrac{2v_\text{ц}}{v_\text{п} + v_\text{к}}$.

Система (1) имеет аналитическое решение в случае $R_1 = R_2 = 0$:

$$\gamma_{1_{(1,2)}} = \arccos\left(\frac{-2se \pm \sqrt{4s^2e^2 - 4(d_2^2 + e^2)(s^2 - d_2^2)}}{2(d_2^2 + e^2)}\right), \tag{2}$$

где $e = -\text{Б} + \text{Д}_\text{к} \sin p$, $d_2 = -\text{П} - \text{Д}_\text{к} \cos p$, $s = a_1 (\text{Б} - \text{Д}_\text{к} \sin p)$; П, Б — координаты ЛА в СК цели, используемой в методе.

Решение (2) берется со знаком плюс, если цель движется в направлении перехватчика, что соответствует случаю $\text{П} \geqslant 0$, и со знаком минус — в случае $\text{П} < 0$.

Решение для реального значения радиуса разворота (R_1 и R_2) определяется с помощью метода ведущего параметра. Параметры радиусов постепенно получают некоторое приращение ΔR, после чего вычисляется значение дифференциала $\dfrac{d\gamma_1}{dR}$ и значение текущего параметра γ_1. В выражении (3) начальное значение $\gamma_{1(0)}$ определяется согласно (2):

$$\gamma_{1(k+1)} = \gamma_{1(k)} + \frac{d\gamma_1}{dR} \Delta R. \tag{3}$$

3. Модификация прямолинейных методов

К классу прямолинейных методов наведения относятся методы «Параллельное сближение» и «Прямой перехват» [1, 2]. Основная отличительная особенность методов этого класса заключается в постоянном направлении движения в упреждающую точку встречи. Однако на практике наиболее вероятна ситуация, когда текущий курс движения ЛА не совпадает с потребным курсом метода наведения. В таком случае, самолету требуется совершить дополнительный маневр для выхода на этот курс.

Для исправления описанного недостатка предлагается модифицировать метод наведения, добавив в расчеты параметр текущего курса перехватчика и первоначальный участок разворота, предназначенный для выхода ЛА на потребный курс.

Предлагаемое решение основывается на численной схеме. Задачу сведем к поиску такого угла разворота α, при котором вычисленные после выполнения разворота текущий курс ЛА и потребный курс перехвата будут совпадать. В общем виде, функционал, корень которого является решением, имеет вид

$$F(\alpha) = \text{К}_\text{п}^\alpha - f(\text{П}^\alpha, \text{Ц}^\alpha), \tag{4}$$

где под f понимается алгоритм расчета курса движения согласно прямолинейному методу, $\text{К}_\text{п}^\alpha$ — курс перехватчика после выполне-

ния разворота на угол α, Π^{α} и $Ц^{\alpha}$ — характеристики положения перехватчика и цели после выполнения разворота на угол α.

Для упрощения вида функции решения задачи и самих вычислений задача решается в СК, связанной с направлением движения перехватчика:

$$\begin{cases} z'_{ц} = (Z_{ц} - Z_{п})\cos(2\pi - К_{п}) + (X_{ц} - X_{п})\sin(2\pi - К_{п}), \\ x'_{ц} = -(Z_{ц} - Z_{п})\sin(2\pi - К_{п}) + (X_{ц} - X_{п})\cos(2\pi - К_{п}), \\ К'_{ц} = К_{ц} - К_{п}. \end{cases}$$

Для расчетов значений Π^{α} и $Ц^{\alpha}$ необходимо вычислить параметры подходящего участка разворота. Центр разворота вычисляется так, чтобы текущий вектор скорости являлся касательной к окружности виража. Угол между прямой, перпендикулярной вектору скорости, на которой располагается центр разворота, и осью абсцисс вычисляется по формуле

$$k_g = \operatorname{arctg}\left(\frac{-1}{\operatorname{tg}(2\pi - (К_{п} - \frac{\pi}{2}))}\right).$$

Координаты центра разворота рассчитываются по формуле

$$(x_0, y_0, z_0) = \begin{cases} \begin{pmatrix} x_{п} + R \cdot \sin(k_g) \cdot \operatorname{sign}(\gamma), y_{п}, \\ z_{п} + R \cdot \cos(k_g) \cdot \operatorname{sign}(\gamma) \end{pmatrix}, & \text{если } \frac{\pi}{2} \leqslant \alpha \leqslant \frac{3\pi}{2}, \\ \begin{pmatrix} x_{п} - R \cdot \sin(k_g) \cdot \operatorname{sign}(\gamma), y_{п}, \\ z_{п} - R \cdot \cos(k_g) \cdot \operatorname{sign}(\gamma) \end{pmatrix}, & \text{иначе,} \end{cases}$$

$$R = \frac{v_{п}^2}{G \cdot \operatorname{tg}(|\gamma|)}. \tag{5}$$

В полученной СК курс выхода с виража определяется как

$$К_{п}^{\alpha} = \begin{cases} \alpha, & \text{направление разворота по часовой,} \\ (2\pi - \alpha), & \text{иначе.} \end{cases}$$

Координаты после выполнения разворота:

$$\begin{cases} z_{п}^{\alpha} = z_0 + R\cos(\psi_{oz}(К_{п}^{\alpha})), \\ x_{п}^{\alpha} = x_0 + R\sin(\psi_{oz}(К_{п}^{\alpha})), \end{cases}$$

$$\begin{cases} z_{\text{ц}}^{\alpha} = z'_{\text{ц}} + \dfrac{v_{\text{ц}}}{v_{\text{п}}} \cdot \alpha \cdot R\cos(\psi_{oz}(\text{К}'_{\text{ц}})), \\ x_{\text{ц}}^{\alpha} = x'_{\text{ц}} + \dfrac{v_{\text{ц}}}{v_{\text{п}}}\alpha \cdot R\sin(\psi_{oz}(\text{К}'_{\text{ц}})), \end{cases}$$

где $\psi_{oz}(\text{К})$ — угол между вектором скорости направленным по курсу К и осью абсцисс, отсчитываемый против часовой стрелки.

Для вычисления корня функции (4) используется метод хорд. Для достижения точность $\varepsilon = 10^{-3}$ в среднем достаточно 2–3 итераций. Поскольку угол возможного разворота принимает значения на отрезке от 0 до 2π, на этом же отрезке и будет производиться поиск приближенного значения корня. В качестве начального приближения α_0 берется значение разворота, необходимого для выхода на курс $f(\text{П}, \text{Ц})$. Итерационный процесс продолжается, пока выполняется условие $|\alpha_i - \alpha_{i-1}| > \varepsilon$. Согласно методу

$$\alpha_i = \alpha_{i-1} - \frac{F(\alpha_{i-1})}{F(b) - F(\alpha_{i-1})}(b - \alpha_{i-1}).$$

4. Корректировка курса при выполнении ПНВ

Изначально предполагается выполнение ПНВ по некоторому фиксированному, начальному значению курса движения. В последствии, для минимизации времени наведения, целесообразно изменять некоторым образом курс для сближения с целью.

Для минимизации расстояния до цели по окончанию ПНВ потребуем, чтобы перехватчик двигался в упреждающую точку встречи с целью по наикратчайшей траектории, что соответствует прямолинейному участку в горизонтальной проекции. Разворот на требуемый курс движения целесообразней всего делать перед участками разгона, поскольку из выражения (5), связывающее радиус разворота и скорость движения, видно, что с ростом скорости увеличивается и траектория разворота.

В основе расчета лежит модифицированный алгоритм «Прямого перехвата», который учитывает текущий курс движения и допускает задание конечной дальности до цели. Возможность варьировать значения дальности позволяет также влиять на курс. Положим

$$\text{Д}_{\text{к}}^{*} = \text{Д}_{\text{к}} + l_{\text{оп}},$$

где $l_{\text{оп}} = v_{\text{ср. гор}} \cdot T_{\text{ПНВ}}$ — дальность опережения, $v_{\text{ср. гор}}$ — горизонтальная составляющая средней скорости выполнения ПНВ, $T_{\text{ПНВ}}$ — время оставшейся части ПНВ.

Тогда упреждающая точка будет выбрана так, чтобы учесть разницу в значении скорости, используемой в расчетах, и изменяющегося значения скорости при выполнении ПНВ.

