Содержание этой страницы

Посвящается моим родителям Владимиру и Ольге.

Введение в машинное обучение

Неформальное определение

В повседневной жизни все мы ежедневно сталкиваемся с принятием решений:

- при текущей дорожной ситуации надо ли ехать на метро или на машине?
- удалить ли письмо в электронной почте как спам или сохранить?
- ожидается ли дождь, и стоит ли брать с собой зонтик?
- стоит ли звонить человеку с определённым предложением или он от него, скорее всего, откажется?
- стоит ли докупить хлеба и молока или их хватит до конца недели?
- какое фокусное расстояние на фотоаппарате установить, чтобы лицо фотографируемого человека получилось чётким?

Аналогичные проблемы принятия решений решаются организациями в массовом порядке:

- сколько хлеба и молока закупить магазину, чтобы удовлетворить спрос до конца недели?
- как почтовому сервису автоматически разделять письма на полезные и спам?
- какими способами и по каким маршрутам отправлять грузы?
- какую погоду предсказать сервису прогноза погоды на оставшийся конец дня?
- как автоматически устанавливать фокусное расстояние на производимых фотоаппаратах?

При массовом и повторяющемся принятии решений целесообразно процесс принятия этих решений автоматизировать. Можно разработать явную систему правил этого процесса. Например, при определении важности письма мы можем смотреть на то, переписывались ли мы ранее с отправителем, принадлежит ли отправитель надежной и известной компании, включает ли текст письма определённые ключевые слова, которые нам заранее не интересны? В этом случае мы как бы явно программируем алгоритм принятия решений. Но проблема заключается в том, что

- Сложно разработать универсальный алгоритм, который бы подходил всем пользователям. Одних может не интересовать получение кредита или психологическая консультация, а для других это может оказаться актуальным.
- Сложно учесть всё многообразие ситуаций. Например, "бесплатная психологическая консультация" может быть сформулирована как "консультация психолога без оплаты", и изначальное правило уже перестанет действовать.

В подобных случаях полезно использовать **машинное обучение** (machine learning).

> Машинное обучение - это процесс, в результате которого компьютер по наблюдаемым данным обучается лучше решать заданную задачу. Метод решения задачи при этом ищется в широком классе функций, параметризованном вектором параметров, который и подбирается по наблюдаемым данным.

Вместо явного прописывания четкой системы правил принятия решений в идеологии машинного обучения эти правила подбираются автоматически по данным. Под компьютером при этом может пониматься любое вычислительное устройство, например смартфон или процессор робота. Рассмотрим более детальное определение:

> Машина учится на заданном **опыте** решать некоторую **задачу**, относительно некоторого **показателя качества**, если

показатель качества растет на задаче после получения опыта.

В нашем примере задача - это классификация писем на спам/не спам, показатель качества - доля верно классифицированных писем, а опыт - коллекция прошлых писем, которые до этого были вручную размечены по классам.

В другом примере задачей выступает предсказание времени в пути, отталкиваясь от текущего времени суток, дня недели, погоды и загрузки дорог, показатель качества - модуль отклонения предсказанного времени от фактического, а опытом - история предыдущих передвижений в известных условиях и с известным временем в пути.

Примеры задач

Приведём примеры популярных задач, решаемых с помощью машинного обучения:

- Предсказать, уйдёт ли клиент к конкурентам? (churn prediction)
- Является ли последовательность финансовых транзакций мошеннической? (fraud detection)
- Предсказание пробок и времени в пути при планировании маршрута (traffic prediction).
- Стоит ли показывать заданный товар покупателю в качестве рекомендации? (recommender systems)
- Рекомендовать ли человека в качестве друга в социальной сети?
- Является ли аккаунт в социальной сети ботом?
- Голосовой ассистент: распознавание речи, автоматический ответ на вопросы, генерация речевого ответа.
- Идентификация человека по лицу. Распознавание номера машины на камерах.
- Подсчёт и отслеживание людей по камерам видеонаблюдения (object tracking). Обнаружение неправомерных действий (activity recognition).
- Автоматическое управление машинами (self-driving cars): распознавание ситуации, планирование маршрута.
- Автоматическая торговля на бирже (algorithmic trading).
- Перевод с одного языка на другой (machine translation).
- Постановка медицинских диагнозов по жалобам пациента и результатам обследований.
- Рекомендация веб-страниц по поисковому запросу (information retrieval).
- Автоматическая оценка ожидаемой зарплаты кандидата по резюме.
- Игра компьютера в шахматы, управление игровыми персонажами.
- Автоматическая оценка квартиры по её характеристикам.
- Хвалит или ругает пользователь товар в своём отзыве? (sentiment analysis)
- Генерация иллюстраций к тексту. Текстовое описание, что показано на изображении.
- Прогноз погоды. Рекомендации фермерам, когда сажать/поливать/удобрять посевы.
- Автоматическое написание программного кода (no code AI).
- Автоматический выбор, каким пользователям какую онлайн-рекламу показать (targeted ads).
- Генерация химических соединений, обладающих требуемыми свойствами:
    - крепкий, но легкий и термостойкий материал с повышенной проводимостью (material design)
    - препарат, обеспечивающий лечение и обладающий минимальными побочными эффектами (drug discovery)

Типы обучения

Машинное обучение (machine learning) описывает в целом подходы про подготовку данных, настройку и оценку прогнозирующих алгоритмов. Этому посвящена первая книга сайта, которую вы сейчас читаете.

