МАШИННОЕ ОБУЧЕНИЕ И НЕЙРОННЫЕ СЕТИ

Глава 1. Введение

Машинное обучение играет важную роль в современном бизнесе и исследованиях. Оно использует различные алгоритмы, включая искусственные нейронные сети, для помощи компьютерным системам в постепенном улучшении их эффективности. Алгоритмы машинного обучения автоматически строят математические модели, используя имеющиеся данные, чтобы принимать решения, не будучи явно запрограммированными для принятия этих решений.

1.1. Определение

Впервые термин машинного обучения был введен в обиход пионером искусственного интеллекта, изобретателем самообучающейся программы для игры в шашки Артуром Сэмюэлом в 1959 году:

Определение 1.1. *Машинное обучение это область исследований, которая даёт компьютеру возможность учиться, не будучи явно запрограммированным.*

В одной из первых книг по машинному обучению "Machine Learning", написанной Томом Митчеллом в 1997, даётся следующее определение:

Определение 1.2. *Говорят, что компьютерная программа учится из опыта $\mathcal{E}$ по отношению к какому-то классу задач $\mathcal{T}$ и критерию качества $\mathcal{P}$, если её качество на задачах из $\mathcal{T}$, измеренное с помощью $\mathcal{P}$, улучшается с использованием опыта $\mathcal{E}$.*

Глава 2. Классификация алгоритмов

В зависимости от постановки задачи используются различные алгоритмы:

- Обучение с учителем.
- Обучение без учителя.
- Частичное обучение.
- Обучение с подкреплением.
- Динамическое обучение.
- Активное обучение.
- Метаобчение.

2.1. Обучение с учителем

Рассматривается решение следующей задачи. Имеется множество объектов (ситуаций) и множество возможных ответов (откликов, реакций). Существует некоторая зависимость между ответами и объектами, но она неизвестна. Известна только конечная совокупность прецедентов — пар (объект, ответ), называемая обучающей выборкой. На основе этих данных требуется восстановить зависимость, то есть построить алгоритм, способный для любого объекта выдать достаточно точный ответ. Для измерения точности ответов определённым образом вводится функционал качества.

Под учителем понимается либо сама обучающая выборка, либо тот, кто указал на заданных объектах правильные ответы.

Типы входных данных

- Признаковое описание или матрица объекты-признаки — наиболее распространённый случай. Каждый объект описывается набором своих характеристик, называемых признаками. Признаки могут быть числовыми или нечисловыми.
- Матрица расстояний между объектами. Каждый объект описывается расстояниями до всех остальных объектов обучающей выборки. С этим типом входных данных работают немногие методы, в частности, метод ближайших соседей, метод парзеновского окна, метод потенциальных функций.

- Временной ряд или сигнал представляет собой последовательность измерений во времени. Каждое измерение может представляться числом, вектором, а в общем случае — признаковым описанием исследуемого объекта в данный момент времени.
- Изображение или видеоряд.

Типы откликов

- Задачи классификации — множество возможных ответов конечно. Их называют идентификаторами (именами, метками) классов.
- Задачи регрессии — ответы являются действительными числами или векторами.

Формальная постановка

Пусть X — множество описаний объектов, Y — множество допустимых ответов. Существует неизвестная целевая зависимость — отображение $y^* : X \to Y$, значения которой известны только на объектах конечной обучающей выборки $X^m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$. Требуется построить алгоритм $a : X \to Y$, который приближал бы неизвестную целевую зависимость как на элементах выборки, так и на всём множестве X.

Говорят также, что алгоритм должен обладать способностью к обобщению эмпирических фактов, или выводить общее знание (закономерность, зависимость) из частных фактов (наблюдений, прецедентов).

Данная постановка является обобщением классических задач аппроксимации функций. В классической аппроксимации объектами являются действительные числа или векторы. В реальных прикладных задачах входные данные об объектах могут быть неполными, неточными, неоднородными, нечисловыми. Эти особенности приводят к большому разнообразию методов обучения с учителем.

2.2. Обучение без учителя

Рассматривается решение следующей задачи. Известны только описания множества объектов (обучающей выборки), и требуется обнаружить внутренние взаимосвязи, зависимости, закономерности, существующие между объектами.

Типы входных данных

- Признаковое описание объектов. Каждый объект описывается набором своих характеристик, называемых признаками. Признаки могут быть числовыми или нечисловыми.
- Матрица расстояний между объектами. Каждый объект описывается расстояниями до всех остальных объектов обучающей выборки.