В результате, общая схема наведения выглядит следующем образом. Вначале ЛА выполняет участки ПНВ, с периодическим корректированием курса движения, для которого выполняется экстраполирование положения цели на значение прошедшего времени наведения, чтобы вычислить актуальное значение $Ц^t$. Затем, следует выполнение участка разворота и продолжение участков ПНВ. После того как ПНВ будет выполнена, экстраполируется положение цели на момент окончания ПНВ и подключаются методы наведения в ГП, использующие актуальные характеристики цели в расчетах. Формируется окончательная траектория сближения с целью и вычисляются соответствующие участки движения, выводящие перехватчик в заданные условия перехвата.

Заключение

В работе рассмотрена задача построения общей траектории перехвата, подразумевающая совместное наведение в горизонтальной и вертикальной проекциях. Для этого использовалось представление траектории как множество отдельных последовательных участков, среди которых были введены специальные участки корректировки курса движения в структуру ПНВ. Результатом работы являются разработанные методы прямолинейного наведения, учитывающие текущий курс движения перехватчика, и схема наведения, позволяющая рассчитать всю траекторию за один цикл решения задачи.

Список литературы

[1] Дуров, В. Р. Боевое применение и боевая эффективность истребителей-перехватчиков. — М. : Воениздат, 1972. — 279 с.

[2] Паньков, С. Я. Теория и методика управления авиацией : учебное пособие. В 2 ч. Ч. 1 / С. Я. Паньков, Ю. Е. Забураев,

А. М. Матвеев ; под общ. ред. В. А. Мещерякова. — Ульяновск : УВАУ ГА, 2006. — 209 с.

[3] Феликсон, А. Е. Метод точного решения задачи ближнего наведения с двумя разворотами // Вестник Концерна ВКО «Алмаз-Антей». — 2018. — №4. — С. 91–99.

Библиографическая ссылка

Седаков, Н. М. Модифицированная модель наведения типа «перехватчик-цель» // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 158–166.

Сведения об авторах

Седаков Никита Михайлович
Студент бакалавриата
Направление «Прикладная математика и информатика»

УДК 330.322 + 519.685

Многоцелевой выбор инвестиционных проектов в условиях стохастической неопределенности

Сергеева Е. А.

Тверской государственный университет

Аннотация. В статье исследуется задача двухкритериальной оценки и выбора многопериодных инвестиционных проектов в условиях стохастической неопределенности. В основу постановки и решения задачи положена концепция ожидаемой полезности. Приводится обоснование и алгоритм построения двухкритериальной функции полезности, отражающей предпочтения инвестора в пространстве противоречивых критериев, а именно критерия максимума остаточной стоимости и максимума уровня дохода. Приводится иллюстративный пример применения предлагаемой модели выбора проектов.

Введение

Используемые в настоящее время на практике модели выбора многопериодных инвестиционных проектов ориентированы, как правило, на учет лишь одного целевого показателя и на детерминированные прогнозные процентные ставки по инвестированию и заимствованию [2]. Кроме того, представленная в данных моделях информация о предпочтениях инвестора не обеспечивает адекватного учета индивидуальных предпочтений конкретного инвестора и может быть получена лишь с большими погрешностями.

Отмеченные обстоятельства обуславливают актуальность задачи развития моделей многоцелевого анализа многопериодных инвестиционных проектов в условиях стохастической неопределенности.

Целью работы является повышение обоснованности выбора предпочтительных многопериодных инвестиционных проектов в условиях стохастической неопределенности относительно процентных ставок путем совокупного учета критериев максимума остаточной

стоимости и максимума уровня дохода, а также использования вероятностного распределения максимальной энтропии процентных ставок.

В основу постановки и решения задачи положены аксиомы рационального поведения инвестора и взаимная независимость учитываемых критериев по предпочтению, что обуславливает существование аддитивной функции полезности [1]. Оценка параметров функции полезности базируется на алгоритме решения задачи компенсации [3]. Выбор предпочтительного проекта проводится согласно концепции ожидаемой полезности, определяемой для каждого анализируемого проекта статистической оценкой по генерируемым реализациям учитываемых критериев.

1. Постановка задачи

Пусть задано конечное множество $I = \{1, 2, \ldots, N\}$ многопериодных инвестиционных проектов с инвестиционным горизонтом T и периодами реализации $t = 0, 1, 2, \ldots, T$. Сформулируем задачу выбора оптимального проекта из множества I в условиях стохастической неопределенности относительно процентных ставок. При этом рассмотрим случай несовершенного рынка капитала.

Следуя работе [2], введем обозначения:

- M_t — базовые платежи, $t = 0, 1, 2, \ldots, T$;
- $f = (f_0, f_1, \ldots, f_T)$ — вектор, определяющий структуру изъятий (структуру дохода);
- $h_t \in \left[\underline{h_t}, \overline{h_t}\right], s_t \in [\underline{s_t}, \overline{s_t}]$ — соответственно диапазоны ставок по дополняющим инвестициям и заимствованиям $t = 1, 2, \ldots, T$;
- $z_t(i)$ — инвестиционные платежи, $i \in I$, $t = 0, 1, \ldots, T$;
- Y^* — заданный инвестором одинаковый для всех проектов уровень изъятий (дохода) при оценке критерия максимума остаточной стоимости;
- C^* — заданный инвестором одинаковый для всех проектов уровень остаточной стоимости при оценке критерия максимума уровня дохода.

Процентные ставки являются случайными с некоторым распределением на прогнозных диапазонах.

Через $C_T(i)$ и $Y(i)$ обозначим остаточную стоимость и уровень изъятий, достигаемые при перечисленных исходных данных при реализации проекта $i \in I$.

Пусть определена функция полезности $u\left(C_T(i), Y(i)\right)$, отражающая предпочтения инвестора на неопределенных значениях остаточной стоимости и уровня изъятий. Тогда, согласно концепции ожидаемой полезности, задача инвестора заключается в выборе инвестиционного проекта, который максимизирует ожидаемую полезность (математическое ожидание функции полезности) [2]:

$$M\{u(C_T(i), Y(i))\} \to \max_{i \in I}, \tag{1}$$

где M — математическое ожидание функции полезности на неопределенных двумерных последствиях$(C_T(i), Y(i)) \in R^2_+$:

$$M\{u\} = \iint\limits_A u\left(C_T(i), Y(i)\right) P\left(C_T(i), Y(i)\right) dC_T(i) dY(i), \tag{2}$$

где $A \subseteq R^2_+$ — область возможных значений реализаций двумерных последствий реализации проектов; $P\left(C_T(i), Y(i)\right)$ — неизвестное распределение учитываемых двумерных последствий.

Решением задачи (1) будет инвестиционный проект, ожидаемая полезность которого является наибольшей:

$$i^* = \arg\max_{i \in I} M\{u(C_T(i), Y(i))\}. \tag{3}$$

2. Конкретизация компонент задачи.

Для решения задачи (3) требуется конкретизировать функцию полезности, способы оценки остаточной стоимости, уровня изъятий и ожидаемой полезности последствий реализации проектов.

Спецификация функции полезности. Согласно работе [1] критерии остаточной стоимости и дохода взаимно независимы по предпочтению, что обуславливает существование аддитивной функции полезности, отражающей предпочтения инвестора в пространстве R^2_+:

$$u(C, Y) = \alpha_1 u_1(C) + \alpha_2 u_2(Y), \tag{4}$$

где $u_1(C)$ — условная однокритериальная функция полезности, отражающая (условные) предпочтения инвестора по критерию капитализации; $u_2(Y)$ — условная однокритериальная функция полезности, отражающая (условные) предпочтения инвестора по критерию дохода; α_1, α_2 — весовые коэффициенты (важности) критериев капитализации и дохода соответственно, $\alpha_1 \geqslant 0, \alpha_2 \geqslant 0, \alpha_1 + \alpha_2 = 1$.

