Проект

Искусственные нейронные сети в задаче классификации

М. Жарков

2018 год

Содержание

1. Обзор

1.1. Классификация

Классификация объектов — одна из стандартных задач машинного обучения. Её можно описать так: имеется множество объектов, которые каким-то образом разделены на классы. Задано конечное множество объектов, для которых известно, к каким классам они относятся. Это множество называется *обучающей выборкой*. Классовая принадлежность остальных объектов неизвестна. Требуется построить алгоритм, способный классифицировать произвольный объект (то есть указать к какому классу он относится) из исходного множества.

В машинном обучении задача классификации относится к разделу *обучения с учителем*. Существует также *обучение без учителя*, когда разделение объектов обучающей выборки на классы не задаётся, и требуется классифицировать объекты только на основе их сходства друг с другом. В этом случае принято говорить о *задачах класстеризации*.

Одним из самых простых типов классификации является *бинарная классификация*, когда различных классов всего два. Данный тип служит основой для решения более сложных задач.

1.2. Методы решения

Для решения задач классификации могут использоваться следующие методы:

- Байесовский классификатор;
- Решающие деревья;
- Логистическая регрессия;
- Искусственные нейронные сети.

Байесовский классификатор — тип алгоритмов классификации, основанный на теореме, утверждающей, что если плотности распределения каждого из классов известны, то искомый алгоритм можно выписать в явном аналитическом виде. Более того, этот алгоритм оптимален, то есть обладает минимальной вероятностью ошибок. На практике плотности распределения классов, как правило, не известны. Их приходится оценивать по обучающей выборке. В результате байесовский алгоритм перестаёт быть оптимальным, так как восстановить плотность по выборке можно только с некоторой погрешностью. В задаче бинарной классификации звуков восстановление плотности классов является плохо решаемой проблемой.

Решающие деревья — средство поддержки принятия решений, структура которого представляет собой *листья* и *ветки*. На ветках дерева записаны атрибу-

ты, от которых зависит целевая функция, в листьях записаны значения целевой функции, а в остальных узлах — атрибуты, по которым различаются случаи. Цель состоит в том, чтобы создать модель, которая предсказывает значение целевой переменной на основе нескольких переменных на входе.

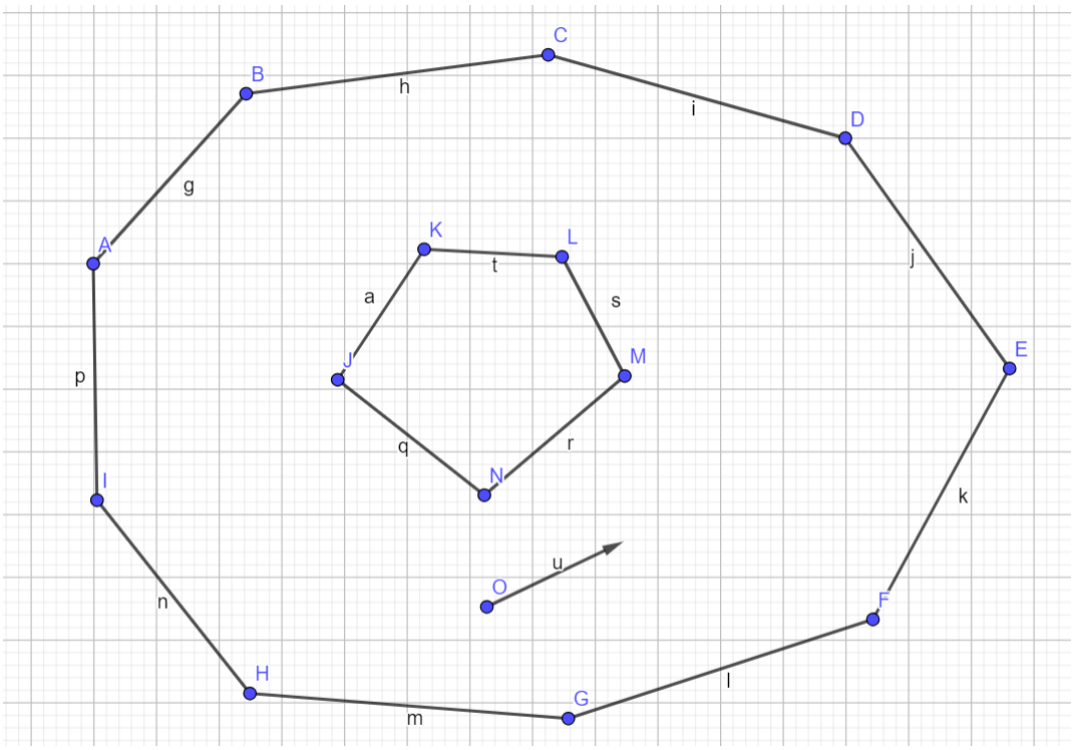
Одним из основных вопросов в реализации решающих деревьев для задачи классификации является выбор атрибутов, по которым будет осуществляться разделение данных на классы.