Типы задач

- Кластеризация. Выборка объектов разбивается на непересекающиеся подмножества, называемые кластерами, так, чтобы каждый кластер состоял из схожих объектов, а объекты разных кластеров существенно отличались. Исходная информация представляется в виде матрицы расстояний.
- Поиск правил ассоциации. Исходная информация представляется в виде признаковых описаний. Задача состоит в том, чтобы найти такие наборы признаков, и такие значения этих признаков, которые особенно часто (неслучайно часто) встречаются в признаковых описаниях объектов.
- Заполнение пропущенных значений. Исходная информация представляется в виде признаковых описаний. Значения некоторых признаков для некоторых объектов могут отсутствовать. Необходимо восстановить пропущенные значения признаков.
- Сокращение размерности. Исходная информация представляется в виде признаковых описаний, причём число признаков может быть достаточно большим. Задача состоит в том, чтобы представить эти данные в пространстве меньшей размерности, по возможности, минимизировав потери информации.
- Визуализация данных. Некоторые методы кластеризации и снижения размерности строят представления выборки в пространстве размерности два. Это позволяет отображать многомерные данные в виде плоских графиков и анализировать их визуально, что способствует лучшему пониманию данных и самой сути решаемой задачи.

2.3. Частичное обучение

— один из методов машинного обучения, использующий при обучении как размеченные, так и неразмеченные данные. Обычно используется небольшое

количество размеченных и значительный объём неразмеченных данных. Частичное обучение является компромиссом между обучением без учителя (без каких-либо размеченных обучающих данных) и обучением с учителем (с полностью размеченным набором обучения). Было замечено, что неразмеченные данные, будучи использованными совместно с небольшим количеством размеченных данных, могут обеспечить значительный прирост качества обучения. Под качеством обучения подразумевается некий функционал качества, например, среднеквадратичная ошибка. Сбор размеченных данных для задачи обучения зачастую требует, чтобы квалифицированный эксперт вручную классифицировал объекты обучения. Затраты, связанные с процессом разметки, могут сделать построение полностью размеченного набора прецедентов невозможным, в то время как сбор неразмеченных данных сравнительно недорог. В подобных ситуациях ценность частичного обучения сложно переоценить.

Примером частичного обучения может послужить сообучение: два или более обучаемых алгоритма используют один и тот же набор данных, но каждый при обучении использует различные — в идеале некоррелированные — наборы признаков объектов.

Альтернативный подход заключается в моделировании совместного распределения признаков и меток. В таком случае для неразмеченых данных метки могут трактоваться как пропущенные данные. Для построения оценки максимального правдоподобия обычно используется EM-алгоритм.

2.4. Обучение с подкреплением

Обучение с подкреплением, идея которого была почерпнута в смежной области психологии, является подразделом машинного обучения, изучающим, как агент должен действовать в окружении, чтобы максимизировать некоторый долговременный выигрыш. Алгоритмы с частичным обучением пытаются найти стратегию, приписывающую состояниям окружающей среды действия, которые должен предпринять агент в этих состояниях. В экономике и теории игр обучение с подкреплением рассматривается в качестве интерпретации того, как может установиться равновесие.

Окружение обычно формулируется как марковский процесс принятия решений (МППР) с конечным множеством состояний, и в этом смысле алгоритмы обучения с подкреплением тесно связаны с динамическим программированием. Вероятности выигрышей и перехода состояний в МППР обычно являются вели-

чинами случайными, но стационарными в рамках задачи.

При обучении с подкреплением, в отличии от обучения с учителем, не предоставляются верные пары „входные данные-ответ“, а принятие субоптимальных решений (дающих локальный экстремум) не ограничивается явно. Обучение с подкреплением пытается найти компромисс между исследованием неизученных областей и применением имеющихся знаний. Баланс изучения-применения при обучении с подкреплением исследовался в задаче многорукого бандита.

Формально простейшая модель обучения с подкреплением состоит из:

1. множества состояний окружения S;
2. множества действий A;
3. множества веществennознaчных скалярных “выигрыше”.

В произвольный момент времени t агент характеризуется состоянием $s_t \in S$ и множеством возможных действий $A(s_t)$. Выбирая действие $a \in A(s_t)$, он переходит в состояние s_{t+1} и получает выигрыш r_t. Основываясь на таком взаимодействии с окружающей средой, агент, обучающийся с подкреплением, должен выработать стратегию $\pi : S \to A$, которая максимизирует величину$R = r_0 + r_1 + \cdots + r_n$ в случае МППР, имеющего терминальное состояние, или величину $R = \sum_t \gamma^t r_t$ для МППР без терминальных состояний (где $0 \leq \gamma \leq 1$ — дисконтирующий множитель для “предстоящего выигрыш”).

Таким образом, обучение с подкреплением особенно хорошо подходит для решения задач, связанных с выбором между долгосрочной и краткосрочной выгодой. Оно успешно применялось в различных областях, таких как робототехника, управление лифтами, телекоммуникации, шашки и нарды.