Оценка параметров функции полезности. Пусть двумерные последствия $\left(C^I, Y^I\right) \neq \left(C^{II}, Y^{II}\right)$ равноценны для инвестора, то есть $\left(C^I, Y^I\right) \sim \left(C^{II}, Y^{II}\right)$. Тогда согласно определению функции полезности, которая отражает предпочтения инвестора, имеем $u\left(C^I, Y^I\right) = u(C^{II}, Y^{II})$. Теперь с учетом вида (4) функции полезности и нормализующего условия на ее параметры получаем систему уравнений относительно α_1, α_2:

$$\begin{cases} \alpha_1 u_1\left(C^I\right) + \alpha_2 u_2\left(Y^I\right) = \alpha_1 u_1\left(C^{II}\right) + \alpha_2 u_2\left(Y^{II}\right), \\ \alpha_1 + \alpha_2 = 1. \end{cases} \tag{5}$$

Из (5) находим

$$\alpha_1 = \frac{u_2\left(Y^{II}\right) - u_2\left(Y^I\right)}{u_1\left(C^I\right) - u_1\left(C^{II}\right) - u_2\left(Y^I\right) + u_2\left(Y^{II}\right)}, \tag{6}$$

$$\alpha_2 = 1 - \alpha_1 = \frac{u_1\left(C^I\right) - u_1\left(C^{II}\right)}{u_1\left(C^I\right) - u_1\left(C^{II}\right) - u_2\left(Y^I\right) + u_2\left(Y^{II}\right)}. \tag{7}$$

Выявление пары равноценных для инвестора двумерных последствий сформулируем как задачу компенсации [2] ухудшения по критерию остаточной стоимости заданного двумерного последствия его улучшением по критерию дохода. Результатом применения алгоритма решения задачи компенсации является выявление пары равноценных для инвестора двумерных последствий, по которым осуществляется оценка параметров функции полезности.

Спецификация распределений процентных ставок. Следуя работе [3], в силу отсутствия информации о распределении вероятностей процентных ставок по дополняющим инвестициям и заимствованиям используем распределения максимальной энтропией на заданных прогнозных интервалах возможных значений ставок в плановых периодах реализации проектов. Известно (см. [3]), что

таким распределением является равномерное распределение вероятностей.

Оценка остаточной стоимости. Обозначим через $C_t(i)$ значение остаточной стоимости, достигаемое проектом $i \in I$ к моменту времени $t = 0, 1, \ldots, T$. Алгоритм оценки остаточной стоимости $C_T(i)$ на момент завершения реализации проекта определяется следующими соотношениями (см. [2]):

при $t = 0$

$$C_t(i) = M_t - f_t Y^* + z_t(i);$$

при $t \geqslant 1$

$$C_t(i) = \begin{cases} M_t - f_t Y^* + z_t(i) + (1 + h_t)\, C_{t-1}(i) & \text{при } C_{t-1}(i) > 0, \\ M_t - f_t Y^* + z_t(i) + (1 + s_t)\, C_{t-1}(i) & \text{при } C_{t-1}(i) < 0. \end{cases} \quad (8)$$

В первом из соотношений (8) осуществляется дополняющее инвестирование по ставке h_t, во втором — дополняющее заимствование по ставке s_t. Величина $f_t Y^*$ в (8) определяет изъятия (доход) на текущее потребление в момент $t = 0, 1, 2, \ldots, T$.

Оценка достигаемого уровня изъятий. Обозначим через $Y(i)$ достигаемый уровень изъятий (дохода) при требуемом уровне C^* остаточной стоимости проекта $i \in I$. Обозначим также через $\varepsilon > 0$ требуемую точность достижения уровня C^* остаточной стоимости проекта.

Приведем возможный алгоритм оценки достигаемого уровня изъятий.

1) Находим величину Y_k, которая приведет к остаточному имуществу $C_{T,k}(i) > C^*$, вычисляемому с использованием соотношений (8), в которых Y^* заменяется на Y_k.

2) Находим величину Y_{k+1}, которая приведет к остаточному имуществу $C_{T,k+1}(i) < C^*$, вычисляемому с использованием соотношений (8), в которых Y^* заменяется на Y_{k+1}.

3) Находим величину Y_{k+2} с помощью линейной интерполяции:

$$Y_{k+2} = Y_k + (C^* - C_{T,k}(i)) \cdot \frac{Y_{k+1} - Y_k}{C_{T,k+1}(i) - C_{T,k}}.$$

4) Вычисляем с использованием соотношений (8), в которых Y^* заменяется на Y_{k+2}, остаточное имущество $C_{T,k+2}(i)$.

5) Если $|C_{T,k+2}(i) - C^*| \leqslant \varepsilon$, то перейти на шаг 6, иначе положить $Y_k = Y_{k+2}$, $C_{T,k}(i) = C_{T,k+2}(i)$ и перейти на шаг 3.

6) Положить $Y(i) = Y_{k+2}$ и завершить работу алгоритма.

Оценка ожидаемой полезности проекта. В силу неизвестности в соотношении (2) распределения двумерных последствий реализации проектов, оценка ожидаемой полезности проводится путем формирования выборки реализаций остаточной стоимости и уровня дохода и статистической оценки математического ожидания функции полезности. Элементы указанной выборки определяются на основе генерирования реализаций процентных ставок по равномерному распределению на заданных диапазонах, которые связаны с плановыми периодами $t = 1, 2, \ldots, T$. По каждой l-й реализации процентных ставок с использованием соотношений (8) и алгоритма оценки уровня дохода вычисляются для каждого проекта реализации остаточной стоимости и уровня дохода. В итоге получаем выборку реализаций $C_T^l(i)$и $Y^l(i)$, $i \in I, l = 1, 2, \ldots, L$, где L — объем выборки.

Ожидаемая полезность для каждого проекта $i \in I$ рассчитывается следующим образом:

$$M\{u\,(C_T(i), Y(i))\} = \frac{1}{L}\sum_{l=1}^{L} u\left(\left(C_T^l(i), Y^l(i)\right)\right). \tag{9}$$

Далее, согласно (3), определяется предпочтительный для инвестора инвестиционный проект.

3. Иллюстративный пример

Рассмотрим множество многопериодных инвестиционных проектов $I = \{0, 1, 2, 3\}$. Плановый горизонт $T = 5$.

Желаемые инвестором уровни дохода и капитализации при формулировке целей максимизации уровня капитализации и максимизации уровня дохода составляют $Y_p = 60$, $C_p = 260$ соответственно. Базовые платежи и платежи, связанные с осуществлением проектов, а также структура изъятий представлены в табл. 1. Заданы

t — момент времени	0	1	2	3	4
M_t — Базовые платежи	500	−200	20	150	300
f_{2t} — Структура изъятий	1.0	1.1	1.2	1.4	1.5
$z_t(0)$ — Инвестиционные платежи по проекту A(0)	−800	600	200	150	−80
$z_t(1)$ — Инвестиционные платежи по проекту B(1)	−700	300	400	30	100
$z_t(2)$ — Инвестиционные платежи по проекту C(2)	−400	−200	700	0	0
$z_t(3)$ — Инвестиционные платежи по проекту D(3)	0	0	0	0	0

Таблица 1.

следующие диапазоны ставок по дополняющим инвестициям и заимствованиям соответственно: $h_t \in [5, 9]$, $s_t \in [9, 12]$.

Ниже представлены результаты, полученные после применения к исходным данным алгоритмов из раздела 2 для вычисления (9) и решения задачи выбора многопериодных инвестиционных проектов.

Имеем

$$M\{u(C_5(0), Y(0))\} = 0.0,$$
$$M\{u(C_5(1), Y(1))\} = 0.896,$$
$$M\{u(C_5(2), Y(2))\} = 0.879,$$
$$M\{u(C_5(3), Y(3))\} = 0.862.$$

Следовательно, решением является проект 1. Именно данный проект согласуется с предпочтениями инвестора, которые отражают совокупный учет критериев капитализации и дохода в условиях стохастической неопределенности относительно процентных ставок.

Заключение

Рассмотренная модель выбора многопериодных инвестиционных проектов, позволяет повысить обоснованность выбора за счет согласованного с предпочтениями инвестора совокупного учета основных, в общем случае противоречивых показателей эффективности проектов, а также за счет использования для описания неопределенности

интервальных прогнозных оценок процентных ставок по инвестированию и заимствованию. Именно указанные положения определяют элементы новизны предложенной в работе модели.

Список литературы

[1] Кини, Р. Л. Принятие решений при многих критериях: предпочтения и замещения / Р. Л. Кини, Х. Райфа. — М. : Радио и связь, 1981. — 560 с.

[2] Михно, В. Н. Модель формирования портфеля многопериодных инвестиций / В. Н. Михно, А. С. Канарейкина // Вестник ТвГУ. Серия: Прикладная математика. — 2017. — № 2. — С. 79–88.

[3] Чисар, И. Теория информации / И. Чисар, Я. Кернер. — М. : Мир, 1985. — 400 с.