Логистическая регрессия — метод построения линейной разделяющей поверхности. В случае двух классов разделяющей поверхностью является гиперплоскость. В задаче бинарной классификации звуков нельзя гарантировать возможность разделения пространства параметров одной гиперплоскостью.

Искусственная нейронная сеть — это математическая модель, построенная в некотором смысле по образу и подобию сетей нервных клеток живого организма. Для решения задачи классификации может использоваться такой тип ИНС, как *многослойный перцептрон Розенблатта*. Он представляет собой передающую сеть, состоящую из генераторов сигнала трёх типов: сенсорных элементов, ассоциативных элементов и реагирующих элементов. Производящие функции этих элементов зависят от сигналов, возникающих либо где-то внутри передающей сети, либо, для внешних элементов, от сигналов, поступающих из внешней среды.

2. Постановка задачи

Есть некотарая трасса состоящая из двух наборов точек соединенных отрезками,при этом контуры замкнутые ивыпуклая(см рисунок).



Есть точка которая двигается по этой трассе с некоторой постоянной скоростью,которая может менять направления движения через равные промежутки времени.Перед тем как выбрать направление становится известно рассотяние до "стенок"в нескольких направлениях.Необходимо написать программу позволяющую проходить по трассе без "сталкновений"с отрезками как можно дольше.При этом желательно чтобы путь проходил как можно ближе к меньшему контору.

