минобрнауки россии

федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

|  |  |
| --- | --- |
| Институт (факультет) | Институт Информационных Технологий |
| Кафедра | Математического и Программного Обеспечения ЭВМ |

**КУРСОВАЯ РАБОТА**

|  |
| --- |
| по дисциплине Математические методы решения задач |
| искусственного интеллекта |

|  |  |
| --- | --- |
| на тему | Математические методы в машинном обучении |

|  |
| --- |
| Выполнил студент группы |
| 1ПИб-02-3оп-22 |
| направление подготовки (специальности) |
| 09.03.04, Программная инженерия |
| *шифр, наименование* |
| Маркелов Сергей Александрович |
| *фамилия, имя, отчество* |

|  |
| --- |
| Руководитель |
| Юдина Ольга Вадимовна |
| *фамилия, имя, отчество* |
| к. т. н., доцент |
| *должность* |

|  |
| --- |
| Дата представления работы |
| «\_\_23\_\_»\_\_\_\_\_\_\_декабря\_\_\_\_\_2024 г. |
|  |
| Заключение о допуске к защите |
|  |
|  |
|  |
|  |
| Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| количество баллов |
| Подпись преподавателя\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Череповец, 2024

*Год*

Аннотация

Данную курсовую работу по дисциплине «Математические методы решения задач искусственного интеллекта» на тему «Математические методы в машинном обучении» выполнил студент группы 1ПИб-02-3оп-22 Института информационных технологий Череповецкого государственного университета Маркелов Сергей Александрович.

Целью курсовой работы является освоение математических методов, применяемых в машинном обучении: регрессии, кластеризации и классификации.

Оформление курсовой работы выполнялось согласно учебно-методическому пособию.

При создании использовалось Техническое задание, описанное в приложении.

Оглавление

[Введение 4](#_Toc185730780)

[1. Описание данных 6](#_Toc185730781)

[2. Формирование новых признаков 12](#_Toc185730782)

[3. Выбор задач машинного обучения 13](#_Toc185730783)

[4. Задача классификации 18](#_Toc185730784)

[5. Задача кластеризации 26](#_Toc185730785)

[6. Задача регрессии 32](#_Toc185730786)

[7. Визуализация данных 47](#_Toc185730787)

[8. Алгоритмы решения задач 57](#_Toc185730788)

[9. Интерпретация и использование результатов 69](#_Toc185730789)

[Заключение 76](#_Toc185730790)

[Список литературы 77](#_Toc185730791)

[Приложение. Техническое задание 81](#_Toc185730792)

# Введение

Машинное обучение – это раздел искусственного интеллекта, который занимается разработкой методов и алгоритмов, позволяющих компьютерам учиться на основе данных и на их основе прогнозировать результаты или выявлять закономерности [1].

Машинное обучение является одной из наиболее быстро развивающихся областей современной науки и техники, находящейся на стыке информатики, математики и статистики. Благодаря машинному обучению возможно создание алгоритмов, способных учиться на основе данных и принимать решения без явного программирования. Это открывает широкие перспективы для применения в различных областях, таких как медицина, финансы, транспорт, промышленность и многие другие.

Математические методы – это совокупность аналитических инструментов и моделей, основанных на математическом аппарате, которые позволяют решать задачи обработки данных, оптимизации и поиска зависимостей.

Основой машинного обучения являются математические методы, которые включают:

* линейную алгебру – для работы с векторами, матрицами и тензорами;
* математический анализ – для исследования функций, производных и градиентов, что важно для оптимизации моделей;
* теорию вероятностей и математическую статистику – для анализа неопределенности, обработки случайных величин и построения вероятностных моделей;
* методы оптимизации – для нахождения экстремумов функций и минимизации ошибок в алгоритмах машинного обучения.

Понимание математических основ этих методов позволяет не только грамотно использовать существующие алгоритмы, но и разрабатывать новые, более эффективные подходы к решению задач анализа и интерпретации данных.

Цель данной курсовой работы заключается в изучении и анализе основных математических методов, применяемых в машинном обучении, а также их роли в построении и оптимизации алгоритмов.

В рамках работы будет проведен анализ предоставленных данных, выбраны задачи для решения. Будет осуществлен процесс обработки данных, формирование признаков, обучение моделей, а также проведена их оценка.

# Описание данных

Исходные данные для исследования и применения методов машинного обучения представлены в файле online\_sales\_dataset.csv (рис. 1).



Рис 1. Файл с исходными данными

Данный датасет содержит анонимные данные о транзакциях онлайн-продаж. Они включают в себя данные о покупках, характеристики клиентов и параметры заказов.

Набор можно использовать для исследования тенденций в продажах, поведения покупателей и управления заказами в онлайн- или розничной торговле. Это поможет понять, как влияют скидки, способы оплаты и поставщики услуг доставки на эффективность продаж и удовлетворенность клиентов. Также набор можно использовать для анализа эффективности продаж, структуры покупок клиентов и операционной эффективности при управлении заказами.

Перед исследованием данных их необходимо привести в пригодный для обработки вид. Для этого используем встроенную в Excel функцию «Текст по столбцам» во вкладке «Данные» (рис. 2).



Рис. 2. Файл с исходными данными после разбиения на столбцы

В ходе визуального анализа в датасете были обнаружены некорректные данные:

* отсутствие значений в некоторых ячейках;
* наличие в данных отрицательных чисел там, где их быть не может (например, отрицательным не может быть количество товаров или их цена);
* наличие выбросов – аномально больших или аномально малых данных (например, скидка не может быть больше 1, т. е. больше 100%).

Для получения корректных результатов необходимо исправить ошибки в данных:

* для пустых ячеек запишем среднее значение данного признака (в случае с количественными признаками) или его моду, т. е. самое часто встречающееся значение (в случае с категориальными признаками);
* аналогично поступим и с выбросами – запишем вместо них среднее значение признака;
* для отрицательных чисел запишем модуль данного числа (т. е. уберем минус).

Датасет включает в себя данные о 49782 транзакциях, каждая из которых характеризуется набором признаков, описывающих покупки и покупателей. Описание признаков представлено в табл. 1.

Таблица 1

Описание признаков

|  |  |  |  |
| --- | --- | --- | --- |
| Название признака | Тип | Описание | Как можно использовать |
| 1 | 2 | 3 | 4 |
| InvoiceNo | Качественный номинальный | Уникальный идентификатор для каждой операции продажи (счета-фактуры) | - |
| StockCode | Качественный номинальный | Код, обозначающий единицу хранения товара на складе | - |
| Description | Качественный номинальный | Краткое описание продукта | - |
| Quantity | Количественный | Количество единиц товара, проданного в рамках транзакции | Признак помогает определить объем продаж и выявить наиболее популярные товары |
| InvoiceDate | Качественный порядковый | Дата выставления счета | Признак позволяет анализировать сезонные колебания продаж и выявлять тренды |
| UnitPrice | Количественный | Цена за единицу товара в валюте транзакции | Признак важен для анализа выручки, а также для оценки прибыльности товаров |

Продолжение табл. 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| CustomerID | Качественный номинальный | Уникальный идентификатор для каждого клиента | - |
| Country | Качественный номинальный | Страна клиента | Признак может определить географические особенности продаж и выявить закономерности в покупках, связанные с регионом |
| Discount | Количественный | Скидка, применяемая к сделке, если таковая имеется | Признак полезен для анализа влияния скидок на покупательское поведение и объем продаж |
| PaymentMethod | Качественный номинальный | Способ оплаты, использованный для транзакции (банковский перевод, оплата кредитной картой, оплата с помощью PayPal) | Признак может быть полезен для анализа предпочтений клиентов в выборе способа оплаты |
| ShippingCost | Количественный | Стоимость доставки | Признак важен для анализа логистики и оценки влияния стоимости доставки на решение о покупке |
| Category | Качественный номинальный | Категория товара | Признак помогает анализировать спрос на разные категории товаров |
| SalesChannel | Качественный номинальный | Место совершения сделки (онлайн или в магазине) | Признак может быть полезен для анализа различий в покупках через разные каналы продаж |

Продолжение табл. 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| ReturnStatus | Качественный номинальный | Статус возврата (возвращен/не возвращен) | Признак важен для анализа причины возвратов и оценки качества продукции и доставки |
| ShipmentProvider | Качественный номинальный | Поставщик услуг по доставке | Признак позволяет проанализировать, какие поставщики доставки наиболее эффективны и какие из них вызывают наибольшее количество возвратов |
| WarehouseLocation | Качественный номинальный | Расположение склада | Признак может быть полезен для анализа логистики и оптимизации маршрутов доставки |
| OrderPriority | Качественный порядковый | Приоритет заказа (низкий, средний, высокий) | Признак позволяет оценить, насколько срочным является заказ и как это может влиять на процесс обработки и доставки товара. |

Типы признаков разделяются на количественные и качественные [18].

Количественные признаки – это признаки, которые измеряются в числовых значениях и могут быть использованы для математических операций (сложение, вычитание и т.д.).

Качественные (категориальные) признаки – это признаки, которые описывают характеристики объектов и выражаются в виде категорий или классов. Эти признаки не могут быть измерены количественно, но их можно классифицировать. Они делятся на два подтипа:

* номинальные признаки – качественные признаки, которые представляют собой категории, не имеющие естественного порядка, например, страна клиента (Россия, Германия, Франция);
* порядковые признаки – качественные признаки, которые также представляют категории, но имеют определенный порядок или ранжирование, например, степень удовлетворенности (низкий, средний, высокий).

Среди признаков можно выделить следующие взаимосвязи:

* влияние скидок на продажи – признак Discount позволит понять, как применение скидок влияет на количество покупок и доходность;
* категория товара и продажа – признак Category позволит исследовать, какие категории товаров пользуются наибольшим спросом и какие из них чаще возвращаются;
* анализ эффективности поставщиков доставки - признак ShipmentProvider поможет выявить, какие поставщики наиболее эффективны, и есть ли зависимость между поставщиком и количеством возвратов;
* региональные различия в покупках – анализ признака Country может выявить особенности покупательского поведения в разных странах и понять, какие товары популярны в различных регионах.

# Формирование новых признаков

Формирование новых признаков помогает улучшить производительность модели и выделить скрытые закономерности в данных. Иногда модель может не дать хороших результатов с исходными признаками, а создание дополнительных может существенно повысить точность и качество предсказаний [5]. В данном наборе данных можно выделить несколько новых признаков:

* Цена со скидкой (DiscountedPrice) – этот новый признак будет отражать реальную цену товара с учетом примененной скидки. Важно учитывать цену товара с учетом всех скидок, так как это напрямую влияет на прибыльность и поведение покупателей.
* Общая стоимость заказа (TotalCost) – этот признак позволяет рассчитать общую стоимость заказа с учетом количества единиц товара и цены за единицу с учетом скидки. Это полезно для оценки суммы, которую клиент потратил за конкретный заказ.
* Месяц и день недели (Month и DayOfTheWeek) – из признака, связанного с датой покупки, можно выделить информацию о месяце и дне недели. Это поможет выявить сезонные тренды и особенности покупок в различные дни недели.
* Суммарные скидки по заказам (TotalDiscount) – этот признак позволяет вычислить общую скидку на каждый заказ. Это может помочь в анализе, как скидки влияют на поведение покупателей и на прибыль.

Введение данных признаков позволит лучше понять поведение клиентов и повысить точность прогнозов.

# Выбор задач машинного обучения

К основным задачам машинного обучения относятся классификация, кластеризация и регрессия.

Классификация – это задача машинного обучения, в которой цель заключается в присваивании объекту одного из заранее заданных классов. То есть, задача состоит в том, чтобы на основе входных данных предсказать, к какому классу принадлежит объект. Используется в задачах, где нужно разделить данные на категории (например, больные/здоровые в медицине, позитивный/негативный отзыв в анализе настроений) [12].

Кластеризация – это задача группировки объектов, где объекты, находящиеся близко друг к другу, объединяются в один кластер. В отличие от классификации, здесь нет заранее определённых классов. Задача кластеризации состоит в том, чтобы обнаружить структуры или паттерны в данных, такие как группы схожих пользователей или товаров. Кластеризация используется для сегментации клиентов, анализа покупательского поведения, поиска групп товаров с похожими характеристиками или выявления аномальных данных [13].

Регрессия – это задача машинного обучения, где цель состоит в предсказании числового значения на основе входных данных. Например, можно предсказать сумму покупки в зависимости от количества товаров, скидки и категории товара. Это задача, где целевая переменная является непрерывной, а не категориальной. Регрессия применяется для прогнозирования числовых значений, таких как предсказание продаж, температуры, стоимости недвижимости и других количественных данных [29].

В рамках анализа транзакционных данных онлайн-продаж можно выделить несколько возможных задач машинного обучения.

1. Задача классификации: прогнозирование популярности различных категорий товаров

Можно классифицировать товары по их категориям, что позволяет понять, какие товары более популярны и в какие сезоны. Это поможет предсказать, товар какой категории купит клиент, опираясь на данные о цене, скидке, времени покупки и других факторах.

В данном случае подходит задача классификации, так как задача сводится к предсказанию категориального признака (категории товара). Для такой задачи хорошо подходят методы классификации, которые учитывают большое количество входных признаков.

Для решения задачи классификации можно использовать следующие методы:

* Метод ближайших соседей (K-Nearest Neighbors, KNN) – это алгоритм классификации, который принимает решение, исходя из того, к какому классу принадлежат ближайшие объекты в пространстве признаков. KNN подходит в случаях, когда границы между классами не ясны, и нужно учитывать всю информацию о ближайших объектах. Этот метод можно использовать для предсказания категорий товара, анализируя, насколько схожи текущие транзакции с предыдущими покупками такого же товара [3].
* Логистическая регрессия – это метод классификации, который предсказывает вероятность того, что объект принадлежит к определенной категории. Он основан на логистической функции, которая ограничивает результат от 0 до 1 (например, вероятность). Логистическая регрессия хорошо подходит для задач с бинарными результатами, такими как «возврат/не возврат». Этот метод достаточно прост и дает хорошие результаты при линейных зависимостях между признаками и целевой переменной (например, скидки, цена, количество) [14].
* Деревья решений – это алгоритмы классификации, которые используют структуру дерева для принятия решений. Каждое внутреннее узловое решение дерева делит данные по определенному признаку, пока не будет достигнут лист, в котором содержится итоговое решение. Деревья решений хорошо справляются с задачами, где есть множество факторов, влияющих на результат. Они легко интерпретируемы, что позволяет понять, какие признаки наиболее важны для принятия решений (например, скидки или категория товара). Также они хорошо работают с категориальными и числовыми признаками [20].

1. Задача регрессии: прогнозирование суммы покупки.

Прогнозирование общей суммы, которую клиент потратит в рамках одной транзакции, основываясь на таких признаках, как количество товара, цена, скидки, категория товара и другие. Это поможет в оценке объемов продаж и прибыли, а также для оптимизации запасов и логистики.

Задача прогнозирования суммы покупки является задачей регрессии, так как необходимо предсказать числовое значение, зависимое от ряда факторов.

Для решения задачи регрессии можно использовать следующие методы:

* Линейная регрессия – это метод, который используется для предсказания числовых значений на основе линейной зависимости между зависимой переменной (целевой) и одной или несколькими независимыми переменными (признаками). Линейная регрессия подходит для задач регрессии, где требуется предсказать числовое значение, которое зависит от множества факторов. Этот метод прост в интерпретации и хорошо работает, если данные имеют линейную зависимость. Например, линейная регрессия может быть использована для предсказания общей суммы покупки на основе таких признаков, как количество товаров, скидка, категория товара, метод оплаты и т.д. Также она может помочь в анализе того, как изменение цен, скидок или других факторов влияет на продажи [17].
* Градиентный бустинг – это ансамблевый метод, который строит серию слабых моделей, каждая из которых исправляет ошибки предыдущей. Он использует градиентный спуск для минимизации ошибки. Градиентный бустинг даёт отличные результаты, особенно на сложных данных с нелинейными зависимостями. Он подходит для задач классификации и регрессии, например, для предсказания возврата товаров или суммы покупки [7].
* Случайный лес – это ансамблевый метод, который строит множество деревьев решений и комбинирует их результаты с помощью голосования (для классификации) или усреднения (для регрессии). Каждый элемент в случайном лесе обучается на случайной подвыборке данных и случайной подвыборке признаков. Случайный лес уменьшает переобучение и повышает точность по сравнению с одиночным деревом решений. Этот метод может быть использован для решения задач классификации и регрессии в контексте транзакций и анализа покупок [16].

1. Задача кластеризации: сегментация клиентов

Сегментация клиентов на основе их покупательского поведения поможет эффективно таргетировать маркетинговые кампании, улучшить обслуживание и предложить персонализированные предложения. Для этого нужно сгруппировать клиентов по схожести их покупок, объемам расходов, частоте покупок и другим характеристикам.