Библиографическая ссылка

Сергеева, Е. А. Многоцелевой выбор инвестиционных проектов в условиях стохастической неопределенности // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 167–174.

Сведения об авторах

Сергеева Елена Александровна
Студентка магистратуры
Направление «Прикладная информатика»

УДК 336.7631

Портфельное инвестирование на основе рыночной модели

Столярова А. А.

Тверской государственный университет

Аннотация. Статья посвящена портфельному инвестированию на основе рыночной модели. В ней рассматривается алгоритм построения портфеля на основе рыночной модели. Апробация работы осуществлялась на российских акциях из списка «Голубых фишек». На разных периодах времени на их основе были построены инвестиционные портфели.

Введение

С развитием фондового рынка и повышением финансовой грамотности инвесторам предоставляются широкие возможности для осуществления инвестиции свободных средств. Большая часть инвесторов несклонна к риску, вследствие чего их привлекает диверсификация средств с помощью портфельного инвестирования. Такой способ инвестирования позволяет уменьшить собственную часть риска, присущую активам, но для этого необходимо прибегать к анализу рынка.

В настоящее время большинство инвесторов используют модели формирования портфеля, которые позволяют достигать целей инвестирования, за счет баланса требований к доходности и риску, выраженных в виде некоторого соотношения, которое необходимо оптимизировать. Одной из наиболее известных моделей такого типа является портфель Марковица [1, 2]. Основной недостаток такого подхода — необходимость решать нелинейную задачу оптимизации. При сложной целевой функции, большом числе активов или дополнительных ограничениях на веса бумаг поиск глобального экстремума может оказаться нетривиальным и затратным по времени. Поэтому востребованы не оптимизационные инвестиционные модели, когда

активы отбираются на основе пороговых значений определенных показателей риска. К таким моделям относится метод формирования портфеля, опирающийся на рыночную модель [3].

1. Необходимые понятия

1.1. Портфельное инвестирование

Портфель в финансах — совокупность инвестиционных вложений юридического или физического лица.

Инвестиционный портфель — набор реальных или финансовых инвестиций. В узком смысле это совокупность ценных бумаг разного вида, разного срока действия и разной степени ликвидности, принадлежащая одному инвестору и управляемая как единое целое [1].

Доходность инвестиций — это прибыль от инвестиций за определенный период времени:

$$R_t = \frac{P_t - P_{t-1}}{P_t}, \tag{1}$$

где P_t есть текущая цена актива.

Основными характеристиками инвестиции являются средняя доходность и инвестиционный риск.

Средняя доходность численно характеризует ожидаемый инвестором результат, а инвестиционный риск связан с возможностью того, что реальная доходность будет меньше ожидаемой.

Ожидаемая доходность и риск отдельных акций с математической тоски зрения представляют собой математическое ожидание $\mu = M(R_t)$ и среднеквадратическое отклонение $\sigma = \sqrt{D\left(R_t\right)}$ случайной величины R_t. Большая дисперсия или изменчивость ожидаемой доходности ценной бумаги означала, что ценная бумага была более рискованной, чем та, которая имела меньшую дисперсию.

Соответствующие характеристики для портфеля активов $\pi = (x_1, \ldots, x_n)$ вычисляются по формулам

$$\mu_\pi = x'\mu, \qquad \sigma_\pi = \sqrt{x'Sx}, \tag{2}$$

где x_i — доля вложений в i-й актив, $\sum x_i = 1$, $\mu = (\mu_1, \ldots, \mu_n)$ — вектор средних доходностей, $S = (\sigma_{ij})$ — матрица ковариаций доходностей. Для оценки этих величин на основе наблюдений используются их выборочные средние, дисперсии и ковариации:

$$\widehat{\mu} = \frac{1}{T}\sum_{t=1}^{T} R_t, \qquad \widehat{\sigma}_{ij} = \frac{1}{T}\sum_{t=1}^{T} (R_{it} - \widehat{\mu}_i)\,(R_{jt} - \widehat{\mu}_j)\,. \tag{3}$$

Объединение в портфель различных инвестиционных инструментов называется диверсификацией вложений. Диверсификация позволяет снижать инвестиционный риск.

1.2. Модель рыночного индекса

Рыночный портфель — это совокупность всех акций, обращающихся на фондовом рынке. Для аппроксимации рыночного портфеля финансовых моделях используют рыночный индекс, представляющий собой портфель из наиболее ликвидных биржевых бумаг, взвешенных по капитализации фирм [2].

Рыночной моделью называют зависимость доходности конкретной ценной бумаги от доходности рыночного индекса, которая формализованном виде представляет собой парную линейную регрессию:

$$R_i = \alpha_{iI} + \beta_{im} R_m + \varepsilon_i, \tag{4}$$

где R_i — доходность ценной бумаги, R_m — доходность рыночного индекса, α_{im} — коэффициент смещения, β_{im} — коэффициент наклона, ε_i — случайная погрешность.

Коэффициент смещения α_{im} указывается в процентах доходности и показывает, насколько от ожидаемых значений смещена текущая доходность ценной бумаги относительно доходности индекса.

Бета-коэффициент β_{im} показывает, насколько текущее изменение доходности бумаги соответствует текущему изменению доходности индекса, то есть отражает синхронность ценовой динамики бумаги и индекса.

Случайная погрешность ε_i характеризует расхождение фактической доходности ценной бумаги с ее ожидаемыми расчетными значениями. Случайная погрешность измеряется, как и доходность, в процентах.

Введем также следующие обозначения: σ_i^2 — дисперсия доходности ценной бумаги, $\sigma_{\varepsilon i}^2$ — дисперсия движения акций, связанная с движением рыночного индекса, σ_m^2 — дисперсия рыночного индекса.

Согласно рыночной модели риск ценной бумаги состоит из двух частей:

$$\sigma_i^2 = \beta_{im}^2 \sigma_m^2 + \sigma_{\varepsilon i}^2, \tag{5}$$

- систематического (недиверсифицируемого) — $\beta_{im}^2 \sigma_m^2$. Систематический риск, известный как рыночный риск, обусловлен общими изменениями на рынке;
- собственного (несистематического) риска $\sigma_{\varepsilon i}^2$. Собственный риск связан с особенностями отрасли, к которой принадлежит актив и особенностями компании, выпустившей эту ценную бумагу.

Диверсификация позволяет уменьшить собственные риски активов.

1.3. Алгоритм построения портфеля на основе рыночной модели

Данный алгоритм основывается на работе [3].

1) Находим прибыль акции и прибыль рынка.

   Прибыль акции: $R_i = \dfrac{P_{it} - P_{it-1}}{P_{it}}$ и $\overline{R}_i = \dfrac{1}{T} \sum\limits_{t=1}^{T} R_{it}$.

   Прибыль рынка: $R_m = \dfrac{P_{mt} - P_{mt-1}}{P_{mt}}$ и $\overline{R}_m = \dfrac{1}{T} \sum\limits_{t=1}^{T} R_{mt}$.

2) Находим альфа-, бета-стоимость каждой акции:

$$\beta_i = \frac{\sum\limits_{t=1}^{T} \left[\left(R_i - \overline{R}_i \right) \left(R_m - \overline{R}_m \right) \right]}{\sum\limits_{t=1}^{T} \left(R_m - \overline{R}_m \right)^2}, \tag{6}$$

$$\alpha_i = \overline{R}_i - \beta_i \overline{R}_m.$$

3) Считаем необходимые риски.

   Рыночный риск: $\sigma_m^2 = \dfrac{1}{T-1}\sum\limits_{t=1}^{T}\left(R_{mt} - \overline{R}_m\right)^2$.

   Систематический риск акций: $\sigma_i^2 = \dfrac{1}{T-1}\sum\limits_{t=1}^{T}\left(R_{it} - \overline{R}_i\right)^2$.

   Собственный риск: $\sigma_{\varepsilon i}^2 = \sigma_i^2 - \beta_i^2\sigma_m^2$.

4) Находим избыточную доходность (ERB):

$$ERB = \frac{\overline{R}_i - R_f}{\beta_i}, \tag{7}$$

   где R_f есть безрисковая ставка процента.