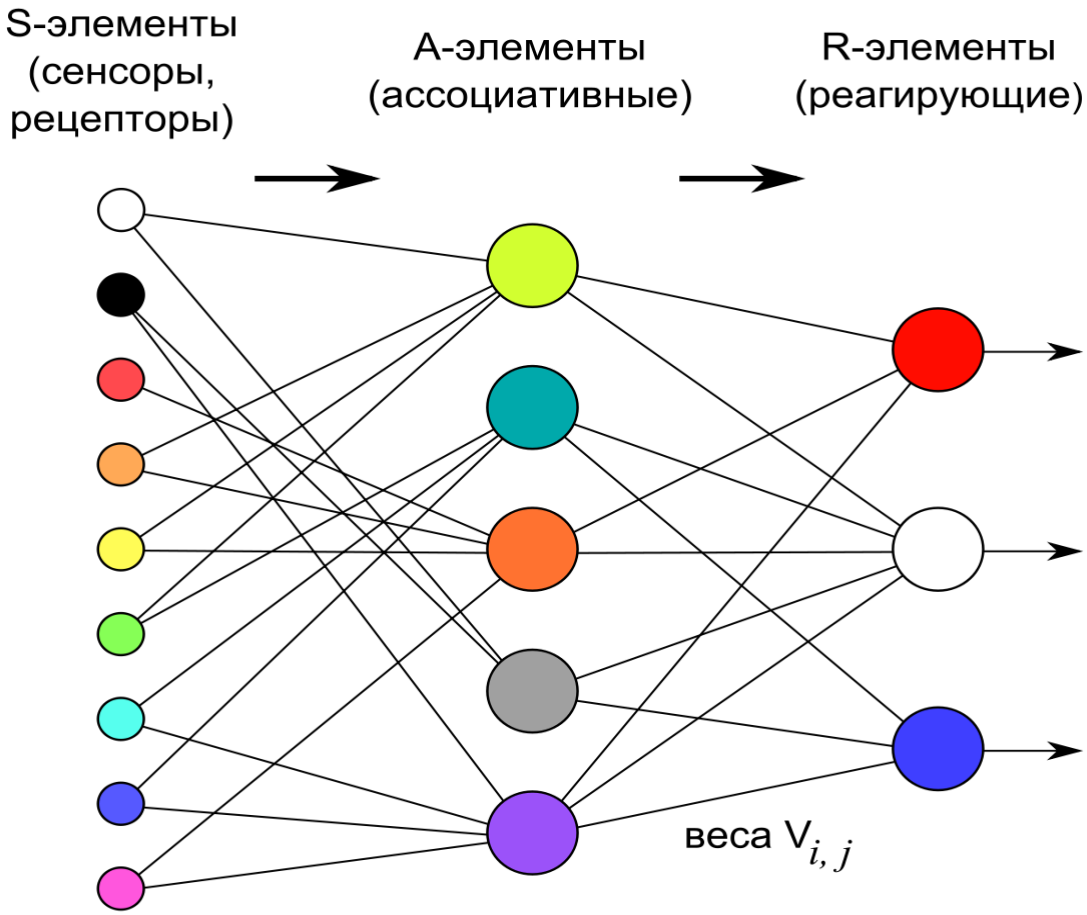
3. Решение задачи

3.1. Перцептрон

Перцептрон состоит из нескольких слоёв нейронов:

1. Входной слой, содержащий псевдо-нейроны, которые передают дальше значения *предикторов* — параметров объекта;
2. Один или несколько скрытых слоёв;
3. Выходной слой, содержащий один нейрон.

Передача сигналов (активация) нейронной сети происходит от входного слоя, через скрытые слои, к выходному слою.



Все нейроны (кроме входного слоя) имеют одинаковое строение, состоят из двух частей — сумматорной и активационной функций. Сумматорная функция определяет то, как нейрон будет использовать входящую информацию из предыдущего слоя. Активационная функция определяет реакцию нейрона, которая будет передана по всем выходным связям в следующий слой.

В качестве сумматорной функции выбрана взвешенная сумма всех входящих

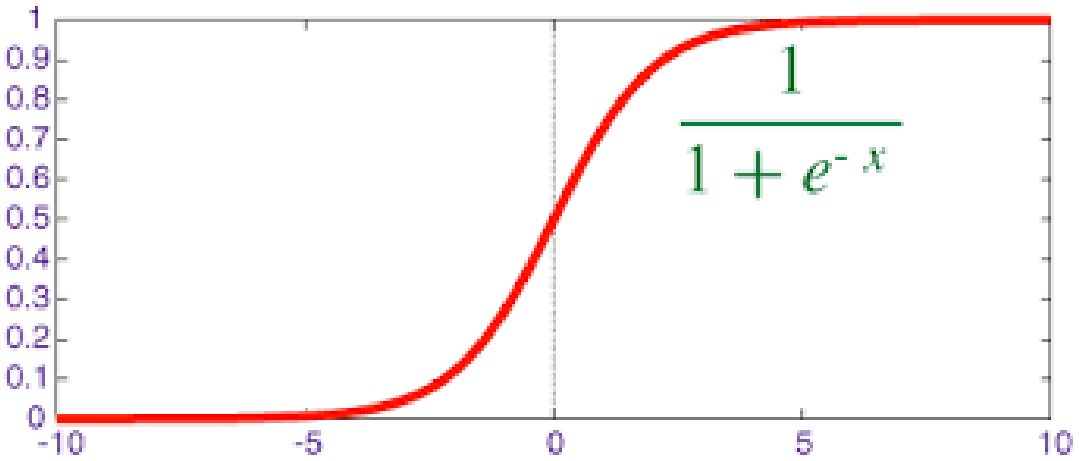
сигналов:

$$S = b + \sum_{j=1}^{m} x_j w_j$$

где m — количество входящих сигналов нейрона, x_j — значение, получаемое по j-ому входу, w_j — вес j-ого входа, b — некоторое смещение, изменяемое в процессе обучения. Смещение можно учитывать в сумме, если добавить в каждый слой, кроме выходного на первое место нейрон, у которого значение активации будет всегда равно 1.

Активационная функция — логистическая (сигмоидальная):

$$\sigma(S) = \frac{1}{1 + e^{-S}}$$



Логистическая функция является гладкой, что необходимо для работы алгоритма обучения. Кроме того, её значение можно интерпретировать как вероятность принадлежности объекта к одному из двух классов.

Обучение нейронной сети — это настройка весов для входящих связей всех нейронов, с целью получения достоверных предсказаний. Для обучения используется *алгоритм обратного распространеннния ошибки*, который основывается на градиентном спуске по простанству весов в сторону уменьшения значений целевой функции ошибки.

Для оценки правдоподобности предсказаний используется *квадратичная функция ошибки*:

$$E = \frac{1}{2N} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$$

где N — количество примеров, $\hat{y}_i$ — предсказанное значение для i-ого примера, y_i — правильный ответ для него.

Для того, чтобы понять, как изменится значение функции ошибки при изменении какого-либо веса входящих сигналов нейрона, нужно взять её частную производную по этому весу.

Сначала считается изменение весов в выходном слое:

$$\Delta w_j = -\alpha \frac{\partial E}{\partial w_j}$$

где $\alpha \in \mathbb{R}$ — скорость обучения.

В векторном виде:

$$\Delta W = -\alpha \nabla_W E$$

где $\nabla_W E = \left(\frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_m} \right)$ — градиент E в точке W.

Посчитаем частную производную от функции E по j-му весу:

$$\frac{\partial E}{\partial w_j} = \frac{\partial \left(\frac{1}{2N} \sum_{i=1}^{N} \left(\hat{y}_i - y_i \right)^2 \right)}{\partial w_j}$$

Так как производная суммы равна сумме производных, возьмём для простоты один пример, а после просуммируем все значения:

$$\frac{1}{2} \cdot \frac{\partial (\hat{y} - y)^2}{\partial w_j} = \frac{1}{2} \cdot \frac{\partial \left(\hat{y} - y \right)^2}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial w_j} = \left(\hat{y} - y \right) \frac{\partial \sigma \left(S \right)}{\partial w_j} =$$

$$= \left(\sigma \left(S \right) - y \right) \sigma\prime \left(S \right) \frac{\partial \sum_{j=1}^{m} x_j w_j}{\partial w_j} = \left(\sigma \left(S \right) - y \right) \sigma \left(S \right) \left(1 - \sigma \left(S \right) \right) x_j$$

Итак, общая формула для j-ого веса по N примерам:

$$\frac{\partial E}{\partial w_j} = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{y}_i - y_i \right) \hat{y}_i \left(1 - \hat{y}_i \right) x_j$$

Далее полученная ошибка распространяется по ИНС в обратном порядке, от выходного слоя ко входному, изменяя веса скрытых слоёв.