Кластеризация подходит для группировки клиентов с учетом их сходства по различным признакам (например, частота покупок, сумма покупок и другие). Методы кластеризации помогут выявить скрытые закономерности в данных.

Для решения задачи кластеризации можно использовать следующие методы:

* K-средних (K-Means) – это алгоритм кластеризации, который разделяет данные на K кластеров, минимизируя внутрикластерное расстояние. Это метод с учителем, который позволяет группировать объекты по сходству. Этот метод полезен, когда нужно разделить клиентов на группы, например, по покупательскому поведению или предпочтениям. Это поможет выявить сегменты клиентов, с которыми можно работать индивидуально (например, для таргетинга рекламных акций). Метод прост в реализации и дает быстрые результаты для выявления скрытых закономерностей в данных [15].
* Иерархическая кластеризация – это метод, который строит иерархию кластеров. Алгоритм создает дерево, показывающее, как группы данных объединяются друг с другом. Иерархическая кластеризация хороша для задач, где нужно выявить не только сегменты, но и взаимосвязи между ними. Она полезна, если необходимо посмотреть на более глубокие уровни сегментации данных, например, для более детализированного анализа клиентских сегментов [9].
* DBSCAN – это алгоритм кластеризации, основанный на плотности, который группирует соседние точки в кластеры, если они находятся на достаточном расстоянии друг от друга и имеют достаточную плотность. Точки, которые не принадлежат ни одному кластеру, считаются выбросами или шумом. DBSCAN подходит для выявления групп клиентов с похожим поведением (например, по типу покупок) и обнаружения аномальных транзакций (мошенничество или необычные паттерны) [21].

# Задача классификации

В данной задаче классификации рассматривается набор данных о транзакциях онлайн-продаж. Нам нужно классифицировать товары по категориям, то есть предсказать, к какой категории относится каждый товар, на основе различных признаков.

Целевая переменная – категория товара.

Для решения задачи можно использовать различные признаки, такие как:

* Quantity (количество) – это количество проданных единиц товара. Признак может быть полезен, так как более популярные товары (с большим количеством продаж) могут принадлежать определенным категориям.
* UnitPrice (цена товара) – цена товара часто зависит от категории. Например, товары в категории «Электроника» могут иметь более высокие цены, чем товары в категории «Одежда».
* Discount (скидка) – товары с более высокими скидками могут часто быть из категории распродаж, например, «Одежда».
* Month и DayOfTheWeek (месяц и день недели) – сезонность и поведение покупателей могут сильно зависеть от времени года или дня недели.

Для решения данной задачи был выбран метод K ближайших соседей (KNN). Данный метод обладает следующими преимуществами:

* KNN является достаточно простым и понятным методом. Модель не требует сложных математических преобразований, и её результаты легко интерпретируются.
* KNN хорошо работает с данными, где нет явных линейных зависимостей между признаками. Например, взаимоотношение между скидкой и возвратом товара может быть нелинейным, и KNN хорошо справляется с такими задачами.
* KNN можно адаптировать для различных типов данных, что подходит для рассматриваемой задачи с разнообразными признаками.
* Для небольших и средних объёмов данных KNN может показать хорошие результаты с минимальной настройкой.

Для выполнения классификации необходимо выбрать одну из метрик:

* Евклидово расстояние – это метрика, которая измеряет прямую линию между двумя точками в многомерном пространстве [8]. Оно рассчитывается по формуле (1):

где d – расстояние; pi и qi – координаты точек в i-том измерении; n – количество измерений (признаков).

* Манхэттенское расстояние – это сумма абсолютных разностей координат между двумя точками [19]. Оно рассчитывается по формуле (2):

где d – расстояние; pi и qi – координаты точек в i-том измерении; n – количество измерений (признаков).

Для решения задачи было выбрано евклидово расстояние по следующим причинам:

* Евклидово расстояние подходит, когда признаки измеряются в схожих единицах и отражают реальные «расстояния» между объектами. Например, разница в значениях признаков таких, как цена или количество, должна интерпретироваться как прямое расстояние между точками.
* Евклидово расстояние более точно учитывает отклонения между точками, особенно если данные предварительно нормализованы.
* Манхэттенское расстояние лучше работает с дискретными или категориальными признаками, а в нашем наборе данных многие признаки (например, цена, скидка, количество) являются непрерывными. Поэтому использование манхэттенской метрики могло бы снизить точность модели.

Данные для классификации представлены на рис. 3. Классификацию будем производить с использованием средств языка программирования Python.

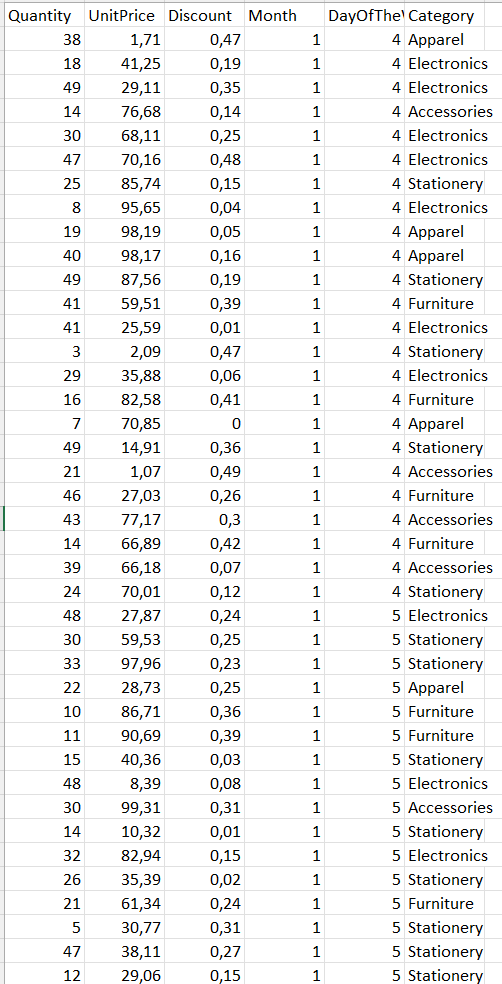


Рис. 3. Данные для классификации

В процессе классификации программа разбивает все товары на 2 класса: невозвращенные (0 класс) и возвращенные (1 класс). Для построения модели из 49782 товаров было отобрано 34847 товаров для обучения модели и остальные 14935 использовались для классификации.

Программа рассчитала значения точности при разных коэффициентах К (количестве соседей) от 1 до 30. Оптимальное количество соседей – 10.

После выполнения классификации получаем матрицу ошибок (рис. 4).

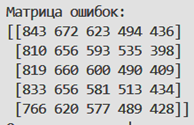


Рис. 4. Матрица ошибок при классификации

Количество строк и столбцов равно количеству классов. Сумма элементов каждой строки равна количеству элементов, которое на самом деле относится к данному классу. Сумма элементов каждого столбца равна количеству элементов, которое модель посчитала относящимся к данному классу. Следовательно, элементы, которые модель распознала верно, находятся на главной диагонали матрицы.

Из матрицы на рис. 4 следует, что программа верно распознала 843 элемента 0 класса, 656 элементов 1 класса, 600 элементов 2 класса, 513 элементов 3 класса и 428 элементов 4 класса. Всего правильно было классифицировано 3040 элементов. Все остальные элементы программа отнесла к неверным классам.

Лучше всего модель справилась с распознаванием элементов 0 класса. Хуже всего – с элементами 4 класса. Это может говорить о различной выразительности признаков, которые различают класс 4.

Помимо матрицы ошибок программа составила отчет по классификации (рис. 5).

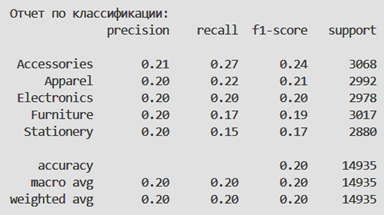


Рис. 5. Отчет по классификации

Precision (точность) показывает, насколько точно модель прогнозирует положительный класс (в данном случае, возврат товара). Если precision для возврата товара низкий, это означает, что модель часто ошибается, предсказывая возврат, когда на самом деле его нет. В этом случае может потребоваться дополнительная работа с признаками или настройка модели.

Программа рассчитала точность для разных классов:

* Accessories – точность 0,21. Только 21% предсказаний Accessories оказались верными;
* Apparel – точность 0,2. Только 20% предсказаний Apparel оказались верными;
* Electronics – точность 0,2. Только 20% предсказаний Electronics оказались верными;
* Furniture – точность 0,2. Только 20% предсказаний Furniture оказались верными;
* Stationery – точность 0,2. Только 20% предсказаний Stationery оказались верными;

Recall (полнота) показывает, как хорошо модель находит все положительные примеры. Если recall для возвратов товара низкий, это означает, что модель пропускает часть реальных возвратов. Это тоже требует внимания, так как может привести к недооценке возвратов.

Программа рассчитала полноту для разных классов:

* Accessories – полнота 0,27. Только 27% всех объектов класса Accessories были правильно классифицированы;
* Apparel – полнота 0,22. Только 22% всех объектов класса Accessories были правильно классифицированы;
* Electronics – полнота 0,2. Только 20% всех объектов класса Accessories были правильно классифицированы;
* Furniture – полнота 0,17. Только 17% всех объектов класса Accessories были правильно классифицированы;
* Stationery – полнота 0,15. Только 15% всех объектов класса Accessories были правильно классифицированы;

F1-score — это баланс между точностью и полнотой. Это важный показатель, который учитывает как ложные срабатывания, так и пропущенные возвраты. Низкий F1-score может свидетельствовать о том, что модель либо слишком строго классифицирует возвраты (в результате чего много ложных отрицательных), либо слишком либерально классифицирует их (в результате чего много ложных положительных).

Программа рассчитала оценки для разных классов:

* Accessories – оценка 0,24. Это средняя гармоническая точность и полнота, свидетельствующая о низком качестве классификации для этого класса;
* Apparel – оценка 0,21. Это средняя гармоническая точность и полнота, свидетельствующая о низком качестве классификации для этого класса;
* Electronics – оценка 0,2. Это средняя гармоническая точность и полнота, свидетельствующая о низком качестве классификации для этого класса;
* Furniture – оценка 0,19. Это средняя гармоническая точность и полнота, свидетельствующая о низком качестве классификации для этого класса;
* Stationery – оценка 0,17. Это средняя гармоническая точность и полнота, свидетельствующая о низком качестве классификации для этого класса;

Macro avg (макро-среднее) вычисляется как среднее всех показателей для каждого класса, без учета их частоты:

* Precision: 0,2 – модель в среднем правильно предсказывает 20% объектов для каждого класса;
* Recall: 0,2 – lля каждого класса модель в среднем правильно определяет только 20% истинных объектов;
* F1-score: 0,2 – низкая общая эффективность.

Weighted avg (взвешенное усреднение) учитывает количество примеров для каждого класса:

* Precision: 0,2 – даже с учетом количества данных по каждому классу, точность модели не улучшается.
* Recall: 0,2 – модель плохо предсказывает классы даже при учете их пропорций;
* F1-score: 0,2 – также свидетельствует о том, что модель неточная.

Accuracy (точность модели) равна 0,2. Это указывает, что модель верно классифицирует лишь 20% объектов. Показатель невысокий, что говорит о проблемах с качеством модели. Точность вычисляется по формуле (3).

В ходе классификации категории товаров с использованием алгоритма KNN модель продемонстрировала низкую точность. Это можно объяснить несколькими объективными причинами:

* Сильная несбалансированность классов – категории товаров представлены неравномерно. Это приводит к тому, что модель «склоняется» к предсказанию доминирующих классов, игнорируя менее частые категории;
* Неудачный выбор признаков – выбранные признаки могут недостаточно четко различать категории товаров;
* Категории товаров могут не зависеть напрямую от выбранных признаков. Например, клиент может купить аксессуары или электронику в схожих условиях (по цене, скидке или времени покупки), и это делает задачу предсказания категорий крайне трудной;
* Возможно, данные содержат значительные перекрытия между классами. Товары из разных категорий могут иметь похожие цены или скидки, что делает их трудными для различения;

Чтобы улучшить качество классификации и повысить точность модели, можно предпринять следующие шаги:

* Использовать методы балансировки данных – увеличение редких классов, например, метод SMOTE для добавления точек редких классов, или наоборот уменьшение частых классов – удаление некоторых точек из доминирующих классов, чтобы уравновесить данные. Это необходимо для предотвращения ситуации, когда модель обучается лучше на доминирующих классах и игнорирует редкие, как было в нашем случае [32];
* Выделить новые признаки, которые могут лучше отражать закономерности. Например, вместо месяцев можно использовать времена года, а вместо точных значений скидок их категории – низкая, средняя, высокая. Преобразование признаков помогает сделать данные более интерпретируемыми для модели и улучшить её способность выявлять зависимости;
* Если категории товаров слишком маленькие и сильно пересекаются, их можно объединить;
* Проверить, насколько классы пересекаются. Если пересечение сильное – проводить классификацию по такой целевой переменной не имеет смысла, необходимо выбрать другую;
* Если модель KNN не дает хороших результатов, можно попробовать более сложные модели: логистическую регрессию, случайный лес, градиентный бустинг и др.

# Задача кластеризации

Кластеризация клиентов на основе их покупательского поведения позволяет группировать пользователей по схожим характеристикам для оптимизации бизнес-процессов и повышения эффективности маркетинговых кампаний. Главная цель задачи заключается в том, чтобы сгруппировать клиентов по схожести их покупок, объемов расходов, частоте покупок и других значимых характеристик.

Для задачи кластеризации целесообразно использовать алгоритм K-means, который позволяет разделить данные на k кластеров по признаку схожести. Этот метод хорошо подходит для группировки объектов с числовыми характеристиками.

Преимущества данного метода:

* простота реализации и интерпретации;
* эффективен на больших наборах данных с числовыми признаками;
* позволяет заранее задать количество кластеров.

Для кластеризации клиентов могут использоваться следующие признаки:

* Quantity (количество проданных товаров) – для оценки частоты и объема покупок;
* UnitPrice (цена за единицу товара) – влияет на средний чек и прибыль;
* TotalDiscount (сумма скидок) – поможет определить, как скидки влияют на клиентов;
* Category (категория товара) – показывает предпочтения клиентов по типу товаров;
* SalesChannel (место совершения покупки) – выявляет различия в покупательском поведении через разные каналы;
* ReturnStatus (статус возврата) – анализ возвратов для определения лояльности клиентов;
* OrderPriority (приоритет заказа) – поможет определить клиентов с высокими требованиями;
* Month (месяц покупки) – для анализа сезонных трендов;
* DayOfTheWeek (день недели) – для анализа предпочтений покупателей в определенные дни недели.

В отличие от задач классификации и регрессии, в кластеризации отсутствует целевая (зависимая) переменная. Алгоритм самостоятельно находит группы (кластеры) на основе схожести значений независимых переменных.

Файл с подготовленными для кластеризации данными представлен на рис. 6.

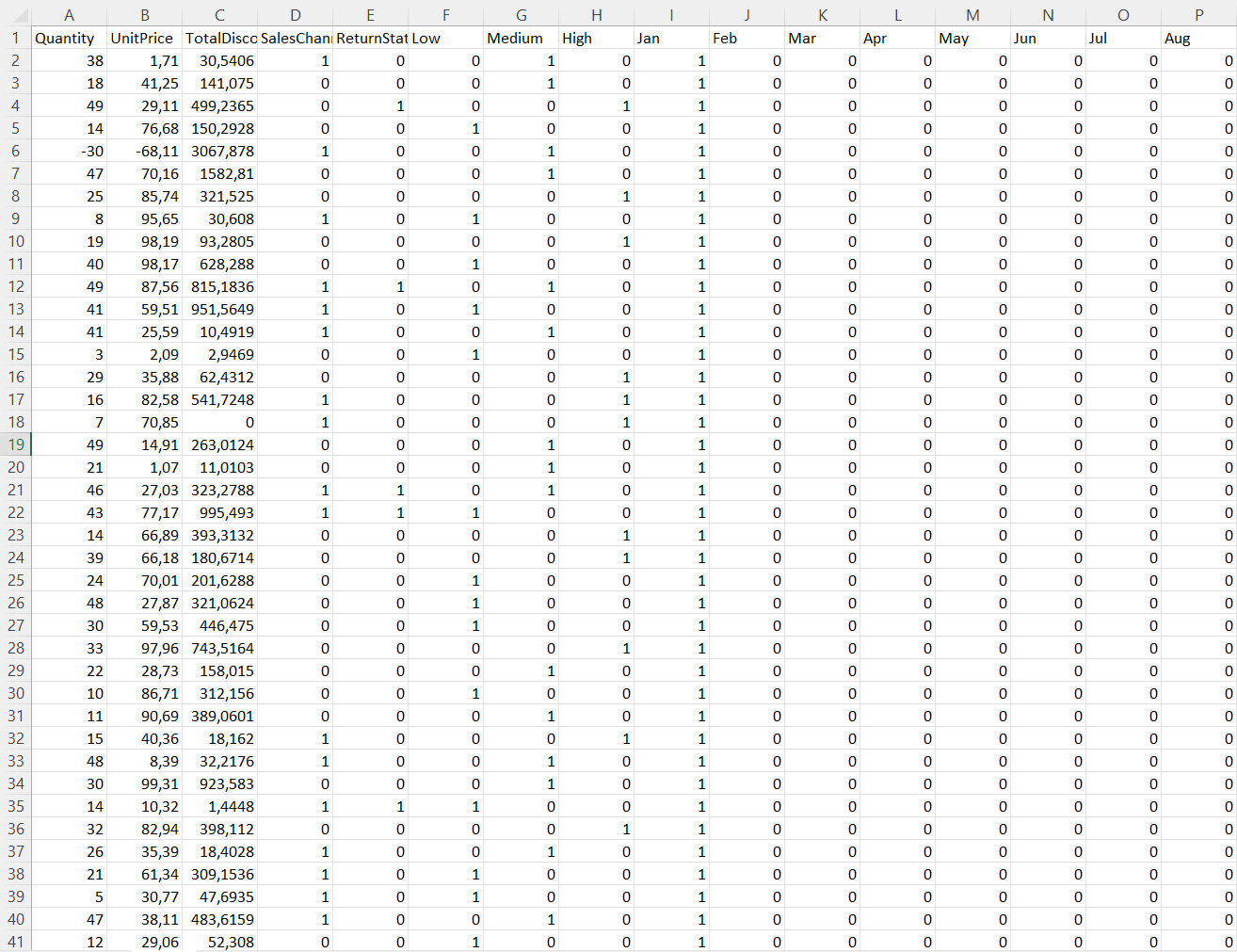


Рис. 6. Данные для кластеризации

Классификацию будем производить с использованием средств языка программирования Python.