5) Производим расчет коэффициента C_i:

$$C_i = \frac{\sigma_m^2 \sum \dfrac{(\overline{R}_i - R_f)\beta_i}{\sigma_{\varepsilon i}^2}}{1 + \sigma_m^2 \sum \dfrac{\beta_i^2}{\sigma_{\varepsilon i}^2}}. \tag{8}$$

6) Принцип отбора акций в портфель осуществляется в соответствии со значением ERB по правилу

$$\begin{cases} ERB > C_i^*, & \text{акция } i \text{ включена в портфель,} \\ ERB \leqslant C_i^*, & \text{акция } i \text{ не включена в портфель.} \end{cases} \tag{9}$$

7) Доля капитала X_i для каждой выбранной акции в портфеле:

$$X_i = \frac{Z_i}{\sum\limits_{t=1}^{n} Z_i}, \qquad Z_i = \frac{\beta_i}{\sigma_{\varepsilon i}^2}\left(\frac{\overline{R}_i - R_f}{\beta_i} - C_i^*\right). \tag{10}$$

8) Для оценки соотношения доходность-риск рассчитываем коэффициент вариации (относительное стандартное отклонение):

$$CV = \frac{\sigma_p}{R_p}. \tag{11}$$

2. Результаты

Данный алгоритм будет рассмотрен применительно к российским «голубым фишкам». Голубые фишки — акции наиболее крупных, ликвидных и надежных компаний со стабильными показателями доходности, а также сами эти компании. Московская биржа рассчитывает индекс голубых фишек на основании цен сделок с акциями 14 наиболее ликвидных эмитентов российского фондового рынка:

1) Сбербанк
2) Лукойл
3) Газпром
4) Алроса
5) X5 Retail Group
6) Магнит
7) МТС
8) НОВАТЭК
9) Роснефть
10) ВТБ
11) Яндекс
12) Северсталь
13) Норникель
14) Сургутнефтегаз

Рассматриваются еженедельные данные. Результаты инвестирования будут анализировать на двух периодах: до-пандемийном, пандемийном. Для исследования взята годовая безрисковая процентная ставка 7%.

Рассмотрим период времени: 01.01.2019–01.02.2020.

На данном периоде оптимальный портфель состоит из 9 активов: «Сбербанк», «Лукойл», «Газпром», «X5 Retail Group», «МТС», «ВТБ», «Яндекс», «Норникель», «Сургутнефтегаз». Наибольшая доля вложения — в акции «Норникель», 0,21. Заметим, что на этом периоде в портфель включено большое число активов ресурсодобывающего сектора. Банковская сфера в основном представлена акциями Сбербанка, связь и IT-сектор акциями МТС и Яндекс.

Средняя доходность и риск такого портфеля равны 0,0066 и 0,00022 соответственно, что соответствует данному стабильному периоду в РФ.

Следующим периодом для изучения работы данного алгоритма взят период пандемии (COVID-19) (02.02.2020–31.12.2021):

Из полученных результатов видно, что в состав оптимального портфеля вошло 8 акций, большая часть которых, как и ранее

№	Название акции	Значение ERB	C_i	Отбор	Доля акции
1.	Сбербанк	0,004	0,0014	$>^*$	0,17
2.	Лукойл	0,005	0,0019	$>^*$	0,14
3.	Газпром	0,003	0,0023	$>^*$	0,035
4.	Алроса	−0,006	0,0018	$<$	0
5.	X5 Retail Group	0,008	0,0019	$>^*$	0,044
6.	Магнит	−0,00032	0,0018	$<$	0
7.	МТС	0,005	0,002	$>^*$	0,15
8.	НОВАТЭК	−0,0013	0,0021	$<$	0
9.	Роснефть	0,0018	0,002	$<$	0
10.	ВТБ	0,0048	0,0022	$>^*$	0,098
11.	Яндекс	0,01	0,0024	$>^*$	0,103
12.	Северсталь	−0,002	0,0021	$<$	0
13.	Норникель	0,009	0,0024	$>^*$	0,21
14.	Сургутнефтегаз	0,005	0,0026	$>^*$	0,05

Таблица 1.

	α	β	Ожидаемая доходность	Риск	Коэффициент вариации
1.	0,0023	0,945	0,0066	0,00022	0,03

Таблица 2.

относится к добывающим компаниям. В составе портфеля также остался представитель IT-сектора, с незначительным снижением веса. Наибольшей долей вложения 0,247 характеризуются акции «Газпром», доля которого увеличилась в портфеле в 7 раз, а акции «Норникеля» заменила «Северсталь» практически с идентичным весом. Перераспределение весов произошло за счет существенного снижения долей банковского сектора и связи. Из табл. 4 следует, что по сравнению с предыдущим периодом ожидаемая доходность портфеля понизилась до 0,0052, а риск наоборот вырос до 0,00085. Это объективная тенденция, связанная с нарастанием международной напряженности.

№	Название акции	Значение	C_i ERB	Отбор	Доля акции
1.	Сбербанк	0,000942	0,0005	$>^*$	0,035
2.	Лукойл	0,000038	0,0003	$<$	0
3.	Газпром	0,003307	0,0013	$>^*$	0,247
4.	Алроса	0,003451	0,00152	$>^*$	0,0983
5.	X5 Retail Group	−0,002569	0,0013	$<$	0
6.	Магнит	0,004582	0,0015	$>^*$	0,105
7.	МТС	−0,003079	0,00097	$<$	0
8.	НОВАТЭК	0,003389	0,0014	$>^*$	0,17
9.	Роснефть	0,001970	0,00149	$>^*$	0,042
10	ВТБ	−0,000150	0,0013	$<$	0
11.	Яндекс	0,005984	0,0014	$>^*$	0,0997
12.	Северсталь	0,010150	0,0015	$>^*$	0,203
13.	Норникель	0,000667	0,0015	$<$	0
14.	Сургутнефтегаз	−0,000609	0,0015	$<$	0

Таблица 3.

	α	β	Ожидаемая доходность	Риск	Коэффициент вариации
1.	0,03	0,942	0,0052	0,00085	0,16

Таблица 4.

Заключение

В данной статье рассмотрены теоретические основы портфельного инвестирования, даны определения основным понятиям. Изучена модель рыночного индекса, рыночного портфеля. При помощи алгоритма построения портфеля на основе рыночной модели были построены портфели с определенными акциями в разные периоды времени.

Портфельное инвестирование на основе рыночной модели позволяет лучше проанализировать рынок в разные периоды времени. Благодаря полученным результатам, в дальнейшем можно спрогнозировать следующие данные по акциям, а также узнать будущий

состав инвестиционного портфеля.

Список литературы

[1] Новиков, А. И. Теория принятий решений и управление рисками в финансовой и налоговой сферах : учебное пособие / А. И. Новиков, Т. И. Солодкая. — 2-е изд., стер. — М. : Издательско-торговая корпорация «Дашков и К°», 2019. — 284 с.

[2] Fisher, D. E. Security Analysis and Portfolio Management / D. E. Fisher, R. J. Jordan. — 5th ed. — New Jersey: Prentice Hall, 1994. — 352 p.

[3] Kamil, A. Portfolio Analysis Using Single Index Model // WSEAS Transactions on Mathematics. — 2003. — Vol. 2. — URL: http://www.academia.edu/1771211/Portfolio_Analysis_Using_Single_Index_Model

Библиографическая ссылка

Столярова, А. А. Портфельное инвестирование на основе рыночной модели // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 175–183.

Сведения об авторах

Столярова Алена Андреевна
Студентка магистратуры
Направление «Прикладная математика и информатика»

УДК 004.9

Оценка кредитоспособности заемщиков

Терентьев Д. А.

Тверской государственный университет

Аннотация. В статье рассматриваются методы решения задачи кредитного скоринга, базирующиеся на построении деревьев решений по алгоритмам CHAID и CART, а также методе k-ближайших соседей.
Результаты работы указывают на необходимость использования ансамблей деревьев при принятии решения, объединения нескольких методов для получения более точных и стабильных результатов. Также проанализировано влияние обучающей выборки на процесс обучения алгоритмов.

Введение

Предоставление кредитов — трудоемкий процесс. Каждая кредитная организация стремится уменьшить риски невозврата кредита, при этом не увеличивая количество субъективных отказов. Конкуренция на рынке способствует тому, что методы кредитного скоринга должны обеспечивать точное объективное решение за достаточно быстрое время. Без использования математических моделей здесь не обойтись. Процесс принятия решения в таких моделях базируется на статистике, что позволяет повысить точность оценки, а также исключить субъективность при анализе клиентов. Использование вычислительных мощностей компьютеров, позволяет получить максимальную скорость принятия решений.