Глава 3. Искусственные нейронные сети

Искусственная нейронная сеть (artificial neural network, ANN), или просто нейронная сеть — это математическая модель, а также ее программные или аппаратные реализации, построенная в некотором смысле по образу и подобию сетей нервных клеток живого организма.

Нейронные сети — один из наиболее известных и старых методов машинного обучения.

3.1. Суть метода

В общем виде нейронная сеть представляет из себя композицию дифференцируемых функций. Пользуясь правилом дифференцирования сложной функции, можно использовать градиентные методы для того, чтобы оптимизировать нейронную сеть под заданную выборку тренировочных данных. Процесс обновления весов нейронной сети называется обратным распространением ошибки. На практике нейронные сети могут содержать десятки миллионов параметров, а в обучающей выборке могут быть миллионы объектов, из-за чего подсчёт градиента нейронной сети по всей выборке будет слишком сложным. В этом случае используются стохастические методы, такие как RMSprop (Root Mean Square Propagation) и Adam.

Алгоритм 1 Алгоритм стохастической оптимизации Adam.

α: Размера шага.
$\beta_1, \beta_2 \in [0, 1)$: Коэффициенты экспоненциального затухания для оценок моментов.
$f(\theta)$: Стохастическая целевая функция с параметрами θ.
θ_0: Начальный вектор параметров.
$m_0 \leftarrow 0$ (Инициализация вектора первого момента).
$v_0 \leftarrow 0$ (Инициализация вектора второго момента).
$t \leftarrow 0$ (Инициализация номера итерации).

while параметры θ_t не сошлись **do**
  $t \leftarrow t + 1$
  $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$
  $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$
  $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
  $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$
  $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$
  $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$
end while

3.2. Нейрон

Основополагающим для нейронных сетей является понятие нейрона:

$$y = f(\sum_i w_i x_i + b), \tag{1}$$

Здесь $w_i, x_i, b \in \mathbb{R}, f : \mathbb{R} \to \mathbb{R}$. x_i – это входные данные нейрона, w_i – веса нейрона, b – смещение нейрона, f – функция активации нейрона, y – выход нейрона. При выборе различных функций активации один нейрон может действовать как линейный классификатор. Наиболее часто используемые функции активации:

- Сигмоида: $\sigma(x) = 1/(1 + e^{-x})$
  Данная функция переводит действительные числа в интервал $(0, 1)$. В частности, большие отрицательные числа становятся близки к 0, а большие положительные числа – к 1. Исторически сигмоида часто использовалась в качестве функции активации, так как её значение может быть проинтерпретировано как частота активации нейрона: 0 соответствует выключенному состоянию, а 1 соответствует полностью насыщенному. На

практике сигмоида используется всё реже из-за двух своих недостатков:

1) *Сигмоида насыщается и убивает градиенты.* Нежелательно свойство сигмоиды иметь близкие к нулю градиенты в области насыщения около 0 и 1. Таким образом, если локальный градиент очень маленький, то он *убьёт* градиент, и почти никакой сигнал не сможет пройти через нейрон к его весам и далее. Также нужно аккуратно инициализировать веса нейрона, иначе большие значения могут сильно замедлить обучение сети.
2) *Выход сигмоиды не центрирован относительно нуля.* Это может сказаться на динамике градиентного спуска, так как если входные данные нейрона всегда положительны, то все элементы градиента весов w во время обратного распространения ошибки будут иметь один и тот же знак. Это может привести к нежелательному зигзагообразному поведению градиента.

- ReLU: $max(0, x)$
  Данная функция имеет как свои плюсы:

1) Экспериментально было получено, что ReLU значительно ускоряет сходимость стохастического градиентного спуска по сравнению с сигмоидой и гиперболическим тангенсом [**?**]. Предположительно это связано с его линейной не насыщаемой формой.
2) По сравнению с сигмоидой и гиперболическим тангенсом, которые требуют сложных вычислений (экспонента), ReLU может быть реализовано как простая пороговая функция.

так и минусы:

1) К сожалению ReLU может быть неустойчива во время обучения и может *умереть*. Например, очень большой градиент проходящий через нейрон может изменить веса так, что нейрон больше никогда не активируется. Если это случится, то через этот нейрон перестанет распространяться градиент ошибки.
2) Формально ReLU не дифференцируема в нуле. Обычно этим можно пренебречь в силу того, что при использовании при вычислениях

чисел с плавающей запятой сложно получить точный ноль. При программной реализации значение производной в нуле зачастую полагают равной либо 0, либо 1.

Нейронная сеть представляет из себя набор нейронов объединенных в граф без циклов. Зачастую нейроны представлены в виде последовательных слоев, соединенных друг с другом. При подсчете количества слоев в сети принято не учитывать слой с входными данными.