Для определения количества кластеров программа подсчитывает силуэтные коэффициенты для каждого числа кластеров (от 2 до 9). Силуэтный коэффициент измеряет, насколько хорошо каждый объект в выборке кластеризуется в своем кластере, по сравнению с другими кластерами. Он варьируется от -1 до 1, где -1 означает, что объект был неправильно отнесен к своему кластеру, 0 указывает, что объект находится на границе 2 кластеров, а 1 означает, что объект идеально относится к своему кластеру.

Границы от 2 до 9 выбраны по следующим причинам:

* чтобы провести кластеризацию, необходимо хотя бы 2 кластера, поэтому минимальное количество кластеров – 2;
* диапазон необходимо ограничить, чтобы не создавать слишком много кластеров, что может привести к излишней детализации. При большом числе кластеров данные начинают дробиться, и количество кластеров становится малозначимым для реального анализа. В качестве максимального значения было выбрано 9 кластеров.

В ходе расчетов программа выбрала 9 кластеров, так как при них достигается наивысшее значение силуэтного коэффициента – 0,17.

Результаты кластеризации (средние значения числовых признаков для каждого кластера) представлены на рис. 7.

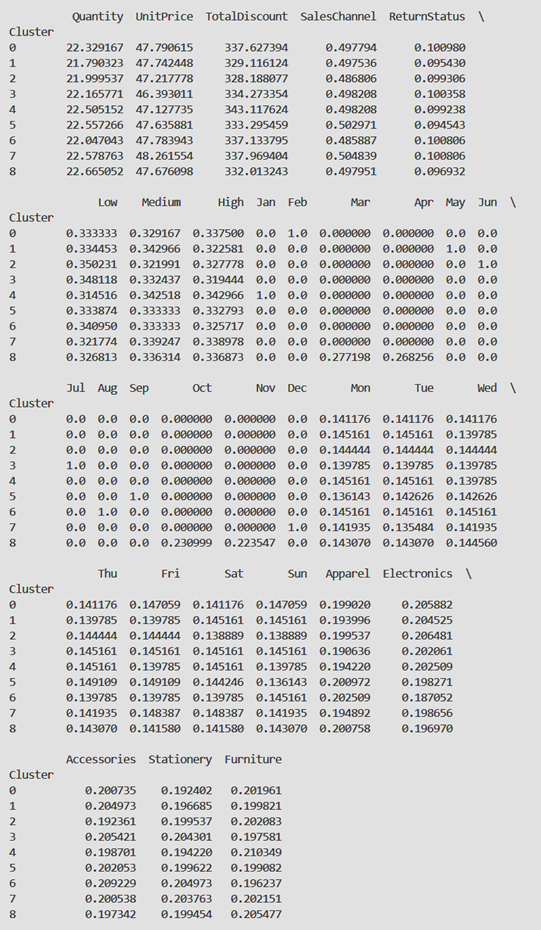


Рис. 7. Средние значения числовых признаков для каждого кластера

Проанализируем результаты кластеризации, чтобы выявить закономерности:

* Количество (Quantity) – в среднем, количество товаров в каждом кластере схоже и варьируется незначительно. Наибольшее значение наблюдается в кластере 8 (22,67), а наименьшее – в кластере 1 (21,79). Однако различия в этом признаке незначительные;
* Цена за единицу (UnitPrice) – все кластеры имеют схожие значения для цен. Цена за единицу варьируется от 46,39 до 48,26, что указывает на схожие ценовые категории в рассматриваемых группах;
* Общая скидка (TotalDiscount) – величины скидок схожи, но в кластере 4 наблюдается несколько более высокий показатель (343,12), что может указывать на наличие больших скидок в этом кластере;
* Канал продаж (SalesChannel) – каналы продаж варьируются от 0.485 до 0.505, но в целом остаются близкими друг к другу;
* Статус возврата (ReturnStatus) – все кластеры имеют схожие значения по статусу возврата, что свидетельствует о том, что возвраты в данной выборке не играют значимой роли в классификации;
* Категории товаров (Apparel, Electronics, Accessories и т.д.) – равномерное распределение товаров по категориям на уровне кластеров. Например, для большинства кластеров показатели по категории Apparel колеблются между 0,19 и 0,2, что показывает стабильное представление этой категории в данных.

Для оценки качества кластеризации используется силуэтный коэффициент s(i), описанный выше [31]. Он рассчитывается для каждого объекта i по формуле (4).

где a(i) – среднее расстояние от объекта i до всех других объектов в своем кластере (внутрикластерное расстояние); b(i) – среднее расстояние от объекта i до объектов в ближайшем соседнем кластере (межкластерное расстояние).

Коэффициент интерпретируется следующим образом:

* s(i) ≥ 0,5 – высокое значение, хорошо разделенные и плотные кластеры;
* 0,1 < s(i) < 0,5 – среднее значение, слабая кластеризация, могут присутствовать пересечения между кластерами;
* s(i) < 0 – низкое значение, ошибочная кластеризация, объекты находятся ближе к чужим кластерам.

В нашем случае s(i) = 0,17 (см. рис. 7), что говорит о слабой кластеризации.

В процессе кластеризации данных были получены результаты, которые не позволили выявить четкие и значимые группы. Данная проблема обусловлена рядом причин, связанных с особенностями набора данных:

* Однородность данных – средние значения числовых признаков для каждого кластера оказались очень схожими, что указывает на отсутствие ярко выраженных паттернов в данных. Это затрудняет разделение объектов на различные кластеры, так как данные не содержат четко выраженных групп;
* Недостаточно информативные признаки – текущие признаки могут недостаточно объяснять различия между группами. Возможно, включение дополнительных признаков могло бы улучшить результаты кластеризации;
* Кластеры перекрываются – объекты разных кластеров находятся слишком близко друг к другу;
* Метрика расстояния не подходит – выбранная метрика расстояния может быть неподходящей для данной задачи.

Для улучшения результатов кластеризации и повышения её информативности можно предпринять следующие шаги:

* Пересмотр признаков – добавление новых признаков, которые лучше отражают различия между группами, таких как демографические данные, поведение покупателей, их предпочтения и т. д.
* Использование других методов кластеризации – попробовать альтернативные методы кластеризации, такие как DBSCAN, который лучше работает с неравномерными данными и может выявить более сложные структуры в данных.

# Задача регрессии

Прогнозирование суммы покупки клиента – важная задача для анализа продаж, управления запасами и повышения прибыли компании. Основная цель – создать модель, которая будет предсказывать итоговую сумму транзакции на основе различных факторов, таких как количество товара, цена, скидки и категория.

Для решения задачи регрессии, заключающейся в прогнозировании суммы покупки, мы выбираем метод линейной регрессии. Линейная регрессия – это один из самых популярных и простых методов машинного обучения, используемый для анализа зависимостей между одной зависимой переменной (целевой) и одной или несколькими независимыми переменными (признаками). Этот метод предполагает, что существует линейная связь между переменными, что делает его удобным и интерпретируемым инструментом для анализа данных.

Преимущества линейной регрессии:

* Простота и интерпретируемость: линейная регрессия дает четкую зависимость между целевой переменной и признаками. Модель позволяет легко понять, как изменение каждого признака влияет на итоговую сумму покупки.
* Быстрота обучения: линейная регрессия требует относительно небольших вычислительных ресурсов, что позволяет быстро обучать модель даже на больших объемах данных.
* Малые требования к данным: линейная регрессия не требует сложной настройки и хорошо работает при отсутствии сильно выраженных зависимостей между признаками. Она также позволяет легко работать с многомерными данными.

Целевой переменной в данной задаче будет сумма покупки (TotalCost), то есть общая сумма, которую клиент тратит в рамках одной транзакции.

Для задачи прогнозирования суммы покупки будем учитывать следующие признаки:

* Количество товара (Quantity) – это количество единиц товара, купленных в одной транзакции. Чем больше единиц товара, тем больше вероятность того, что сумма покупки будет выше. Этот признак напрямую связан с целевой переменной.
* Цена товара (UnitPrice) – играет важную роль в определении итоговой суммы. Чем выше цена, тем больше сумма покупки.
* Скидка (Discount) – может значительно повлиять на итоговую цену покупки. Зачастую скидки используются для стимулирования продаж, и их наличие или величина может оказывать существенное влияние на общую сумму.
* Категория товара (Category) – может включать различные типы продукции, которые могут иметь разные ценовые диапазоны. Этот признак поможет учесть различия в ценах и объемах продаж между категориями товаров.
* Тип канала продажи (SalesChannel) – может повлиять на цену товара и количество покупок.
* Статус возврата (ReturnStatus) – может быть полезным для анализа покупок с возвратами. Покупатели, которые часто возвращают товар, могут влиять на прогнозируемую сумму.
* Месяц (Month) – время года или месяц может влиять на сумму покупки из-за сезонных распродаж, праздников и других факторов. Например, в праздничные месяцы продажи могут быть выше.
* День недели (DayOfTheWeek) – покупки могут зависеть от дня недели, особенно если речь идет о распродажах, которые чаще проходят в выходные.
* Страна (Country) – местоположение может существенно влиять на покупки в зависимости от региона. Например, в одном регионе товары могут быть более популярными, чем в другом.

Перед тем как применить метод линейной регрессии, нужно подготовить и преобразовать данные в подходящий формат для обучения. Некоторые признаки в наборе данных являются категориальными. Для того чтобы применить метод линейной регрессии, нам нужно преобразовать эти категориальные данные в количественные.

Для преобразования будем использовать 2 метода:

* Label Encoding – для категориальных признаков, принимающих 2 значения (SalesChannel, ReturnStatus). Одному из признаков будет присвоено значение 0, а другому – 1 [23].
* One-Hot Encoding – для категориальных признаков, принимающих более 2 различных значений (Country, Category, Month, DayOfTheWeek). Каждый уникальный категориальный признак будет представлен отдельным столбцом, где для каждого значения будет стоять 1 или 0, в зависимости от того, является ли это значением данного столбца [24].

Файл с подготовленными для построения линейной регрессии данными представлен на рис. 8.

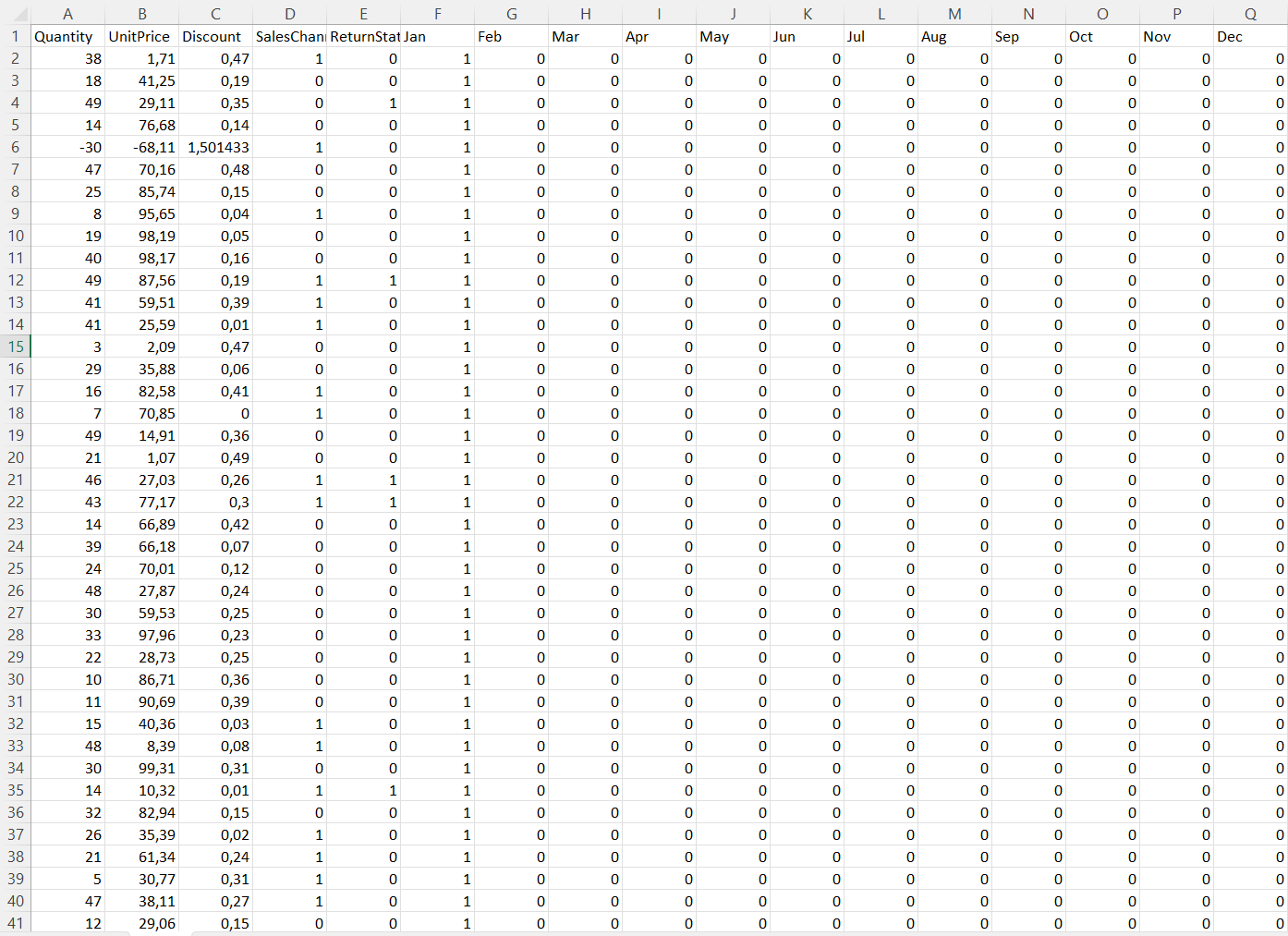


Рис. 8. Данные для построения линейной регрессии

Уравнение линейной регрессии – это математическая модель, которая описывает зависимость между независимой переменной Х и зависимой переменной Y с помощью прямой линии. Уравнение линейной регрессии в простом случае имеет вид (5):

где Y – зависимая переменная; Х – независимая переменная; – свободный член, который представляет собой значение Y, когда X = 0 (пересечение с осью Y); – коэффициент наклона, который показывает, как изменение Х влияет на Y; - ошибка модели, которая отражает отклонение между фактическими и предсказанными значениями.

Для множественной линейной регрессии уравнение расширяется на несколько независимых переменных (6):

где Y – зависимая переменная; Х1, Х2, …, Хn – независимые переменные; n – количество независимых переменных; – свободный член, который представляет собой значение Y, когда все X равны 0 (пересечение с осью Y); – коэффициенты наклона для каждой из независимых переменных, которые показывает, как изменения соответствующих Хi влияет на Y; – ошибка модели, которая отражает отклонение между фактическими и предсказанными значениями.

В нашем случае присутствует 41 независимая переменная, поэтому уравнение регрессии будет иметь вид (7):

Линейную регрессию будем строить с использованием средств языка программирования Python.