1. Кредитный скоринг на основе алгоритмов классификации

Задача кредитного скоринга заключается в определении кредитоспособности заемщика, на основе его характеристик и характеристик других заемщиков из обучающей выборки. В отношении каждого заемщика нужно определить: давать кредит или нет. Таким образом, мы имеем дело с задачей бинарной классификации.

Известно множество подходов к решению такой проблемы, в частности метод k-ближайших соседей. Среди достаточно интересных и несложных методов можно также выделить деревья решений.

В данной статье будет проведен сравнительный анализ эффективности работы следующих алгоритмов:

- CHAID (Chi-squared Automatic Interaction Detection) — автоматическое обнаружение взаимодействий с помощью критерия хи-квадрат;
- CART (Classification and Regression trees) — дерево классификации и регрессии;
- KNN (k-nearest neighbors) — k-ближайших соседей.

1.1. Деревья решений

Дерево решений — эффективный инструмент интеллектуального анализа данных и предсказательной аналитики. Он помогает в решении задач по классификации и регрессии.

Дерево решений представляет собой иерархическую древовидную структуру, состоящую из правила вида «Если ..., то ...». За счет обучающего множества правила генерируются автоматически в процессе обучения. Основной сферой применения деревьев решений является поддержка процессов принятия управленческих решений, используемая в статистике, анализе данных и машинном обучении [1].

Описание объектов — набор правил в дереве решений, который позволяет компактно описывать объекты. Поэтому вместо сложных структур, описывающих объекты, можно хранить деревья решений.

Ансамбль деревьев или случайный лес — это алгоритм коллегиального принятия решения комитетом или ансамблем решающих деревьев. Он применяется для увеличения точности и устойчивости алгоритмов, а также решения проблемы переобучения. Для построения леса обучающая выборка разбивается на несколько наборов данных, на основе каждого из которых строится свое дерево решений, позволяющее классифицировать объект. В качестве решения возвращается класс с наибольшим количеством голосов.

Некоррелированность деревьев снижает вероятность индивидуальных ошибок и повышает точность прогноза.

Для построения отдельных деревьев используются различные алгоритмы, различающиеся видом информационной функции, выявляющей наиболее важные характеристики объекта, в которых происходит дальнейшее ветвление дерева. Мы остановимся на двух методах: CHAID и CART.

Алгоритм CHAID.

1) Находим характеристику, которая даст наибольшее количество информации. Для этого используем критерий согласия Пирсона (критерий согласия χ^2):

$$X^2 = n \sum_{i=1}^{k} \frac{(n_i/n - P_i(\theta))^2}{P_i(\theta)}. \tag{1}$$

2) Разбиваем текущий узел дерева, на новые узлы, количество которых совпадает с количеством возможных значений в выбранной характеристике.

3) Для каждого нового узла выполняем шаги 1 и 2 до тех пор, пока во всей оставшейся выборке целевая функция не будет иметь одинаковое значение.

Алгоритм CART.

1) Для нахождения характеристики, по которой будет строиться расщепление, выбираем подгруппу из возможных значений каждой характеристики:

$$Gini = \sum_{i=1}^{C} p_i \left(1 - p_i\right), \tag{2}$$

$$Gini_{split}(T) = \frac{N_1}{N} Gini\left(T_1\right) + \frac{N_2}{N} Gini\left(T_2\right). \tag{3}$$

2) Разбиваем текущий узел дерева на два новых узла: значение находится в выбранной на предыдущем шаге подгруппе или нет.

3) Для каждого нового узла выполняем шаги 1 и 2 до тех пор, пока во всей оставшейся выборке целевая функция не будет иметь одинаковое значение.

1.2. Метод ближайших соседей

Метод ближайших соседей — непараметрический алгоритм машинного обучения, основанный на поиске ближайших объектов с похожими свойствами. Для классификации используют следующее правило: относить объект к тому классу, к которому относится большинство его ближайших соседей. В качестве меры близости обычно используется Евклидово расстояние [2].

Алгоритм KNN.

1) Вычислить расстояние до каждого из объектов обучающей выборки.
2) Отобрать k объектов обучающей выборки, расстояние до которых минимально.
3) Класс классифицируемого объекта — это класс, наиболее часто встречающийся среди k-ближайших соседей.

2. Сравнительный анализ результатов классификации

2.1. Предварительная обработка данных

Для оценки эффективности алгоритмов будет использоваться выборка данных с информацией о 1000 клиентов, для каждого из которых определены 20 характеристик качественного или числового типа. Примеры характеристик представлены в табл. 1.

Методы CHAID и CART работают с качественными характеристиками, которые разбиты на группы. Чтобы не исключать числовые данные, требуется их предварительная обработка: числовые характеристики разбиваются на группы.

В методе ближайших соседей считается расстояние. Поэтому в отличии от деревьев решений, здесь нужны числовые значения характеристик. Для этого каждой группе во всех характеристиках присваивается численное значение.

2.2. Результаты классификации

В таблицах ниже приведены данные в процентах о числе правильно классифицированных заемщиков. Для разного числа деревьев и

Название	Возможные значения	Тип
Кредитная история	- Все кредиты возвращены - Оплата существующих кредитов без задержек - Существуют задержки с выплатой в прошлом	Качественный
Жилье	- Аренда - Собственность - Отсутствует	Качественный
Количество кредитов в этом банке		Числовой
Сумма кредита		Числовой

Таблица 1. Характеристики заемщика.

длины выборки алгоритм запускался 10 раз. Алгоритм определяет класс каждого из 100 клиентов: хороший клиент или плохой. На выходе получается количество верно определенных классов клиентов. В табл. 2 и 3 приведены средние значения по результатам 10 запусков.

Рассмотрим результаты работы алгоритмов CHAID и CART при построении лишь одного дерева решений. Видно, что при использовании обучающей выборки, содержащей информацию о 100 клиентах, получается весьма неудовлетворительная точность в 62.2% и 63.8% соответственно. При увеличении объема выборки точность начинает расти, превышая отметку в 70% для обоих методов. Это показывает важность размера обучающей выборки.

Далее рассмотрим деревья решений среднего размера. Пусть это будут обучающие выборки длиной в 100 клиентов. Нас интересует зависимость точности от количества деревьев, то есть оптимальный размер ансамбля деревьев. Видно, что с ростом числа деревьев в ансамбле точность повышается. Исключением является алгоритм CHAID, где при увеличении числа деревьев с 25 до 50 точность осталась прежней. На самом деле, можно заметить, что и в алгоритме CART при переходе от 25 деревьев к 50, точность увеличилась

		Число деревьев				
		1	3	7	25	50
Длина выборки	100	62.2	65	65	72.2	72.5
	200	72.6	68.1	70.2	75.5	73.4
	300	65.1	69.3	72.8	76.1	75.8
	400	64	69.1	72.8	78	78
	500	68.7	69.9	75.6	77.4	78.4
	600	66.8	75	75.1	79.3	79.3
	700	74.8	75.4	78.8	80	79.4
	800	70.6	73.5	76.3	79.8	80.2

Таблица 2. Метод CHAID.

		Число деревьев				
		1	3	7	25	50
Длина выборки	100	63.8	67.2	67.2	73.9	74.3
	200	74.3	69.5	71.2	74.9	72.6
	300	65.5	70.7	73.6	73.7	73.4
	400	65.3	69.5	72	73.8	74.6
	500	65	69	74.1	75.2	75
	600	67.1	72	74.7	76.7	76.4
	700	72.2	75.8	78.2	77.1	71.1
	800	71.1	72.6	76.9	77.9	77.8

Таблица 3. Метод CART.

весьма несущественно, хотя число деревьев выросло в два раза. Причина этого кроется в том, что с определенного момента строятся однотипные деревья. Это происходит из-за небольшого количества записей в обучающей выборке, относительно размера выборки, использующейся для построения одного дерева решений в ансамбле.

Таким образом, в плане точности ансамбли предпочтительней отдельных деревьев. Однако при их использовании нужно помнить о том, что избыточное количество деревьев, не только не улучшает результат, но и снижает производительность.

Для метода ближайших соседей (табл. 4) использовалась вся обучающая выборка, в отличии от CHAID и CART. По результатам

Количество соседей	Точность
1	77
2	74
3	77
4	71
5	70
6	74
7	69
8	73
9	70
10	75

Таблица 4. Метод KNN.

видно, что лучшая точность была достигнута при $k = 1$ и $k = 3$. Далее, при объединении методов использовалось $k = 3$. Причина в том, что при большем числе соседей уменьшается случайность метода.