Глубокое обучение (deep learning) - подраздел машинного обучения про сложные многоуровневые модели (нейросети), способные

решать более сложные задачи прогнозирования. С ростом вычислительных мощностей и объёма данных существует устойчивый тренд на замену классических алгоритмов машинного обучения на нейросетевые, обеспечивающие большую точность и возможность генерировать не только численные ответы, но и ответы в виде сложно структурированных данных, таких как текст, речь, изображение и видео. Глубокому обучению посвящена вторая часть книги.

Обучение с подкреплением (reinforcement learning) - также подраздел машинного обучения, в котором строится не однократный прогноз независимо для каждого объекта, а вырабатывается интерактивная стратегия поведения в изменяемой среде.

Примером обучения с подкреплением может служить автоматическая игра в шахматы, в которой необходимо последовательно генерировать каждый следующий ход. Успех генерации определяется не только текущим ходом, но и всей последовательностью решений в течение партии. Обучение с подкреплением также применяется в управлении игровыми персонажами в играх, машинами-роботами на дорогах, дронами, продвинутыми чат-ботами и роботизированными ассистентами.

Структура книги

Учебник посвящён классическому машинному обучению. В первой части изучаются основные постановки задач машинного обучения и фундаментальные понятия, необходимые для настройки моделей. Во второй части рассказывается про подготовку данных перед их использованием прогнозирующими моделями. В третьей части представлен общий вид классификаторов, необходимый для понимания их работы. Четвёртая часть посвящена метрическим методам регрессии и классификации, которые строят прогнозы, отталкиваясь от расстояний между изучаемыми объектами. Пятая часть знакомит читателя с линейной регрессией вместе с её всевозможными обобщениями и усложнениями. Оценивание качества прогнозов в задаче регрессии представлено в шестой части. В седьмой части даётся определение линейных классификаторов в общем виде, а также рассказывается про популярные методы этого класса - метод опорных векторов и логистическую регрессию. Построению многоклассовых классификаторов из набора бинарных посвящена восьмая часть. В девятой части описываются основные методы градиентной оптимизации, применяемые в классическом машинном обучении. В десятой части описаны методы оценки точности работы классификаторов. Одиннадцатая часть посвящена решающим деревьям. В двенадцатой части читатель познакомится с понятием переобученных и недообученных моделей на примере разложения на смещение и разброс. В тринадцатой части описывается принцип построения прогнозов, используя не одну модель, а сразу несколько, и приводятся описания популярных методов построения композиций моделей, включая усреднение, голосование, бэггинг, стэкинг и другие. Четырнадцатая часть описывает алгоритм бустинга - самого популярного и успешного метода построения композиций моделей. Пятнадцатая и шестнадцатая части посвящены различным подходам к интерпретации простых и более сложных моделей машинного обучения.

Учебник не затрагивает тему использования многослойных нейросетей (глубокого обучения). Этой теме посвящён второй учебник сайта.

Примеры кода для запуска методов

Многие методы, описанные в учебнике, сопровождаются примерами их запуска на языке python с использованием библиотек `sklearn`, `numpy` и `matplotlib`. Для этих библиотек использовались версии 1.3.0, 1.26.0 и 3.8.4 соответственно. Для анализа данных и тестирования различных методов машинного обучения удобно использовать бесплатную среду разработки jupyterlab. Для удобства установки рекомендуется использовать менеджер пакетов anaconda. В приводимых примерах для генерации данных используются следующие функции:

```
import numpy as np
from sklearn.datasets import make_moons

def get_demo_classification_data():
    X,Y = make_moons(n_samples=3000, noise=0.3 ,random_state=0)    # генерируем данные для классификации
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.4, stratify=Y, random_state=0)    # раз
    return X_train, X_test, Y_train, Y_test

def get_demo_regression_data():
    np.random.seed(0)
    X = np.random.normal(size=[3000,5])
    NOISE = 0.3*np.random.normal(size=[3000])
    Y = X.mean(axis=1)+(X**2).mean(axis=1)+NOISE     # генерируем данные для регрессии
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.4, random_state=0)    # разбиваем выбор
    return X_train, X_test, Y_train, Y_test
```

Примеры запуска методов будут идти после описания самих методов, но вы также можете посмотреть код сразу всех примеров по ссылке с результатами его работы.

Предыдущая страница
« Машинное обучение

Следующая страница
Основы машинного обучения »