Коэффициенты, высчитанные в результате работы программы, представлены на рис. 9.



Рис. 9. Коэффициенты регрессии

Рассмотрим получившиеся коэффициенты:

* Quantity = 27,86 – это означает, что если количество товара увеличится на единицу, то сумма покупки увеличится на 27,86 единиц, при прочих равных. Это довольно высокая величина, которая говорит о сильной связи между количеством товара и суммой покупки, что вполне логично;
* UnitPrice = 11,68 – если цена товара увеличится на единицу, сумма покупки увеличится на 11.68 единиц. Положительный коэффициент говорит о том, что повышение цены товара приводит к увеличению общей суммы покупки, что также логично;
* Discount = -54,10 – если скидка увеличится на 1 единицу, то сумма покупки уменьшится на 54,10 единиц. Отрицательный коэффициент показывает, что скидки уменьшают общую сумму покупки, что опять же логично;
* SalesChannel = -1,31 – это означает, что если канал продаж изменится с онлайн на оффлайн, то это повлияет на сумму покупки. Коэффициент отрицателен, т. е. для оффлайн-продаж сумма покупки будет ниже;
* ReturnStatus = 4,14 – если статус возврата увеличится, сумма покупки может увеличиться на 4,14 единиц. Это может указывать на то, что чаще возвращают товары с более высокой стоимостью;
* Apparel = -3,01 – если доля покупок одежды увеличится на 1 единицу, общая сумма покупки уменьшится на 3,01 единицы. Это означает, что товары в категории одежды, вероятно, имеют меньший вклад в итоговую сумму покупки по сравнению с другими товарами;
* Electronics = 1,86 – если доля покупок электроники увеличится, сумма покупки возрастет на 1,86 единицы. Чем больше товаров в категории электроники, тем выше будет сумма;
* Accessories = 6,45 – если доля аксессуаров увеличится, сумма покупки увеличится на 6,45 единиц. Это значит, что покупатели, как правило, покупают больше аксессуаров, что увеличивает общую стоимость заказа;
* Stationery = -7,08 – канцелярия имеет отрицательное влияние на сумму покупки, то есть товары этой категории, вероятно, имеют низкую цену и, следовательно, меньше влияют на общий объем покупки;
* Furniture = 1,78 – мебель имеет положительное влияние на итоговую сумму покупки. Это логично, так как мебель, как правило, имеет более высокую цену и увеличивает стоимость покупки;
* Australia = -9,10 – заказы из Австралии имеют отрицательное влияние на сумму покупки. Это может указывать на более низкие объемы продаж или скидки, предлагаемые в этой стране;
* Spain = 6,5 – заказы из Испании увеличивают итоговую сумму покупки. Это может быть связано с более высокими ценами или объемами продаж в этой стране;
* Germany = 10,73 – заказы из Германии также увеличивают сумму покупки, возможно, из-за более высоких цен на товары или других факторов;
* Netherlands = -14,15 – заказы из Нидерландов снижают итоговую сумму покупки, что может указывать на особенности цен или объемов продаж;
* United Kingdom = 8,03 – заказы из Великобритании имеют положительное влияние на сумму покупки;
* Sweden = 1,41 – заказы из Швеции слегка увеличивают сумму покупки;
* Belgium = 14,01 – заказы из Бельгии имеют значительный положительный вклад в итоговую сумму покупки;
* Norway = -10,02 – заказы из Норвегии уменьшают сумму покупки;
* Italy = 0,41 – заказы из Италии почти не влияют на сумму покупки (коэффициент близок к нулю);
* Portugal = -12,83 – заказы из Португалии значительно снижают итоговую сумму покупки;
* France = -3,93 – заказы из Франции снижают сумму покупки;
* United States = 8,94 – заказы из США увеличивают сумму покупки;
* Jan = 2,29 – cумма покупки в январе будет на 2,29 единиц выше, чем в другие месяцы;
* Feb = -9,23 – в феврале сумма покупки на 9,23 единицы меньше;
* Mar = -10,41 – в марте сумма покупки на 10,41 единиц меньше;
* Apr = 6,12 – в апреле сумма покупки на 6,12 единиц больше;
* May = 4,06 – в мае сумма покупки на 4,06 единицы больше;
* Jun = -0,59 – в июне сумма покупки на 0,59 единиц меньше;
* Jul = 10,14 – в июле сумма покупки на 10,14 единиц больше;
* Aug = 4,74 – в августе сумма покупки на 4,74 единицы больше;
* Sep = 4,97 – в сентябре сумма покупки на 4,97 единицы больше;
* Oct = -1,26 – в октябре сумма покупки на 1,26 единицы меньше;
* Nov = 6,03 – в ноябре сумма покупки на 6,03 единицы больше;
* Dec = -16,86 – в декабре сумма покупки на 16,86 единиц меньше;
* Mon = -2,51 – в понедельник сумма покупки на 2,51 единицы меньше;
* Tue = 0,5 – во вторник сумма покупки на 0,5 единиц больше;
* Wed = 4,25 – в среду сумма покупки на 4,25 единицы больше;
* Thu = -0,05 – в четверг сумма покупки на 0,05 единицы меньше;
* Fri = -0,62 – в пятницу сумма покупки на 0,62 единицы меньше;
* Sat = 1,59 – в субботу сумма покупки на 1,59 единицы больше;
* Sun = -3,16 – в воскресенье сумма покупки на 3,16 единицы меньше.

Исходя из данных коэффициентов, сделаем выводы:

* Количество товара (Quantity) и цена за единицу (UnitPrice) оказывают сильное положительное влияние на общую сумму покупки. Это логично, поскольку увеличение количества или цены товара ведет к росту итоговой суммы;
* Скидки (Discount), напротив, оказывают значительное отрицательное влияние на итоговую сумму, что также понятно – скидка уменьшает цену покупки;
* ReturnStatus показывает, что возврат товара увеличивает сумму покупки, что может быть связано с тем, что чаще возвращают более дорогие товары, например, электронику;
* Электроника (Electronics) и аксессуары (Accessories) имеют положительное влияние на сумму покупки, что говорит о высокой стоимости этих категорий;
* Канцелярия (Stationery) и одежда (Apparel) снижают сумму покупки. Это может быть связано с тем, что товары в этих категориях часто имеют более низкую цену, чем другие;
* Мебель (Furniture), хотя и влияет положительно, имеет умеренный коэффициент, что указывает на сравнительно высокие цены на мебель, но не на столь высокий уровень продаж;
* Некоторые страны оказывают значительное влияние на итоговую сумму покупки. Например, Германия и Бельгия имеют положительные коэффициенты, что может быть связано с покупательской способностью этих рынков;
* В то же время, Нидерланды и Португалия имеют отрицательные коэффициенты, что может указывать на низкую покупательную способность или особенности рынка в этих странах;
* По месяцам, декабрь выделяется значительным отрицательным коэффициентом, что может быть связано с различными сезонными скидками и распродажами перед Новым Годом и Рождеством;
* В июле наблюдается высокий положительный коэффициент, что может быть связано с летними распродажами или сезонными товарами.
* Дни недели также влияют на покупательскую активность. Например, в среду и вторник покупатели склонны тратить больше, а в воскресенье и понедельник – наоборот, меньше.

Проведем проверку на точность и адекватность модели с помощью коэффициента детерминации.

Коэффициент детерминации (R2) – это статистический показатель, который используется для оценки качества модели регрессии [28]. Он показывает, какая доля изменения целевой переменной (y) объясняется изменениями в независимых переменных (X):

* R2 = 1: модель идеально объясняет все данные, все изменения целевой переменной полностью объясняются независимыми переменными;
* R2 = 0: модель не объясняет никакой части изменений целевой переменной, то есть результаты предсказания не лучше, чем просто усредненное значение целевой переменной;
* R2 < 0: это возможно в случае, когда модель работает хуже, чем простое среднее значение целевой переменной (например, если модель неверно предсказывает все значения); это также может означать, что модель переобучена или выбраны неправильные признаки.

В нашем случае коэффициент R2 = 0,6. Это означает, что модель объясняет 60% вариации целевой переменной, а оставшиеся 40% не могут быть объяснены используемыми признаками. Это достаточно хороший результат для регрессионной модели, однако есть возможности для ее улучшения.

Также рассчитаем уровни значимости для каждой переменной.

Уровень значимости – это статистический показатель, который позволяет оценить, насколько значима независимая переменная в регрессионной модели [34]. По сути, он отвечает на вопрос: можно ли считать влияние данной переменной на целевую переменную статистически значимым, или это влияние произошло случайно?

В регрессионной модели значение p-value связано с гипотезами:

* нулевая гипотеза (H0): коэффициент переменной равен нулю (β = 0), то есть переменная не влияет на целевую переменную;
* альтернативная гипотеза (H1): коэффициент переменной не равен нулю (β ≠ 0), то есть переменная оказывает влияние.

За уровень значимости α примем 0,05. Такой уровень значимости означает, что вероятность ошибки первого рода (отклонить нулевую гипотезу, когда она на самом деле верна) составляет 5%. Если p-значение меньше 0,05, то переменная считается статистически значимой, и нулевая гипотеза отвергается.

Правила интерпретации:

* если p ≤ 0,05 (при уровне значимости 5%) – переменная значима и имеет статистически подтверждённое влияние на целевую переменную;
* если p > 0,05 – переменная незначима, её вклад в объяснение целевой переменной не доказан. В таком случае можно рассмотреть исключение данной переменной из модели.

В результате расчетов программа получила следующие значения уровней значимости:

* Quantity: p ≈ 0 < 0,05 – значение p, близкое к нулю, указывает на статистически значимую связь между переменной Quantity и целевой переменной. Таким образом, количество оказывает существенное влияние на результат и является важным фактором в модели;
* UnitPrice: p ≈ 0 < 0,05 – значение p, близкое к нулю, для переменной UnitPrice также говорит о сильной статистической значимости этой переменной. Цена единицы товара сильно влияет на результат, и её изменение будет существенно влиять на целевую переменную;
* Discount: p = 0,000012 < 0,05 – очень низкое значение p для переменной Discount подтверждает её важность. Это указывает на то, что скидка оказывает значительное влияние на результат и её следует учитывать при построении модели;
* SalesChannel: p = 0,8 > 0,05 – высокое значение p означает, что переменная SalesChannel не имеет статистической значимости. Это высокое p-значение свидетельствует о том, что канал продаж не оказывает существенного влияния на зависимую переменную в данной модели;
* ReturnStatus: p = 0,64 > 0,05 – большое значение p для переменной ReturnStatus указывает на то, что статус возврата товара также не имеет значимого влияния на результат. Это высокое p-значение предполагает, что эта переменная не имеет существенного влияния на целевую переменную;
* Apparel: p ≈ 0 < 0,05 – очень низкое значение p для переменной Apparel свидетельствует о её высокой значимости. Это означает, что категория одежды сильно влияет на результат, и её следует учитывать при анализе;
* Electronics: p ≈ 0 < 0,05 – переменная Electronics также имеет p, близкое к нулю, что подтверждает её статистическую значимость в модели. Категория электроники оказывает значительное влияние на результат;
* Accessories: p ≈ 0 < 0,05 – аналогично предыдущим категориям товаров, Accessories имеет p ≈ 0, что свидетельствует о её высокой значимости. Эта переменная также влияет на предсказания модели;
* Stationery: p ≈ 0 < 0,05 – Stationery с p ≈ 0 также оказывает сильное влияние на целевую переменную, что подтверждает её важность в модели;
* Furniture: p ≈ 0 < 0,05 – переменная Furniture имеет p, близкое к нулю, что указывает на её значимость. Мебель оказывает влияние на результат и должна учитываться при прогнозировании;
* Australia: p = 0,004 < 0,05 – низкое значение p для Австралии подтверждает ее значимость как региона. Это означает, что продажи в Австралии существенно влияют на результат модели;
* Spain: p = 0,28 > 0,05 – значение p для Испании указывает на низкую значимость этой переменной в модели. Таким образом продажи в Испании не оказывают значительного влияния на результат;
* Germany: p = 0,54 > 0,05 – значение p для Германии также достаточно высокое, что указывает на слабую значимость этой переменной в контексте модели;
* Netherlands: p = 0,0005 < 0,05 – очень низкое значение p для Нидерландов подтверждает значимость этой переменной. Продажи в Нидерландах оказывают влияние на результат и поэтому переменная должна быть учтена в модели;
* United Kingdom: p = 0,36 > 0,05 – значение p для Великобритании указывает на то, что эта страна не оказывает значительного влияния на зависимую переменную;
* Sweden: p = 0,09 > 0,05 – значение p немного превышает стандартную границу значимости 0,05. Это означает, что переменная не является статистически значимой на уровне 5%.
* Belgium: p = 0,82 > 0,05 – очень высокое значение p для Бельгии указывает на отсутствие статистической значимости. Эта переменная практически не влияет на результат;
* Norway: p = 0,003 < 0,05 – низкое значение p для Норвегии говорит о значимости этой переменной. Продажи в Норвегии имеют сильное влияние на результат модели;
* Italy: p = 0,08 > 0,05 – p-значение для Италии немного больше 0,05. Это означает, что переменная не является статистически значимой;
* Portugal: p = 0,0009 < 0,05 – очень низкое значение p для Португалии подтверждает статистическую значимость этой переменной. Продажи в Португалии оказывают существенное влияние на результат;
* France: p = 0,02 < 0,05 – значение p для Франции указывает на значимость этой переменной. Франция оказывает определённое влияние на зависимую переменную;
* United States: p = 0,42 > 0,05 – значение p для США указывает на слабую значимость. Продажи в США не оказывают значительного влияния на результат;
* Jan: p = 0,1 > 0,05 – значение p для января указывает на слабое влияние января на результат;
* Feb: p = 0,004 < 0,05 – низкое значение p для февраля подтверждает важность этого месяца. Февраль оказывает существенное влияние на целевую переменную;
* Mar: p = 0,002 < 0,05 – очень низкое значение p для марта также указывает на значимость месяца. Это временная переменная оказывает сильное влияние на результат;
* Apr: p = 0,25 > 0,05 – значение p для апреля указывает на слабое влияние месяца на результат. Эта переменная не оказывает существенного воздействия;
* May: p = 0,16 > 0,05 – значение p для мая также указывает на слабое влияние месяца;
* Jun: p = 0,051 > 0,05 – июнь находится на границе значимости. Месяц оказывает умеренное влияние на результат;
* Jul: p = 0,49 > 0,05 – высокое значение p для июля свидетельствует о слабом влиянии месяца на результат;
* Aug: p = 0,18 > 0,05 – значение p для августа также указывает на слабую значимость;
* Sep: p = 0,23 > 0,05 – значение p для сентября подтверждает слабую значимость этого месяца в модели;
* Oct: p = 0,06 > 0,05 – значение p для октября указывает на слабое влияние месяца на результат;
* Nov: p = 0,28 > 0,05 – значение p для ноября также указывает на слабую статистическую значимость месяца;
* Dec: p = 0,0003 < 0,05 – очень низкое значение p для декабря подтверждает значимость месяца в модели. Этот месяц оказывает существенное влияние на результат;
* Mon: p = 0,000003 < 0,05 – очень низкое значение p для понедельника подтверждает сильную статистическую значимость этого дня недели;
* Tue: p = 0,00003 < 0,05 – значение p для вторника также свидетельствует о его значимости для результата.;
* Wed: p = 0,0003 < 0,05 – для среды наблюдается статистическая значимость, указывающая на ее влияние на зависимую переменную;
* Thu: p = 0,00002 < 0,05 – очень низкое значение p для четверга подтверждает его влияние на результат;
* Fri: p = 0,00001 < 0,05 – пятница также имеет низкое значение p, подтверждающее её значимость;
* Sat: p = 0,00006 < 0,05 – для субботы наблюдается низкое значение p, что указывает на важность этой переменной в модели;
* Sun: p = 0,000002 < 0,05 – очень низкое значение p для воскресенья подтверждает, что этот день недели имеет сильное влияние на результат.

В результате анализа значимости переменных в модели линейной регрессии можно сделать несколько важных выводов:

* Переменные с p-значениями ниже 0,05 оказывают значительное влияние на целевую переменную и являются статистически значимыми. Среди них стоит выделить такие факторы как количество товара, цена, скидка, а также некоторые категории товаров, такие как одежда, электроника, аксессуары, канцелярские товары и мебель. Эти переменные имеют p-значения, близкие к нулю, что свидетельствует о сильном влиянии на результат.
* Также значительное влияние оказывают географические переменные, например Австралия, Нидерланды и Португалия.
* Переменные, чьи p-значения находятся вблизи 0,05, тоже оказывают влияние, хотя оно менее выраженное. К таким переменным можно отнести Швецию, январь и октябрь. Эти переменные все еще статистически значимы, но их влияние слабее, чем у более значимых факторов.
* Переменные с p-значениями выше 0,05 не оказывают статистически значимого влияния на целевую переменную. Это касается, например, канала продаж и статуса возврата. Эти переменные можно рассматривать как незначимые для модели, и их влияние на целевую переменную можно считать пренебрежимо малым.
* Особое внимание следует уделить переменным с экстремально низкими p-значениями, такими как скидка. Они имеют очень сильное влияние на модель, что подчеркивает их важность для точных прогнозов.