Далее все три метода были объединены в одну программу. Для каждого клиента любой из выше представленных алгоритмов определяет его класс. Получается, что при использовании трех методов, мы имеем три предположения о принадлежности клиента к тому или иному классу. После, по большему числу голосов определяется финальный класс клиента. Так как количество использованных методов нечетное, спорных ситуаций не возникает.

Из табл. 5 видно, что минимальная точность получается при использовании 1 дерева длиной 100, а увеличение точности зависит от числа деревьев и длины выборки. Все так же, как в алгоритмах CHAID и CART, но точность выше. Если рассматривать только максимальную точность: CHAID — 80.2%, CART — 78.2%, KNN — 77%, объединение методов — 82.2%, то разница не сильно заметна. Но для худшей точности разница очевидна: ниже 70%, а в случае алгоритмов CHAID и CART ниже 65%. Объединение же алгоритмов обеспечивает худшую точность в 73.4%, что немногим хуже лучшей точности в методе ближайших соседей.

Также видно, что при количестве деревьев равном 7 и длине выборки равной 500, мы имеем удовлетворительную точность в

		Число деревьев				
		1	3	7	25	50
Длина выборки	100	73.4	75	75	76.5	76.5
	200	76.5	76.8	77.3	78.6	76.9
	300	75.3	77.5	78.7	78.5	78.4
	400	74.9	76.9	78.1	79.2	79
	500	77.1	76.7	81.2	81.1	80.5
	600	76	79.8	80.4	81.6	81.1
	700	77.4	79.9	80.4	81.5	82
	800	79.1	79.3	80.8	82.3	82.6

Таблица 5. Объединение всех методов.

81.2%. При увеличении любого из двух параметров, точность не опускается ниже отметки в 80% процентов. На основе этого, можно сделать вывод о том, что объединение методов — важный шаг к увеличению точности и стабильности результатов.

Заключение

В данной работе на примере данных о заемщиках была проиллюстрирована работа алгоритмов классификации, основанных на деревьях решений и методе ближайших соседей. Апробация показала, что для повышения точности и устойчивости результатов необходимо применять ансамбли решающих деревьев. Помимо этого автором было реализовано объединение двух классических подходов, позволившее увеличить точность на самых неудачных обучающих выборках. Соответствующее уменьшение разброса между наилучшими и наихудшими результатами способствует повышению стабильности результатов классификации.

Список литературы

[1] Шахиди, А. Деревья решений: общие принципы : [сайт]. — 2019. — 4 дек. — URL: https://loginom.ru/blog/decision-tree-p1 (дата обращения: 12.05.2022)

[2] Chakure, A. K-Nearest Neighbors KNN Algorithm : [сайт]. — 2019. — Jul 6. — URL: https://medium.datadriveninvestor.com/k-nearest-neighbors-knn-algorithm-bd375d14eec7 (дата обращения: 12.05.2022)

Библиографическая ссылка

Терентьев, Д. А. Оценка кредитоспособности заемщиков // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 184–192.

Сведения об авторах

Терентьев Дмитрий Алексеевич
Студент магистратуры
Направление «Прикладная математика и информатика»

УДК 004.85

Глубокое обучение с подкреплением

Худнев А. С.

Тверской государственный университет

Аннотация. В данной статье предлагается подход к модификации model-based алгоритмов обучения с подкреплением. Реализована модификация алгоритма Dreamer, влияющая на политику исследования среды, используя принцип максимизации энтропии нового состояния. Приведены графики, сравнивающие эффективность оригинального и модифицированного алгоритмов.

Введение

В последние годы глубокое обучение с подкреплением продемонстрировало способность эффективно решать сложные задачи взаимодействия агента со средой, при этом зачастую демонстрируя успехи, превосходящие возможности людей в данных задачах. Так, в 2019 году компания OpenAI продемонстрировала алгоритм OpenAI Five, который смог победить лучшие киберспортивные команды в мире в компьютерной игре Dota 2 [2]. Это и многие другие достижения продемонстрировали потенциал обучения с подкреплением в решении задач сложного итеративного взаимодействия с окружением, менеджмента ресурсов, планирования и других.

Однако, немаловажным ограничением данных алгоритмов является необходимость использования большого количества опыта взаимодействия со средой для обучения агента что делает методы обучения с подкреплением неприменимыми для многих практических задач. Зачастую получение нового опыта взаимодействия со средой является дорогим и долгим. На данный момент имеется ряд подходов, позволяющих уменьшить количество опыта взаимодействия со средой, необходимого для обучения агентов. Так, алгоритмы model-based обучения с подкреплением используют обучаемую модель мира для того, чтобы генерировать искусственный опыт, с помощью которого обучается агент, при этом сама модель мира учится предсказывать динамику среды на реальном опыте.

На данный момент алгоритмы model-based обучения с подкреплением и методы исследования среды исследуются и развиваются отдельно друг от друга. Разработка метода исследования среды, который бы учитывал специфики model-based подхода, может значительно повысить эффективность model-based алгоритмов. Это даст возможность получения эффективных алгоритмов, требующих существенно меньшего количества опыта взаимодействия со средой, чем существующие методы, что позволит в дальнейшем решать более широкий спектр прикладных задач при помощи алгоритмов глубокого обучения с подкреплением.

1. Алгоритм Dreamer

Dreamer — один из новейших model-based алгоритмов обучения с подкреплением. Данный алгоритм состоит из двух ключевых частей: модели мира и агента. Агент учится максимизировать математическое ожидание суммарной дисконтированной награды используя для этого синтезированный с помощью модели мира опыт, а также применяет получаемые моделью мира скрытые представления состояния среды во время взаимодействия с настоящим окружением. В Dreamer модель мира (рис. 1) состоит из нескольких частей. В процессе обучения она учится строить скрытое представление состояния, состоящее из детерминированной части h_i и стохастической части z_i. Детерминированная часть состояния строится рекуррентно при помощи рекуррентных нейронных сетей, которая принимает на вход детерминированную часть предыдущего состояния и преобразованное с помощью полносвязной нейронной сети вектор, являющийся конкатенацией предыдущего состояния и совершенного агентом действия. Стохастическая часть может быть получена из приближаемого моделью мира априорного распределения $p(z_i \mid h_i)$ или апостериорного распределения $p(z_i \mid h_i, o_i)$. Первое распределение применяется в случае, когда модель используется для генерации синтетических траекторий для обучения агента, второе распределение используется во время обучения модели мира или в процессе взаимодействия агента

Модель мира обучается таким образом, чтобы минимизировать адаптированный ELBO-loss и тем самым приблизить распределения

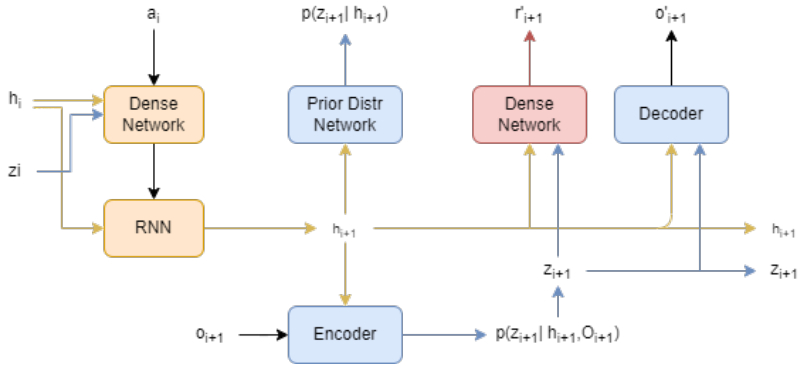

Рис. 1. Схема алгоритма Dreamer.