Таким образом, анализ показал, что для улучшения модели следует оставить те переменные, которые оказывают значительное влияние на целевую переменную, и исключить те, которые не имеют статистической значимости. Это позволит сделать модель более точной и эффективной для прогноза.

# Визуализация данных

Визуализация данных является важным этапом анализа, который позволяет не только лучше понять структуру и характеристики исходных данных, но и визуально оценить результаты работы модели [6].

Для начала построим различные диаграммы и графики, визуализирующие исходные данные.

На рис. 10 представлена столбчатая диаграмма, показывающая количество клиентов по странам в процентном соотношении.

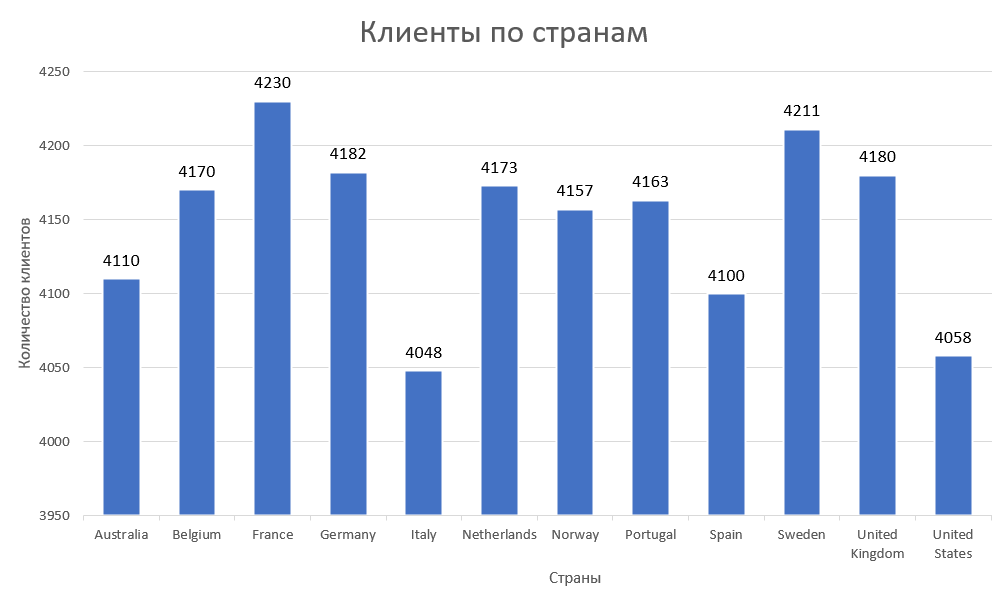


Рис. 10. Количество клиентов по странам

Количество клиентов по странам распределено равномерно и колеблется в пределах 4000-4250 человек.

На рис. 11 представлена круговая диаграмма, показывающая, какими способами оплаты пользуются клиенты.

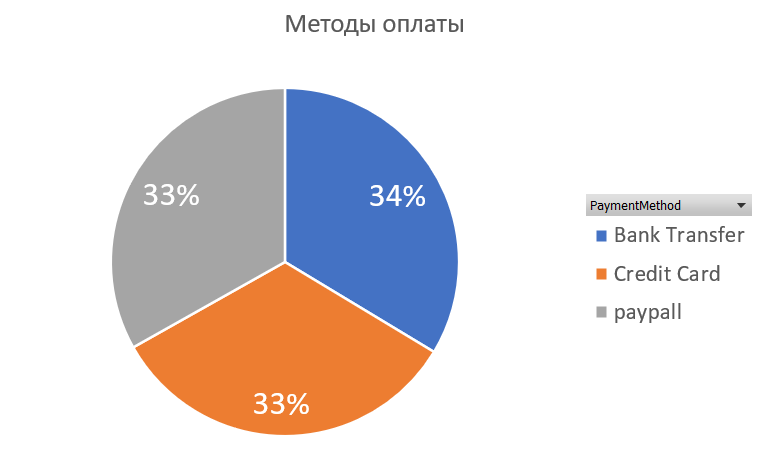


Рис. 11. Способы оплаты

Исходя из диаграммы мы можем сделать вывод, что клиенты в равной степени пользуются всеми доступными методами оплаты.

На рис. 12 представлена столбчатая диаграмма, отображающая количество проданных товаров по различным категориям.

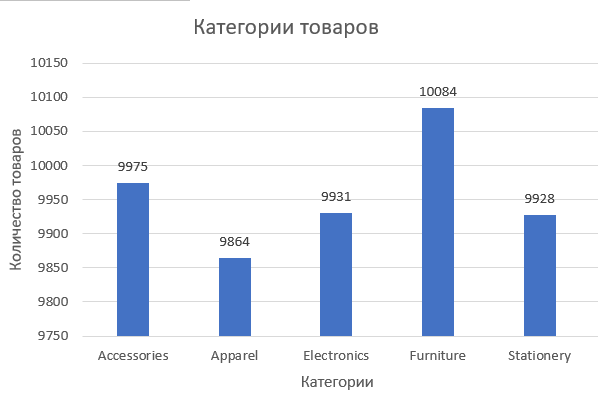


Рис. 12. Количество проданных товаров по категориям

Из данной гистограммы можно сделать вывод, что наиболее популярным товаром является мебель, а наименее популярным – одежда. Однако разница между количеством проданных товаров в разных категориях практически незначительна.

Места приобретения товаров (в магазине или онлайн) в процентном соотношении представлены в круговой диаграмме на рис. 13.

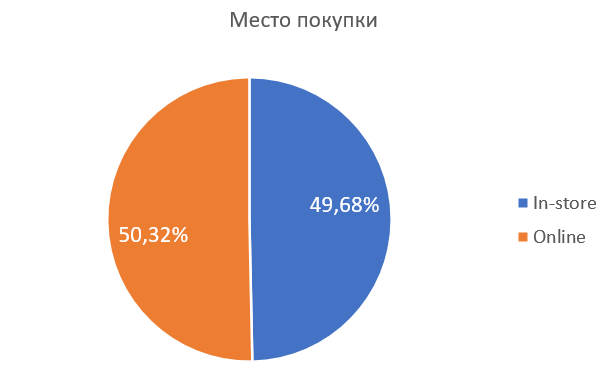


Рис. 13. Места приобретения товаров

Количество покупок, совершенных онлайн и оффлайн, распределилось практически поровну.

На рис. 14 представлена круговая диаграмма с процентным соотношением возвращенных товаров к невозвращенным.



Рис. 14. Количество возвращенных и невозвращенных товаров

Исходя из данной диаграммы мы можем сделать вывод, что в среднем возвращается каждый 10-тый товар.

На рис. 15 представлена гистограмма, отображающая количество покупок в зависимости от месяца.

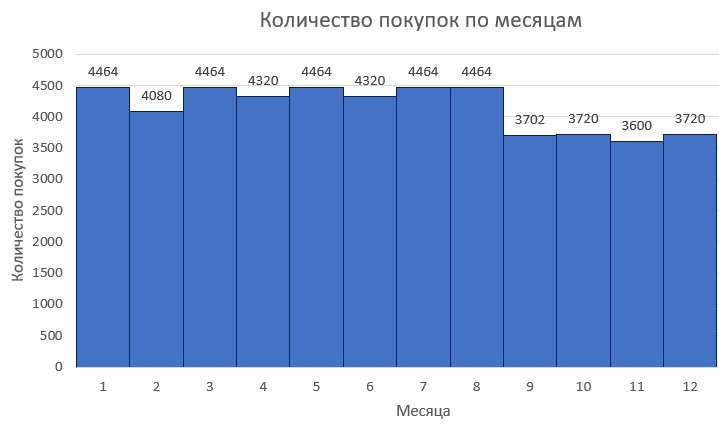


Рис. 15. Количество покупок по месяцам

На данной гистограмме можно увидеть резкий спад количества покупок в последние 4 месяца года. Если с января по август в среднем покупалось 4000-4400 товаров, то с сентября по декабрь – около 3600-3700 товаров.

На рис. 16 представлена гистограмма, отображающая количество покупок в зависимости от дня недели.

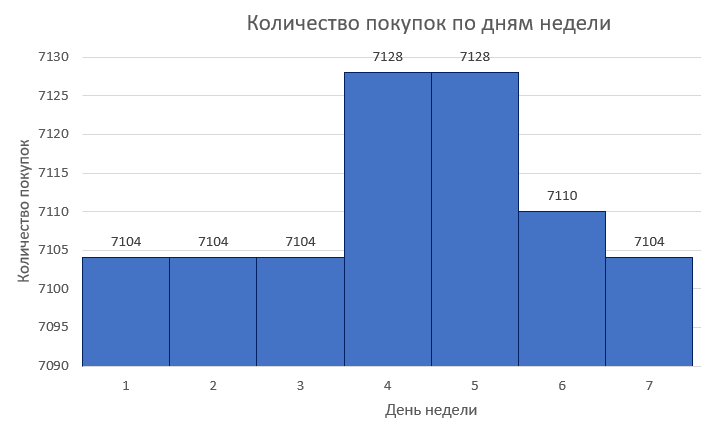


Рис. 16. Количество покупок по дням недели

На данной гистограмме можно увидеть, что количество покупок в разные дни практически равно. Небольшой всплеск наблюдается лишь в четверг и пятницу.

Перейдем к визуализации примененных ранее методов.

На рис. 17 представлен график точности классификации методом KNN при различных значениях коэффициента K – количества соседей.

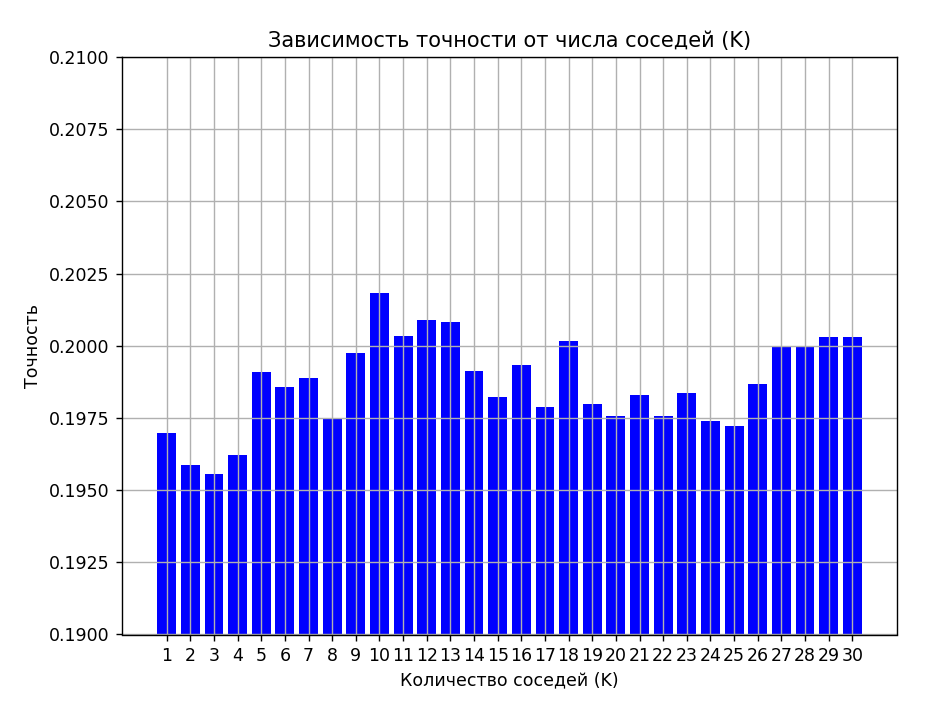


Рис. 17. Зависимость точности от числа соседей

Для классификации построим диаграмму распределения цены товаров по категориям (рис. 18).

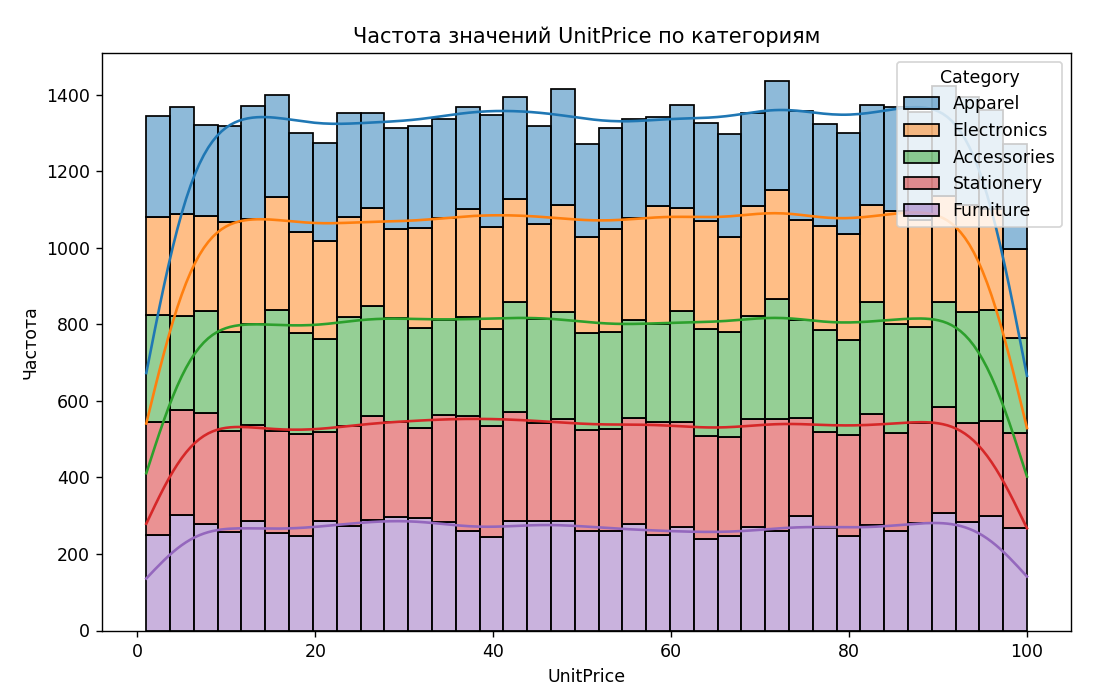


Рис. 18. Цена товаров в зависимости от класса

Матрицу ошибок можно визуализировать в виде тепловой карты (рис. 19).

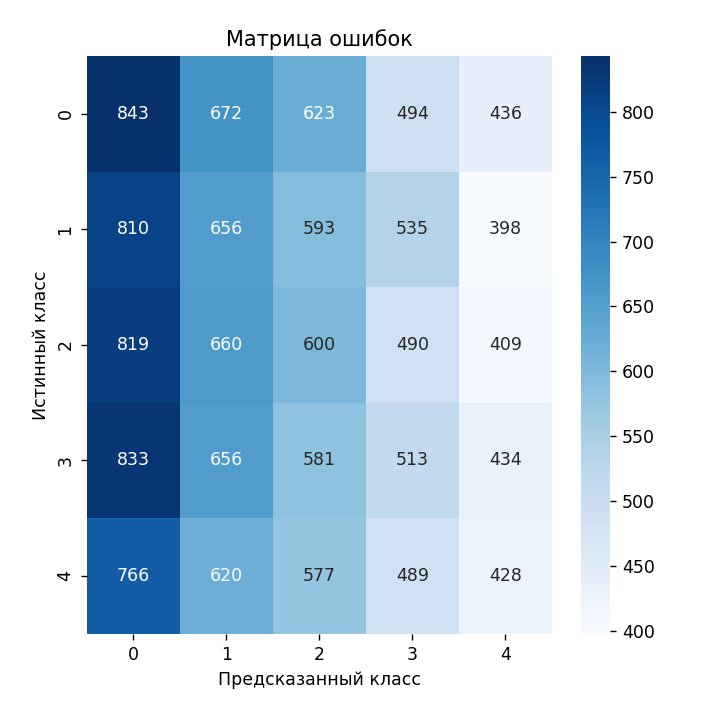


Рис. 19. Матрица ошибок

Для кластеризации построим график силуэтных коэффициентов при разных значениях количества кластеров, чтобы выбрать оптимальное число (рис. 20).

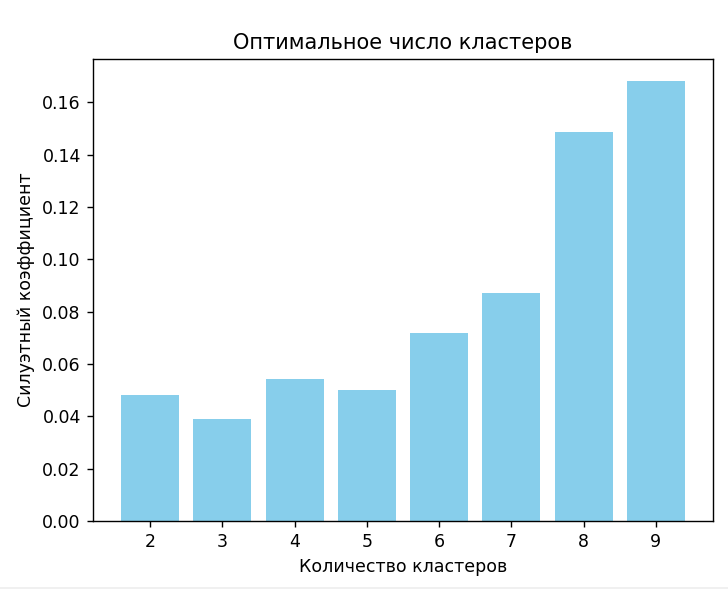


Рис. 20. График силуэтных коэффициентов для разного количества кластеров

Визуализацию кластеров в 2D проведем с помощью метода t-SNE (t-Distributed Stochastic Neighbor Embedding) – это метод для снижения размерности и визуализации данных, который позволяет преобразовать многомерные данные в двумерное или трехмерное пространство с минимальными потерями информации о структуре данных. Группы схожих точек будут отображаться рядом друг с другом в двумерном или трехмерном пространстве.

Визуализация кластеров методом t-SNE в нашей задаче представлена на рис. 21.

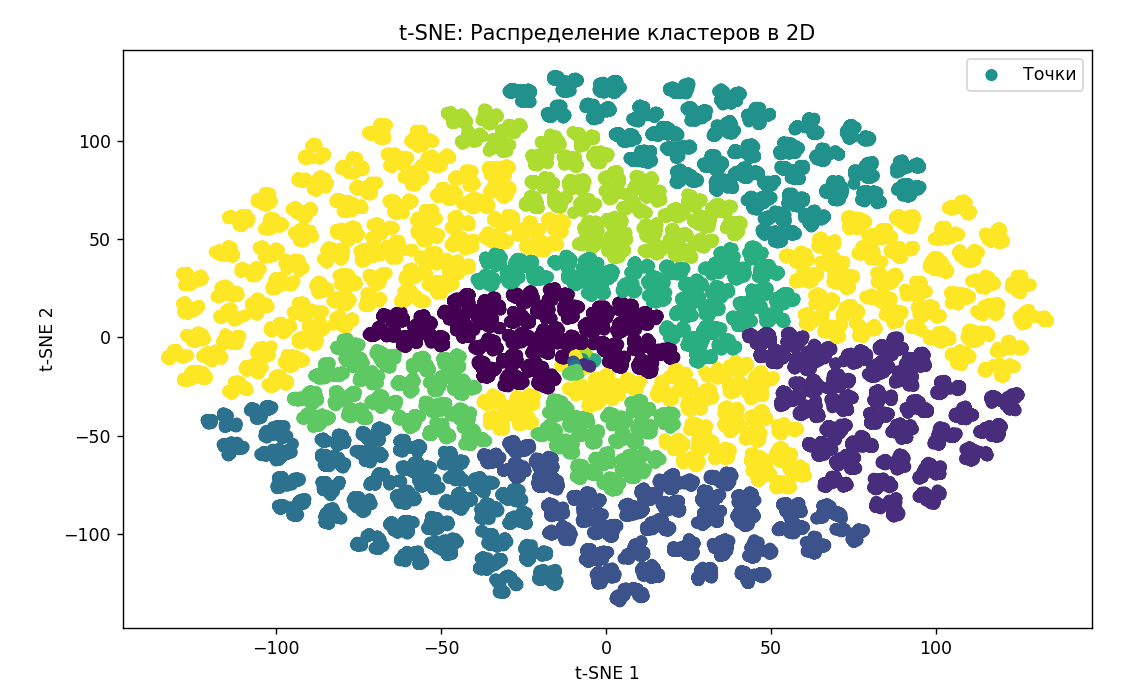


Рис. 21. Распределение кластеров

Для регрессионной модели построим график «Истинные значения vs Предсказанные значения» (рис. 22). Этот график отображает как хорошо модель предсказала целевую переменную на тестовых данных. Если модель идеально предсказывает значения, все точки будут лежать на прямой, которая рисуется как идеальная линия – линия y = x, которая помогает визуально увидеть, насколько близки предсказания модели к истинным значениям.

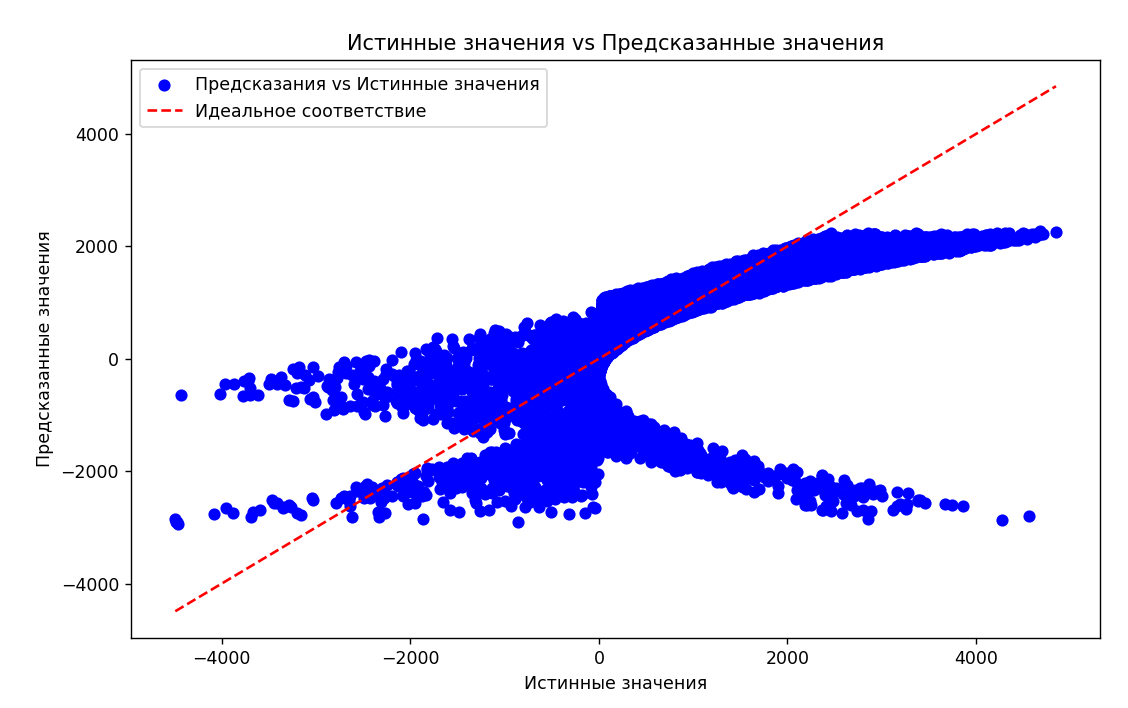


Рис. 22 Истинные значения vs Предсказанные значения

# Алгоритмы решения задач

Все 3 задачи машинного обучения были решены с помощью написания программ на языке программирования Python.

Python – это один из самых популярных языков программирования для машинного обучения, благодаря своей простоте, гибкости и мощным библиотекам, которые делают процесс разработки моделей быстрым и эффективным [25].

Python предлагает множество библиотек, которые делают работу с машинным обучением удобной:

* NumPy – библиотека для работы с многомерными массивами и матрицами. Она используется для выполнения математических операций, таких как линейная алгебра, статистика и случайные числа. В ML NumPy часто используется для предобработки данных, манипуляций с данными и вычислений [35].
* Pandas – библиотека для работы с таблицами данных (DataFrames). Она позволяет удобно загружать, очищать, фильтровать и преобразовывать данные. Pandas помогает обрабатывать структурированные данные, такие как данные в формате CSV или Excel, что важно для подготовки данных в ML [26].
* Matplotlib, Seaborn – эти библиотеки предназначены для визуализации данных. Matplotlib предоставляет базовые инструменты для построения графиков [4], в то время как Seaborn – более высокоуровневая библиотека, основанная на Matplotlib, но с удобным синтаксисом для создания статистических графиков [11].
* Scikit-learn – это одна из самых популярных библиотек для машинного обучения. Она предоставляет готовые алгоритмы для классификации, регрессии, кластеризации, уменьшения размерности, а также инструменты для предобработки данных и оценки модели. Среди алгоритмов есть линейная регрессия, KNN, SVM, деревья решений и ансамбли [30].
* TensorFlow, Keras – эти библиотеки используются для построения более сложных нейронных сетей и глубокого обучения. TensorFlow – это библиотека от Google, которая обеспечивает масштабируемость и оптимизацию для работы с большими данными [33]. Keras – это высокоуровневая обертка, которая упрощает создание и обучение нейронных сетей, работающая поверх TensorFlow [22].
* PyTorch – еще одна популярная библиотека для глубокого обучения, которая используется для построения нейронных сетей. Она поддерживает динамическое вычисление графа, что делает её более гибкой и удобной для исследований [27].

Процесс создания модели машинного обучения можно разделить на несколько этапов [10]:

1. Предобработка данных – это важный этап, включающий загрузку данных, очистку от пропусков, нормализацию или стандартизацию данных, а также преобразование категориальных признаков в числовые (например, с помощью one-hot encoding).
2. Разделение на тренировочные и тестовые данные – обычно данные делятся на две части: одну для обучения модели (тренировочные данные), другую – для проверки качества модели (тестовые данные). Для этого в Scikit-learn есть метод train\_test\_split.
3. Выбор и обучение модели – выбирается алгоритм машинного обучения (например, линейная регрессия, деревья решений, SVM) и обучается модель на тренировочных данных.
4. Оценка модели – после обучения важно оценить качество модели на тестовых данных. Для этого используются метрики, такие как точность, F1-score, ROC-AUC, MSE и другие.
5. Оптимизация модели – на этом этапе можно попробовать улучшить модель, изменяя её гиперпараметры (например, с помощью GridSearchCV в Scikit-learn).

а) Задача классификации

Для реализации классификации была написана следующая программа:

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split, cross\_val\_score

from sklearn.preprocessing import StandardScaler # Импортируем StandardScaler

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

# Загрузка данных

data = pd.read\_excel('KP1.xlsx')

# Разделение данных на независимые и зависимые переменные

X = data.drop('Category', axis=1) # Независимые переменные

y = data['Category'] # Зависимая переменная

# Нормализация данных

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X) # Применяем нормализацию

# Разделение на тренировочную и тестовую выборки

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.3, random\_state=42)

# Перебор возможных значений K с кросс-валидацией

k\_range = range(1, 31)

k\_scores = []

for k in k\_range:

knn = KNeighborsClassifier(n\_neighbors=k)

scores = cross\_val\_score(knn, X\_train, y\_train, cv=5, scoring='accuracy') # 5-fold cross-validation

k\_scores.append(scores.mean())

# Определяем наилучшее значение K

best\_k = k\_range[k\_scores.index(max(k\_scores))]

print(f"Оптимальное значение K: {best\_k}")

# Создание и обучение модели KNN с оптимальным K

knn = KNeighborsClassifier(n\_neighbors=best\_k)

knn.fit(X\_train, y\_train)

# Прогнозирование на тестовых данных

y\_pred = knn.predict(X\_test)

# Оценка точности модели

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Точность модели: {accuracy:.4f}")

# Отчет по классификации

print("Матрица ошибок:")

print(confusion\_matrix(y\_test, y\_pred))

print("Отчет по классификации:")

print(classification\_report(y\_test, y\_pred))

# Матрица ошибок

cm = confusion\_matrix(y\_test, y\_pred)

# Визуализация матрицы ошибок как тепловой карты

plt.figure(figsize=(6, 6))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')

plt.xlabel('Предсказанный класс')

plt.ylabel('Истинный класс')

plt.title('Матрица ошибок')

plt.show()

# График частоты значений UnitPrice для разных классов

plt.figure(figsize=(10, 6))

# Используем seaborn для построения графиков частоты (гистограмм или KDE) для каждой категории

sns.histplot(data=data, x='UnitPrice', hue='Category', multiple='stack', kde=True)

plt.title('Частота значений UnitPrice по категориям')

plt.xlabel('UnitPrice')

plt.ylabel('Частота')

plt.show()

Данная программа выполняет следующий алгоритм:

1. Загрузка данных: данные загружаются из файла Excel с помощью библиотеки pandas. Эти данные содержат как независимые переменные (признаки), так и зависимую (целевую) переменную.
2. Разделение на независимые и зависимые переменные: независимые переменные сохраняются в переменной X, а зависимая переменная – в переменной y. Категория извлекается путем удаления столбца 'Category' из данных.
3. Применяется нормализация (масштабирование) для всех независимых переменных с помощью StandardScaler. Это необходимо для того, чтобы привести все переменные к одному масштабу, что помогает улучшить производительность моделей, использующих расстояние, как в KNN.
4. Разделение данных на тренировочную и тестовую выборки: данные разделяются на тренировочную и тестовую выборки с использованием train\_test\_split(), где 30% данных отводится для тестирования, а 70% - для тренировки.
5. Для каждого значения K (количества соседей) в диапазоне от 1 до 30 вычисляется точность модели с использованием кросс-валидации. Кросс-валидация выполняется с 5 фолдами. Для каждого K вычисляется средняя точность, и она сохраняется в список k\_scores.
6. На основании точности для каждого значения K выбирается оптимальное значение (то, которое дало наибольшую точность).
7. Создается и обучается модель KNN с использованием оптимального значения K, найденного на предыдущем шаге.
8. Модель применяется для прогнозирования классов на тестовых данных, и результаты сохраняются в переменной y\_pred.
9. Оценка точности модели: точность модели оценивается с помощью функции accuracy\_score(). Выводится также матрица ошибок и отчет по классификации, содержащий важные метрики (точность, полнота, F-меры).
10. Визуализация результатов:

* Матрица ошибок: отображается тепловая карта с использованием seaborn.heatmap(), показывающая, как хорошо модель справляется с классификацией для каждого класса.
* График зависимости точности от числа соседей (K): строится столбчатая диаграмма, которая показывает, как меняется точность модели в зависимости от количества соседей (K).
* График распределения средней цены (UnitPrice) по категориям: строится столбчатая диаграмма с помощью seaborn.barplot(), показывающая среднюю цену для каждой категории.Матрица ошибок визуализируется с помощью библиотеки seaborn в виде тепловой карты, где отображаются предсказанные и истинные классы.