$p(o_i, z_i \mid h_i)$ (реконструкция наблюдения) и $p(r_i \mid z_i, h_i)$ (распределение получаемых наград за переход в состояния (z_i, h_i)). Для подсчета функции потерь и ее градиента используется случайные отрезки траекторий, полученных во время взаимодействия с настоящей средой и сохраненные в буфере опыта. Для обучения агента используются данные, синтезированные при помощи модели мира. В качестве начальных скрытых состояний для построения синтетических траекторий используются скрытые состояния, полученные из отрезков реальных траекторий, использовавшихся для обновления модели мира. Поскольку модель мира дифференцируема, так как представлена нейронной сетью, есть возможность оптимизировать суммарную награду напрямую, посчитав градиент по совершенным агентом действиям. Чтобы избежать накопления ошибки синтезированных состояний, авторы алгоритма предлагают использовать λ-returns — метод оценки суммарной награды на основе последовательности переходов:

$$TD_\lambda = (1-\lambda)\sum_{k=1}^{n}\lambda^k\left[\gamma^k \cdot V_\pi(s^k) + \sum_{i=0}^{k-1}\gamma^i r^i\right] + \\ + \lambda^n\left[\gamma^n \cdot V_\pi(s_n) + \sum_{i=0}^{n-1}\gamma^i r^i\right]. \quad (1)$$

При этом агент использует две нейронные сети: актора, отвечающего за политику, и критика, приближающего функцию значений. Стоит отметить, что при обновлении актора оценки суммарной награды для всех состояний в синтетических траекториях усредняются. При обучении критика полученные в ходе обновления актора оценки суммарных наград используются в качестве значений целевой функции для подсчета функции ошибки.

Для исследования среды алгоритмом Dreamer используется случайный шум, аддитивно добавляемый к совершаемым агентом действиям. При этом важно отметить, что несмотря на то, что политика актора является стохастичной, ее энтропия не регуляризируется.

2. Максимизация энтропии состояния

Основным преимуществом алгоритмов model-based обучения с подкреплением является то, что для обучения агента они требуют существенно меньшего количества опыта взаимодействия со средой, чем классические model-free алгоритмы. Тем не менее, на данный момент современные model-based методы не используют каких-либо сложных техник исследования среды, ограничиваясь лишь простыми подходами. Это приводит к тому, что получаемый алгоритмом опыт, используемый в последствии для обучения модели мира, является менее информативным, чем мог бы быть в случае, если бы алгоритм использовал улучшенные методы исследования среды.

Большинство современных методов исследования среды, стремятся повысить новизну получаемого во время взаимодействия со средой опыта различными способами.

Один из способов повысить новизну состояния — максимизировать энтропию новых состояний.

Пусть X — случайная переменная с функцией плотности вероятности p, дисперсия которой лежит на множестве $\chi \in \mathbb{R}^q$. Тогда его дифференциальная энтропия задается как

$$\mathcal{H}(X) = -\mathbb{E}_{x\sim p(x)}[\log p(x)].$$

Поскольку трудно оценить p с помощью многомерных данных, можно использовать оценку энтропии на основе k-ближайших соседей

(k-NN) [3]:

$$\widehat{H_N^k} = \frac{1}{N}\sum_{i=1}^{N}\log\frac{N\times\left\|x_i - x_i^{k\text{-NN}}\right\|_2^q \times \widehat{\pi}^{\frac{q}{2}}}{k\times\Gamma\left(\frac{q}{2}+1\right)} + C_k, \tag{2}$$

$$\widehat{H_N^k} \propto \frac{1}{N}\sum_{i=1}^{N}\log\left\|x_i - x_i^{k\text{-NN}}\right\|_2, \tag{3}$$

где $x_i^{k\text{-NN}}$ — это k-NN для x_i в наборе $\{x_i\}_{i=1}^N$, $C_k = \log k - \Psi(k)$ — это смещение, Ψ — пси-функция, Γ — гамма-функция, q — размерность x.

Чтобы определить внутреннее вознаграждение, пропорциональное оценке энтропии состояния, используя (3), мы следуем идее из статьи [1], которая рассматривает каждый переход как частицу, следовательно, внутреннее вознаграждение выглядит следующим образом:

$$r^i(s_i) = \log(\left\|y_i - y_i^{k\text{-NN}}\right\|_2 + 1, \tag{4}$$

где y_i — сжатое состояние из энкодера, а $y_i^{k\text{-NN}}$ — это k-NN для y_i из буфера сжатых состояний $\{y_1, y_2, \ldots, y_N\}$.

Основная идея данного представления заключается в том, что измерение расстояния между состояниями в фиксированном пространстве представлений дает более стабильное внутреннее вознаграждение, поскольку расстояние между данной парой состояний не меняется во время обучения.

Модифицируем политику агента, аддитивно добавляя энтропию состояния к ожидаемой награде, таким образом заставляя агента исследовать среду. Тогда общая награда высчитывается так:

$$r_j^{total} = r^e(s_j, a_j) + \beta_t r^i(s_j), \tag{5}$$

где $r^e(s_j, a_j)$ — награда, вычисленная нейросетью, $\beta_t \geqslant 0$ — гиперпараметр, который определяет компромисс между разведкой и эксплуатацией во время обучения.

3. Результаты

На рис. 2 представлены сравнительные графики обучения модифицированного (dreamer_) и оригинального (dreamer) алгоритма Dreamer в среде OpenAi Gym Montezuma Revenge.

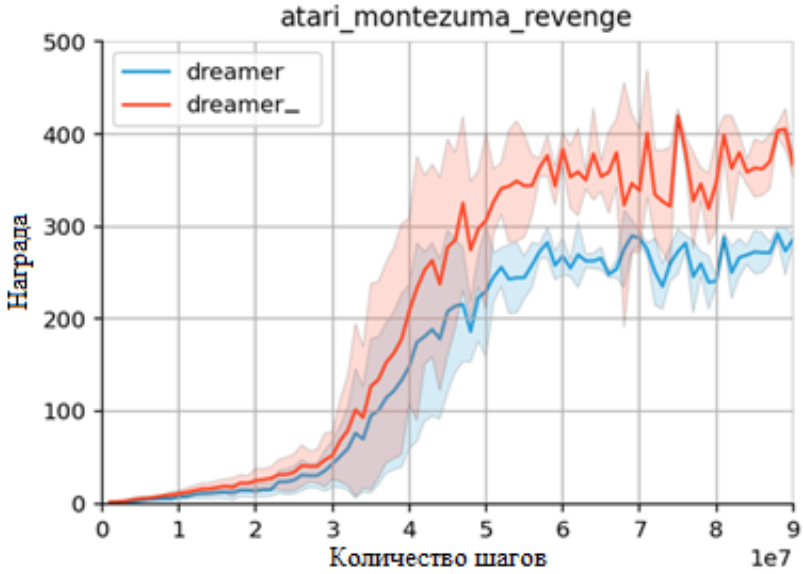

Рис. 2. График сравнения эффективности алгоритмов.

Сплошная линия и заштрихованные области представляют среднее значение и стандартное отклонение, соответственно, для пяти прогонов.

Заключение

В этой статье рассмотрен model-based алгоритм обучения с подкреплением Dreamer. Также описан метод, модифицирующий политику исследования среды агентом. Показано, что улучшенный метод исследования среды в model-based алгоритмах обучения с подкреплением, напрямую влияет на эффективность их работы. Данный метод можно использовать и в других алгоритмах обучения с подкреплением использующих model-based архитектуру.

Список литературы

[1] Liu, H. Behavior from the void: Unsupervised active pre-training / H. Liu, P. Abbeel // Advances in Neural Information Processing Systems / Eds. Ranzato M. [et al.] — Red Hook, NY : Curran Associates, Inc., 2021. — P. 18459–18473.

[2] Berner, C. Dota 2 with Large Scale Deep Reinforcement Learning / C. Berner, G. Brockman, B. Chan [et al.] — 2019. — URL: arXiv: 1912.06680. — Загл. с титул. экрана.

[3] Singh, H. Nearest Neighbor Estimates of Entropy / H. Singh, V. Hnizdo, A. Demchuk, N. Misra // American Journal of Mathematical and Management Sciences. — 2003. — Vol. 23, № 3. — P. 301–321.

Библиографическая ссылка

Худнев, А. С. Глубокое обучение с подкреплением // Студенческая конференция факультета ПМиК. Сборник трудов. — Тверь : ТвГУ, 2022. — С. 193–199.

Сведения об авторах

Худнев Алексей Сергеевич
Студент магистратуры
Направление «Фундаментальная информатика и информационные технологии»

Научное издание

Студенческая конференция факультета ПМиК. Сборник трудов

Тверь
18–29 апреля 2022 г.

Под редакцией Б. Н. Карлова

Подписано в печать 23.05.2022
Усл. п. л. 11,63. Уч.-изд. л. 6,7.
Тираж 10 экз. [электр.].
Заказ № 116 от 21.05.2022

Тверской государственный университет
Издательство Тверского государственного университета
Адрес: 170100, г. Тверь, Студенческий пер., 12, корпус Б
Тел.: (4822) 35-60-63