б) Задача кластеризации

Для реализации кластеризации была написана следующая программа:

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans

from sklearn.metrics import silhouette\_score

# Устанавливаем максимальное количество отображаемых столбцов

pd.set\_option('display.max\_columns', None)

# Загрузка данных

data = pd.read\_excel('KP2.xlsx')

# Убираем пробелы из имен столбцов

data.columns = data.columns.str.strip()

# Переменные

numerical\_columns = ['Quantity', 'UnitPrice', 'TotalDiscount', 'SalesChannel', 'ReturnStatus',

'Low', 'Medium', 'High', 'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul',

'Aug', 'Sep', 'Oct', 'Nov', 'Dec', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat',

'Sun', 'Apparel', 'Electronics', 'Accessories', 'Stationery', 'Furniture']

# Создаем набор данных с числовыми признаками

X = data[numerical\_columns]

# Нормализация данных

scaler = StandardScaler()

X\_normalized = scaler.fit\_transform(X)

# Выбор оптимального числа кластеров с помощью силуэтного коэффициента

silhouette\_scores = []

cluster\_range = range(2, 10)

for n\_clusters in cluster\_range:

kmeans = KMeans(n\_clusters=n\_clusters, random\_state=42)

labels = kmeans.fit\_predict(X\_normalized)

sil\_score = silhouette\_score(X\_normalized, labels)

silhouette\_scores.append(sil\_score)

# Построение столбчатой гистограммы силуэтного коэффициента

plt.bar(cluster\_range, silhouette\_scores, color='skyblue')

plt.xlabel('Количество кластеров')

plt.ylabel('Силуэтный коэффициент')

plt.title('Оптимальное число кластеров')

plt.show()

# Применение K-Means с оптимальным числом кластеров

optimal\_clusters = cluster\_range[silhouette\_scores.index(max(silhouette\_scores))]

# Применение K-Means с оптимальным числом кластеров

kmeans = KMeans(n\_clusters=optimal\_clusters, random\_state=42)

data['Cluster'] = kmeans.fit\_predict(X\_normalized)

# Анализ кластеров

print(f"\nКоличество точек в каждом кластере:\n{data['Cluster'].value\_counts()}")

# Средние значения по числовым признакам для каждого кластера

cluster\_means = data.groupby('Cluster')[numerical\_columns].mean()

print(f"\nСредние значения числовых признаков для каждого кластера:\n{cluster\_means}")

# Расчет силуэтного коэффициента

sil\_score = silhouette\_score(X\_normalized, data['Cluster'])

print(f"\nСилуэтный коэффициент: {sil\_score}")

# Вычисление корреляции между признаками

correlation\_matrix = X.corr()

# Настройка графика тепловой карты

plt.figure(figsize=(12, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5, square=True)

# Отображаем тепловую карту

plt.title('Тепловая карта корреляции между признаками')

plt.show()

# Визуализация кластеров с использованием PCA

from sklearn.decomposition import PCA

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X\_normalized)

plt.figure(figsize=(10, 6))

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=data['Cluster'], cmap='viridis', label='Точки')

plt.title('PCA: Распределение кластеров в 2D')

plt.xlabel('PC1')

plt.ylabel('PC2')

plt.legend()

plt.show()

# Визуализация кластеров с использованием t-SNE

from sklearn.manifold import TSNE

tsne = TSNE(n\_components=2, random\_state=42)

X\_tsne = tsne.fit\_transform(X\_normalized)

plt.figure(figsize=(10, 6))

plt.scatter(X\_tsne[:, 0], X\_tsne[:, 1], c=data['Cluster'], cmap='viridis', label='Точки')

plt.title('t-SNE: Распределение кластеров в 2D')

plt.xlabel('t-SNE 1')

plt.ylabel('t-SNE 2')

plt.legend()

plt.show()

Данная программа выполняет следующий алгоритм:

1. Загрузка и подготовка данных:
   * Данные загружаются из файла Excel с помощью библиотеки pandas.
   * Убираются пробелы из имен столбцов с помощью data.columns.str.strip().
   * Для анализа выбираются числовые столбцы, указанные в переменной numerical\_columns.
2. Нормализация данных:
   * Применяется стандартная нормализация числовых данных с помощью StandardScaler из библиотеки sklearn, что делает данные более подходящими для алгоритма кластеризации.
3. Выбор оптимального числа кластеров:
   * Используется силуэтный коэффициент (silhouette\_score) для оценки качества кластеризации. Для каждого числа кластеров в диапазоне от 2 до 9 вычисляется силуэтный коэффициент, и строится график зависимости силуэтного коэффициента от числа кластеров.
   * Оптимальное количество кластеров выбирается как число, при котором силуэтный коэффициент максимален.
4. Применение алгоритма K-Means:
   * После выбора оптимального числа кластеров, применяется алгоритм K-Means для кластеризации данных с использованием n\_clusters=optimal\_clusters.
   * Кластеры добавляются в исходный DataFrame как новый столбец Cluster.
5. Анализ кластеров:
   * Выводится количество точек в каждом кластере с помощью value\_counts().
   * Рассчитываются средние значения числовых признаков для каждого кластера с помощью groupby().
   * Вычисляется общий силуэтный коэффициент для всей кластеризации.
6. Корреляция между признаками:
   * Вычисляется корреляция между признаками с помощью метода .corr().
   * Строится тепловая карта корреляции с использованием библиотеки seaborn для визуализации взаимосвязей между признаками.
7. Визуализация кластеров:
   * Для визуализации кластеров используется метод t-SNE (t-distributed Stochastic Neighbor Embedding)
   * Результаты отображаются на графике, где каждый кластер имеет свой цвет, а точки отображаются по осям компонентов.

в) Задача регрессии

Для реализации регрессии была написана следующая программа:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

import statsmodels.api as sm

# Чтение данных из xlsx файла

df = pd.read\_excel('KP3.xlsx')

# Предполагаем, что последний столбец - целевая переменная

X = df.iloc[:, :-1].values # Все столбцы, кроме последнего (признаки)

y = df.iloc[:, -1].values # Последний столбец (целевая переменная)

# Получаем названия столбцов признаков

columns = df.columns[:-1]

# Создание модели линейной регрессии (sklearn)

model = LinearRegression()

# Обучение модели

model.fit(X, y)

# Коэффициенты линейной регрессии

coefficients = model.coef\_

intercept = model.intercept\_

# Прогнозирование на тех же данных

y\_pred = model.predict(X)

# Вычисление R^2 (коэффициент детерминации)

r2 = r2\_score(y, y\_pred)

# Вычисление MSE (среднеквадратичная ошибка)

mse = mean\_squared\_error(y, y\_pred)

# Форматированный вывод с названием столбца и коэффициентом

print("Коэффициенты:")

for col, coef in zip(columns, coefficients):

print(f"{col} = {coef:.6f}")

print(f"\nСвободный член (перехват): {intercept:.6f}")

print(f"\nКоэффициент детерминации R^2: {r2:.6f}")

print(f"\nСреднеквадратичная ошибка MSE: {mse:.6f}")

# Используем statsmodels для расчета p-value

# Добавляем константу (свободный член)

X\_with\_intercept = sm.add\_constant(X)

# Строим модель с использованием statsmodels

stats\_model = sm.OLS(y, X\_with\_intercept).fit()

# Выводим сводную таблицу результатов с большим количеством знаков после запятой

print("\nРезультаты регрессии (statsmodels):")

print(stats\_model.summary2(xname=["const"] + list(columns)).tables[1].to\_string(float\_format='%.20f'))

# Визуализация результатов

plt.figure(figsize=(10, 6))

# График истинных значений vs предсказанных значений

plt.scatter(y, y\_pred, color='blue', label='Предсказания vs Истинные значения')

# Линия y = x для отображения идеального соответствия

plt.plot([y.min(), y.max()], [y.min(), y.max()], color='red', linestyle='--', label='Идеальное соответствие')

plt.xlabel('Истинные значения')

plt.ylabel('Предсказанные значения')

plt.title('Истинные значения vs Предсказанные значения')

plt.legend()

plt.show()

Данная программа выполняет следующий алгоритм:

1. Чтение данных: данные загружаются из Excel-файла KP3.xlsx с использованием pandas.read\_excel(). Ожидается, что последний столбец содержит целевую переменную, а все остальные столбцы – признаки (независимые переменные).
2. Подготовка данных:

* X содержит все столбцы, кроме последнего (признаки);
* y содержит последний столбец (целевая переменная);
* columns сохраняет имена всех признаков.

1. Создание и обучение модели линейной регрессии с использованием sklearn:
   * Создается объект модели LinearRegression из библиотеки sklearn;
   * Модель обучается на данных X и y с помощью метода .fit().
2. Получение коэффициентов модели:
   * После обучения модели выводятся коэффициенты для каждого признака и значение свободного члена (перехвата);
   * Вычисляются предсказания для тех же данных с помощью метода .predict().
3. Оценка качества модели: рассчитывается коэффициент детерминации R2 и среднеквадратичная ошибка (MSE) с помощью функций r2\_score и mean\_squared\_error соответственно.
4. Использование statsmodels для получения статистических показателей:
   * Добавляется константа (свободный член) в матрицу признаков для использования в модели;
   * Строится модель с использованием statsmodels.OLS(), которая позволяет вычислить дополнительные статистические метрики, такие как p-value для каждого коэффициента;
   * Выводится сводная таблица с результатами регрессии, включая коэффициенты, стандартные ошибки, t-статистики и p-value с точностью до 20 знаков после запятой.
5. Визуализация результатов:
   * Строится график, на котором по оси X откладываются истинные значения, а по оси Y — предсказанные значения;
   * Также добавляется линия y = x (красная пунктирная линия), которая представляет идеальное соответствие между истинными и предсказанными значениями. Это позволяет оценить, насколько хорошо модель предсказывает значения.

# Интерпретация и использование результатов

а) Задача классификации

Для решения задачи классификации, модель KNN была использована для предсказания категорий товаров. Оценка работы модели была проведена с использованием различных метрик, таких как точность (precision), полнота (recall), F1-оценка, и точность модели (accuracy).

Матрица ошибок является важным инструментом для понимания, как хорошо модель классифицирует объекты разных классов. Она показывает количество правильных и неправильных предсказаний для каждого класса. Программа верно классифицировала 3040 элементов, при этом лучше всего модель справилась с классификацией товаров 0-го класса (невозвращенные товары), в то время как хуже всего она предсказывала товары 4-го класса. Такой результат может свидетельствовать о том, что признаки, используемые для классификации, не могут достаточно точно различать товары разных категорий, особенно в 4-м классе.

Точность (Precision) – показывает, насколько точно модель предсказывает возвраты товара. Для всех классов точность оказалась низкой (около 20%), что означает, что модель часто ошибается в предсказаниях.

Полнота (Recall) – отражает, как хорошо модель находит все реальные возвраты товара. Для всех классов полнота также невысока (около 20% для каждого класса), что указывает на то, что модель пропускает значительную часть положительных примеров.

Такие результаты говорят о том, что модель не выявляет достаточное количество возвратов и часто делает ложные предсказания, что требует доработки.

F1-оценка — это средняя гармоническая между точностью и полнотой. Она дает более полное представление о качестве классификации, чем отдельные метрики точности и полноты. Низкие значения F1-оценки (около 0,2 для всех классов) указывают на то, что модель имеет значительные проблемы с классификацией. Это может быть связано с неудачным выбором признаков или несбалансированностью классов.

Макро-среднее (Macro avg) – в данном случае средние значения точности, полноты и F1-оценки для всех классов составляют 0,2, что свидетельствует о плохом качестве модели в целом.

Взвешенное усреднение (Weighted avg) – несмотря на то, что это усреднение учитывает количество объектов в каждом классе, модель всё равно демонстрирует низкие результаты по точности (0,2), полноте (0,2) и F1-оценке (0,2).

Точность модели составила 0,2, что означает, что только 20% товаров были правильно классифицированы. Это очень низкий показатель, который свидетельствует о слабой эффективности модели в её текущем виде. Снижение точности может быть вызвано несколькими факторами, такими как несбалансированные классы, недостаточные различия между признаками, или высокое пересечение между классами.

Причины низкой точности модели:

* Несбалансированность классов: в наборе данных классы представлены неравномерно, что может приводить к тому, что модель склоняется к предсказаниям доминирующих классов. Это может стать причиной низкой точности для менее представленных классов.
* Неудачный выбор признаков: признаки, такие как цена, скидка и количество, могут не быть достаточно информативными для различения классов товаров. Возможно, эти признаки не дают модели четкой картины для точного разделения классов.
* Перекрытие между классами: товары разных категорий могут иметь схожие значения признаков, таких как цена или скидка, что делает задачу классификации сложной и приводит к ошибкам в предсказаниях.
* Низкая выразительность признаков: признаки могут не отражать важные аспекты, которые различают товары разных категорий. Например, товары из разных категорий могут часто иметь схожие характеристики, что затрудняет точное их различение.

Рекомендации для улучшения модели:

* Балансировка классов: для улучшения качества модели можно использовать методы балансировки данных, такие как SMOTE (метод увеличения редких классов) или удаление точек из доминирующих классов. Это позволит модели более точно различать редкие классы, избегая доминирования наиболее частых категорий.
* Выделение новых признаков: для улучшения классификации можно выделить новые признаки, которые лучше отражают различия между категориями товаров. Например, можно заменить месяцы на времена года или классифицировать скидки по категориям (низкая, средняя, высокая).
* Объединение классов: если классы слишком малы и сильно перекрываются, можно объединить их в более крупные группы, что может облегчить классификацию.
* Использование более сложных моделей: если KNN не дает удовлетворительных результатов, стоит рассмотреть использование более сложных методов, таких как логистическая регрессия, случайные леса или градиентный бустинг. Эти модели могут быть более эффективными для данной задачи и могут лучше справляться с классификацией сложных данных.
* Пересмотр признаков: стоит внимательно пересмотреть признаки, чтобы убедиться, что они достаточно информативны для классификации. Возможно, стоит использовать другие признаки или преобразовать существующие (например, категориальные данные можно преобразовать в бинарные переменные).

б) Задача кластеризации

Алгоритм K-means работает путем разделения клиентов на k кластеров, где каждая точка данных (клиент) будет отнесена к тому кластеру, в котором центроид (среднее значение всех объектов кластера) находится ближе всего. Количество кластеров k было определено на основе анализа силуэтного коэффициента, который показал оптимальное количество кластеров в 9. Силуэтный коэффициент для каждого клиента измеряет, насколько хорошо он вписывается в свой кластер, по сравнению с другими кластерами. В данном случае коэффициент составил 0,17, что указывает на слабое разделение данных.

Причины слабой кластеризации:

* Однородность данных: средние значения признаков для кластеров очень схожи, что может говорить о том, что в данных нет четко выраженных различий, которые можно было бы использовать для классификации клиентов.
* Невозможность выделить четкие группы: если различия между группами минимальны, то алгоритм может не выявить значимые паттерны, что приводит к слабой кластеризации.
* Недостаточная информативность признаков: может быть, текущие признаки не дают достаточного контекста для разделения клиентов. Например, без учета поведенческих характеристик или других данных, такие как предпочтения клиентов, сложно выделить сегменты.
* Перекрытие кластеров: визуально или через метрики можно увидеть, что кластеры перекрываются, а объекты разных кластеров находятся слишком близко друг к другу.

Рекомендации по улучшению качества кластеризации:

* Добавление новых признаков. Возможно, текущие признаки недостаточно отражают различия между клиентами. Включение дополнительных данных (например, демографические характеристики, история покупок) может улучшить результаты.
* Использование других методов кластеризации. Алгоритм K-means может не идеально подходить для данной задачи, так как он предполагает, что кластеры имеют сферическую форму и одинаковые размеры. Использование алгоритма DBSCAN или Agglomerative Clustering может лучше работать с неравномерными данными и выявлять более сложные структуры.

в) Задача регрессии

Целью задачи являлось прогнозирование суммы покупки клиента на основе различных факторов, таких как количество товара, цена, скидки и категория. Для решения задачи выбрана линейная регрессия.

Модель линейной регрессии была обучена на данных, и были получены следующие коэффициенты для признаков:

* Quantity (27,86) – увеличение количества товара на 1 единицу увеличивает сумму покупки на 27,86 единиц.
* UnitPrice (11,68) – увеличение цены на 1 единицу увеличивает сумму покупки на 11,68 единиц.
* Discount (-54,10) – увеличение скидки на 1 единицу уменьшает сумму покупки на 54,10 единиц.
* Категории товаров (Apparel, Electronics, Accessories и т.д.) – имеют разные влияния на итоговую сумму (например, Electronics влияет на увеличение суммы покупки, а Stationery снижает).
* Страны (Australia, Germany, UK и т.д.) – влияние разных стран на сумму покупки варьируется, например, заказы из Германии увеличивают сумму покупки, а из Австралии – уменьшают.

Месяцы (Jan, Feb, Mar и т.д.) – влияние времени года на покупку также варьируется, например, в декабре сумма покупки снижается.

Дни недели (Mon, Tue, Wed и т.д.) – наибольшее влияние на сумму покупки оказывают вторник и среда, а наименьшее – понедельник и воскресенье.

Для оценки качества модели был использован коэффициент детерминации (R2). R2 = 0,6 – это значит, что модель объясняет 60% вариации целевой переменной (суммы покупки). Результат говорит о хорошем качестве модели, но есть возможности для ее улучшения.

Для оценки значимости каждой переменной в модели был рассчитан уровень значимости p:

* Существенные переменные (p < 0,05): Quantity, UnitPrice, Discount, Apparel, Electronics, Accessories, Stationery и др.
* Несущие минимальное влияние (p > 0,05): SalesChannel, ReturnStatus и некоторые другие переменные.

На основе уровня значимости p можно сделать выводы о значимости факторов, влияющих на итоговую сумму покупки.

Данные, которые мы получили после анализа и обучения модели линейной регрессии, можно использовать для множества целей:

* Прогнозирование будущих покупок: используя обученную модель, можно предсказать, сколько клиент потратит в будущем, основываясь на текущем количестве товара, цене, скидке и других факторах. Это полезно для планирования запасов товаров, оценки продаж на разных рынках, прогнозирования выручки на определённые периоды времени.
* Оптимизация ценообразования: зная, что цена влияет на сумму покупки, можно оптимизировать стратегию ценообразования. Например, если увеличение цены на товар на 1 единицу увеличивает общую сумму покупки на 11,68, это может дать информацию о том, насколько можно поднять цену без потери прибыли.
* Измерение влияния скидок и акций: можно анализировать, как скидки и маркетинговые акции влияют на покупки в различных странах или категориях товаров. Например, если скидки оказывают сильное отрицательное влияние на общую сумму покупки, можно тестировать различные типы скидок или акций.
* Персонализированные предложения: на основе того, как различные факторы (категория товара, страна, месяц) влияют на покупки, можно создавать более персонализированные маркетинговые кампании. Например, можно запускать специальные предложения в определённые месяцы (например, в декабре скидки могут уменьшать сумму покупки, что нужно учитывать).
* Оценка спроса по регионам: если модель показывает, что клиенты из Германии покупают больше, чем из Австралии, это может помочь скорректировать стратегию распределения товаров между странами и регионами.
* Машинное управление скидками: модели можно интегрировать в системы, которые автоматически подстраивают размер скидок и цен в зависимости от множества факторов, таких как категория товара, страна или сезон.
* Принятие стратегических решений: руководство может использовать выводы, сделанные из данных, для корректировки долгосрочной стратегии компании. Например, если регрессия показала, что продажи электроники высоки, это может повлиять на решение увеличить инвестиции в этот сегмент.

# Заключение

В ходе выполнения курсовой работы были решены несколько математических задач, связанных с анализом данных о покупках в магазине, с применением методов машинного обучения: классификация методом K ближайших соседей, классификация методом K-Means и регрессия методом линейной регрессии.

Модель регрессии показала удовлетворительные результаты. Предсказания достаточно точные, что свидетельствует о правильной настройке модели и выбранных признаках. В целом, эта модель работает стабильно, и результаты соответствуют ожиданиям.

Модель классификации не справилась с разделением данных на классы. Точность предсказаний низкая, возможно, из-за неподобающих признаков или недостаточного обучения модели. Требуется улучшение модели, возможно, через добавление более релевантных признаков.

Кластеризация не дала осмысленных и четких групп. Выделенные кластеры выглядят случайными, и нет четкого разделения объектов. Это может быть связано с неправильным выбором числа кластеров или алгоритма.

# Список литературы

1. Лимановская, О. В. Основы машинного обучения : учебное пособие / О. В. Лимановская, Т. И. Алферьева. – Екатеринбург : Изд-во Урал. Ун-та, 2020 – 88 с.
2. Методика и организация самостоятельной работы студентов: учебно-методическое пособие / Е.В. Ершов, Л.Н. Виноградова, В.В. Селивановских [и др.]. – Череповец: ФГБОУ ВПО ЧГУ, 2015. – 243 с.
3. Алгоритмы KNN. Быстрый поиск ближайших соседей – Яндекс.Образование [электр.ресурс] https://education.yandex.ru/handbook/ml/article/metricheskiye-metody. Дата обращения: 22.12.2024.
4. Библиотека Matplotlib для построения графиков – Skillbox [электр.ресурс] https://skillbox.ru/media/code/biblioteka-matplotlib-dlya-postroeniya-grafikov/. Дата обращения: 22.12.2024.
5. Введение в Feature Engineering для начинающих дата-сайентистов и ML-инженеров – Хабр [электр.ресурс] https://habr.com/ru/companies/skillfactory/articles/842444/. Дата обращения: 22.12.2024.
6. Визуализация данных: что лежит в основе, где начинается творчество – Medium [электр.ресурс] https://medium.com/arg-design/визуализация-данных-что-лежит-в-основе-где-начинается-творчество-b23379a7ae82. Дата обращения: 22.12.2024.
7. Градиентный бустинг – Яндекс.Образование [электр.ресурс] https://education.yandex.ru/handbook/ml/article/gradientnyj-busting. Дата обращения: 22.12.2024.
8. Евклидова метрика – Википедия [электр.ресурс] https://ru.wikipedia.org/wiki/Евклидова\_метрика. Дата обращения: 22.12.2024.
9. Иерархическая кластеризация: основы и примеры – SkyPro [электр.ресурс] https://sky.pro/wiki/python/ierarhicheskaya-klasterizaciya-osnovy-i-primery/. Дата обращения: 22.12.2024.
10. Как создать свою первую модель машинного обучения на Python – Хабр [электр.ресурс] https://habr.com/ru/articles/759934/. Дата обращения: 22.12.2024.
11. Как строить красивые графики на Python с Seaborn – Хабр [электр.ресурс] https://habr.com/ru/companies/otus/articles/540526/. Дата обращения: 22.12.2024.
12. Классификация в машинном обучении – SkyPro [электр.ресурс] https://sky.pro/wiki/python/klassifikaciya-v-mashinnom-obuchenii/. Дата обращения: 22.12.2024.
13. Кластеризация в ML: от теоретических основ популярных алгоритмов к их реализации с нуля на Python – Хабр [электр.ресурс] https://habr.com/ru/articles/798331/. Дата обращения: 22.12.2024.
14. Логистическая регрессия: что это простыми словами – SkillFactory [электр.ресурс] https://blog.skillfactory.ru/glossary/logisticheskaya-regressiya/. Дата обращения: 22.12.2024.
15. Машинное обучение: Кластеризация методом K-means. Теория и реализация. С нуля – Хабр [электр.ресурс] https://habr.com/ru/articles/868542/. Дата обращения: 22.12.2024.
16. Метод случайного леса: основы и примеры – SkyPro [электр.ресурс] https://sky.pro/wiki/python/metod-sluchajnogo-lesa-osnovy-i-primery/. Дата обращения: 22.12.2024.
17. Основы линейной регрессии – Хабр [электр.ресурс] https://habr.com/ru/articles/514818/. Дата обращения: 22.12.2024.
18. Переменные в научных исследованиях – Научные переводы [электр.ресурс] https://научныепереводы.рф/peremennye-v-issledovaniyah/. Дата обращения: 22.12.2024.
19. Расстояние городских кварталов – Википедия [электр.ресурс] https://ru.wikipedia.org/wiki/Расстояние\_городских\_кварталов. Дата обращения: 22.12.2024.
20. Руководство к использованию деревьев решений в машинном обучении и науке о данных – Medium [электр.ресурс] https://medium.com/nuances-of-programming/руководство-к-использованию-деревьев-решений-в-машинном-обучении-и-науке-о-данных-c10030f05349. Дата обращения: 22.12.2024.
21. DBSCAN: What is it? When to Use it? How to use it – Medium [электр.ресурс] https://elutins.medium.com/dbscan-what-is-it-when-to-use-it-how-to-use-it-8bd506293818. Дата обращения: 22.12.2024.
22. Introduction to Keras – Deep Learning Library – Medium [электр.ресурс] https://medium.com/mlait/introduction-to-keras-deep-learning-library-2844b39f0496. Дата обращения: 22.12.2024.
23. Label Encoding. Most of the data frames contains either – Medium [электр.ресурс] https://maheswararedypr.medium.com/label-encoding-21484ca4f270. Дата обращения: 22.12.2024.
24. One Hot Encoding in Machine Learning – GeeksForGeeks [электр.ресурс] https://www.geeksforgeeks.org/ml-one-hot-encoding/. Дата обращения: 22.12.2024.
25. Python – основные понятия и возможности языка. Гайд для начинающих – Medium [электр.ресурс] https://danyefimoff.medium.com/python-основные-понятия-и-возможности-языка-гайд-для-начинающих-5139116262bf. Дата обращения: 22.12.2024.
26. Python Pandas Guide - Learn Pandas For Data Analysis – Medium [электр.ресурс] https://medium.com/edureka/python-pandas-tutorial-c5055c61d12e. Дата обращения: 22.12.2024.
27. PyTorch — ваш новый фреймворк глубокого обучения – Хабр [электр.ресурс] https://habr.com/ru/articles/334380/. Дата обращения: 22.12.2024.
28. R-квадрат. Как понять, насколько точно линейная модель дала предсказание? – Medium [электр.ресурс] https://congyuzhou.medium.com/r-квадрат-ac59c154a462. Дата обращения: 22.12.2024.
29. Regression in machine learning – GeeksForGeeks [электр.ресурс] https://www.geeksforgeeks.org/regression-in-machine-learning/. Дата обращения: 22.12.2024.
30. Scikit Learn (Beginners) – Part 1 – Medium [электр.ресурс] https://deepanshu-gaur.medium.com/scikit-learn-part-1-introduction-fa05b19b76f1. Дата обращения: 22.12.2024.
31. Silhouette Index – Cluster Validity index | Set 2 – GeeksForGeeks [электр.ресурс] https://www.geeksforgeeks.org/silhouette-index-cluster-validity-index-set-2/. Дата обращения: 22.12.2024.
32. Synthetic Minority Oversampling Technique – Хабр [электр.ресурс] https://habr.com/ru/articles/842480/. Дата обращения: 22.12.2024.
33. TensorFlow Tutorial – A Comprehensive Guide To Deep Learning Using TensorFlow – Medium [электр.ресурс] https://medium.com/edureka/tensorflow-tutorial-ba142ae96bca. Дата обращения: 22.12.2024.
34. Understanding the P-Value in Regression – Medium [электр.ресурс] https://medium.com/analytics-vidhya/understanding-the-p-value-in-regression-1fc2cd2568af. Дата обращения: 22.12.2024.
35. What is NumPy in Python — Introduction to NumPy — NumPy Tutorial – Medium [электр.ресурс] https://medium.com/edureka/python-introduction-to-numpy-numpy-tutorial-4ac06c717971. Дата обращения: 22.12.2024.

# Приложение

МИНОБРАНАУКИ РОССИИ

федеральное государственное бюджетное

образовательное учреждение высшего образования

ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Институт информационных технологий

наименование института (факультета)

Математическое и программное обеспечение ЭВМ

наименование кафедры

Математические методы решения задач искусственного интеллекта

наименование дисциплины в соответствии с учебным планом

УТВЕРЖДАЮ

Зав. кафедрой МПО ЭВМ

д. т.н. \_\_\_\_\_\_\_\_\_ Ершов Е.В.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2024 г.

Программирование на языке низкого уровня

Техническое задание на курсовую работу

Листов 7

Руководитель: Юдина О. В.

Исполнитель: студент гр. 1ПИб-02-3оп-22

Маркелов С. А.

2023 г.

Введение

Курсовая работа направлена на решение задач с применением различных математических методов машинного обучения и искусственного интеллекта.

1. Основания для разработки

Основанием для разработки является задание на курсовую работу по дисциплине «Математические методы решения задач искусственного интеллекта», выданное на кафедре МПО ЭВМ ИИТ ЧГУ.

Дата утверждения: 12 ноября 2023 года.

Наименование темы разработки: «Математические методы в машинном обучении».

2. Назначение разработки

Основной задачей курсовой работы является применение математических методов искусственного интеллекта (классификация, кластеризация, регрессия) для решения задач по анализу данных.

3. Требования к моделям

* 1. Требования к функциональным характеристикам

К методам решения задач предъявляются следующие требования:

* полнота и правильность использования предоставленных данных;
* корректное применение инструментов анализа данных и их применение к набору данных;
* обоснованная зависимость между результатами исследования и качеством выводов, а также их предсказательной способности;
* верное исследование и толкование полученных результатов.
  1. Требования к надежности

Решения задач должны иметь высокие значения коэффициентов, оценивающих ее точность (силуэтный коэффициент, коэффициент R2 и т. д.).

* 1. Условия эксплуатации

Для корректной работы программы необходимо:

* наличие любого устройства, на котором возможно запускать файлы формата .py и .xlsx (компьютер, ноутбук);
* наличие на устройстве современной версии любой операционной системы.
  1. Требования к составу и параметрам технических средств

Для корректной работы программы необходимо:

* ОЗУ: не менее 1 Гб для 32-разрядной ОС, 2 Гб для 64-разрядной ОС;
* процессор не менее чем с 2 ядрами и тактовой частотой не ниже 1 ГГц;
* разрешение экрана не менее 800 х 600;
* видеокарта с видеопамятью не менее 1 Гб;
* наличие мыши и клавиатуры.
  1. Требования к информационной и программной совместимости

Для корректной работы программы необходимо:

* установленная операционная система (например, Windows XP, Vista, 7, 8, 8.1, 10, 11, macOS, Linux);
* установленная среда программирования «Visual Studio Code»;
* установленный редактор электронных таблиц «Microsoft Excel» из пакета офисных программ «Microsoft Office».
  1. Требования к маркировке и упаковке

Требования к маркировке и упаковке программы не предъявляются.

* 1. Требования к транспортированию и хранению

Требования к транспортированию и хранению не предъявляются.

* 1. Специальные требования

Специальные требования не предъявляются.

4. Требования к программной документации

* 1. Содержание расчетно-пояснительной записки

Программная документация должна содержать расчётно-пояснительную

записку с содержанием:

1. Титульный лист
2. Аннотация
3. Оглавление
4. Введение
5. Основная часть
   1. Описание данных
   2. Формирование новых признаков
   3. Выбор задач машинного обучения
   4. Задача классификации
   5. Задача кластеризации
   6. Задача регрессии
   7. Визуализация данных
   8. Алгоритмы решения задач
   9. Интерпретация и использование результатов
6. Заключение
7. Список литературы
8. Приложения
   1. Техническое задание
   2. Требования к оформлению

Текстовые документы оформляются на белых листах формата А4, графический материал допускается представлять на листах формата A3. В соответствии с общими требованиями поля листа определяются следующим образом: левое – 30 мм, правое – 10 мм, верхнее – 20 мм, нижнее – 20 мм. Формат текста: Word for Windows, через полтора интервала (около тридцати строк на листе), шрифт – Times New Roman Cyr, размер шрифта – 14, отступ первой строки абзаца – 0,75 см. Количество знаков в строке, считая пробелы, – 60. Текст программы может быть расположен в две колонки, шрифт – Times New Roman Cyr, размер шрифта – 8. Нумерация всех страниц (в том числе и приложений) сквозная. Номер проставляется в середине верхнего поля страницы арабской цифрой. Номера страниц на титульном листе, аннотации и оглавлении не проставляются.

Наименование разделов, подразделов, пунктов должно быть кратким и соответствовать содержанию. Каждая новая глава печатается с новой страницы. Это же правило относится и к другим основным структурным частям работы: аннотации, оглавлению, введению, заключению, списку литературы, приложениям. Наименование разделов (основных частей) пишется прописными буквами по центру строки. Расстояние между заголовками и текстом, а также между заголовками разделов и подразделов должно быть равно двум интервалам. Наименования подразделов и пунктов размещаются с абзацного отступа (0,75 см) и печатаются с прописной буквы, без подчеркивания и без точки в конце. Расстояние между последней строкой текста предыдущего раздела и последующим заголовком при расположении их на одной странице должно быть равно трем интервалам. Разделы и подразделы нумеруются арабскими цифрами с точкой. Разделы имеют порядковые номера 1, 2 и т.д. Номер подраздела состоит из номера раздела и порядкового номера подраздела, входящего в данный раздел, разделенных точкой (например: 1.1, 2.5). При использовании ссылок на пункты, разделы и подразделы указывается порядковый номер раздела или пункта (например: «в разд. 2», «в п. 2.3.1»). Перечисления нумеруются арабскими цифрами со скобкой (например: 2), 3) и т. д.) с абзацного отступа. Для этого используются нумерованные списки. Допускается также применение маркированных списков одного вида. Таблицы в основном применяются для оформления цифрового материала. Шрифт – Times New Roman Cyr, размер шрифта – 14. Иногда возможен 10-й размер шрифта. Обозначения единиц физических величин необходимо применять в системе СИ. Номер таблицы размещается в правом верхнем углу над заголовком.

5. Технико-экономические показатели

Требования не предъявляются.

6. Стадии и этапы разработки

В данном пункте описаны стадии и этапы разработки программы (табл. П1.1).

Таблица П1.1

Стадии и этапы разработки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапа разработки | Сроки разработки | Результат выполнения | Отметка о выполнении |
| Определение темы для курсовой работы | 12.11.2024 | Утверждена тема для разработки | Выполнено |
| Оформление технического задания | 18.11.2024 | Оформленное техническое задание | Выполнено |
| Изучение и описание данных | 26.11.2024 | Данные изучены и описаны | Выполнено |
| Формирование своих признаков | 02.12.2024 | Новые признаки | Выполнено |
| Решение задачи классификации | 05.12.2024 | Решенная задача классификации | Выполнено |
| Решение задачи кластеризации | 08.12.2024 | Решенная задача кластеризации | Выполнено |
| Решение задачи регрессии | 10.12.2024 | Решенная задача регрессия | Выполнено |
| Визуализация данных | 16.12.2024 | Конечный вариант программы | Выполнено |
| Интерпретация результатов | 18.12.2024 | Результаты проанализированы, сделаны выводы | Выполнено |
| Оформление сопроводительной документации | 19.12.2024 | Оформленная сопроводительная документация | Выполнено |

7. Порядок контроля и приемки

В данном пункте описан порядок контроля и приемки курсовой работы (табл. П1.3).

Таблица П1.2

Порядок контроля и приемки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование контрольного этапа выполнения курсовой работы | Сроки контроля | Результат выполнения | Отметка о приемке результата контрольного этапа |
| Сдача технического задания | 18.11.2024 | Согласованное техническое задания | Выполнено |
| Сдача расчетно-пояснительной записки | 19.12.2024 | Согласованная расчетно-пояснительная записка | Выполнено |
| Сдача курсовой работы | 23.12.2024 | Получение оценки за выполненную работу |  |