Введём следующие обозначения:

- w_{jk}^l — значение j-ого веса k-ого нейрона в l-ом слое (вес связи из j-ого нейрона $l-1$ слоя в k-ый нейрон l-ого слоя);
- m_l — количество нейронов в l-ом слое;
- s_k^l — значение сумматорной функции k-ого нейрона в l-ом слое;
- a_k^l — значение активационной функции k-ого нейрона в l-ом слое;
- $\delta_k^l = \frac{\partial E}{\partial s_k^l}$ — ошибка k-ого нейрона в l-ом слое.

Зная значение ошибки δ_k^l для каждого нейрона, можно получить соответствующее изменение его весов:

$$\Delta w_{jk}^l = -\alpha \delta_k^l a_j^{l-1}$$

Посчитаем значение ошибки для нейронов выходного (L-ого) слоя. Для простоты возьмём один пример:

$$\delta_k^L = \frac{\partial E}{\partial s_k^L} = \frac{1}{2} \cdot \frac{\partial \left(\sigma\left(s_k^L\right) - y_k\right)^2}{\partial s_k^L} = \left(a_k^L - y_k\right) a_k^L \left(1 - a_k^L\right)$$

где y_k — правильный ответ для k-ого нейрона выходного слоя.

Теперь выразим ошибку нейрона на l-ом слое через ошибки на $l+1$ слое:

$$\delta_j^l = \sigma\prime\left(s_j^l\right) \sum_{k=1}^{m_{l+1}} w_{jk}^{l+1} \delta_k^{l+1} = a_j^l \left(1 - a_j^l\right) \sum_{k=1}^{m_{l+1}} w_{jk}^{l+1} \delta_k^{l+1}$$

3.2. Предикторы

Для получения численных предикторов вычисляется рассояние от точки движения до прямых ограждающих трассу.В каждый момент поворота есть вектор предыдущего движения и по два вектра в разные стороны от него. Значения длин отрезков по этим пяти векторам и есть предикторы.

Естественноучитывается только расстояние до ближайшей стенки(если прямая ,заданная вектором, пересекает несколько "стенок то берется кратчайший путь).

4. Реализация

4.1. Подготовка предикторов

Для работы с картой используется несколько классов :

1. **Dot** - точка хранящая координаты по x и y.
2. **line** - это отрезки заданные уравнением (имеется функция проверки наличия точки на данном отрезке-**check()**).
3. **point** - движушаяся точка, у которой есть пять векторов направления.

Также имеется функция **cross** позволяющая при помощи простейших формул геометрии искать точки пересечения прямых,постоиных по одному из векторов **point** и отрезка. Третий параметор это номер вектора по которой нужно строить прямую.

В **main** созданы точки карты и по ним созданы массивы **lineS** и **lineB** ,хранящие нужные отезки контура.

Далее описан поиск расстояний от движушейся точки до отрезков при использовании выше описанных инструментов. При этом сначала ишутся длинна до внешнего конткра,а потом до внутреннего,ведь до последнего расстояние всегда меньше.

Получинные данные записываюся в массив и являются придикаторами.

4.2. Нейросеть

Ответом на вопрос куда нужно двигатся должен быть массив состоящий из 4-х нулей и 1-ой единицы. Так как только по одному направлению точко сможет двигаться дальше.

Для обучения нейросети потребуется верное решение.В данной задаче верным решение может являтся любой массив, который не приведет к столкновению с контуром. Но так как в задаче необходимо найти путь наиболее близкий к внутреннему контуру,то будем выбирать самый левый путь по которому можно ехать(движение осуществляется вправо по "треку".Оно записывается в массив **Id**.

Перцепртон описан в классе **perc** , у которого все веса входящие и выходящие записаны в массивах; также имеется фунция меняющая виса ; сумматорная функция ,ссумирующая все входящие значения умноженные на их веса.

Активационная функция представленна отдельной функцией.

В **mane** получается массив **distance[]** в качестве параметра вектор значений предикторов одного примера и проводит последовательную активацию нейронов по слоям от входного к выходному(значения активационных функций записы-

ваются в соответствующие массивы **column**), после чего возвращает значение активации нейрона в выходном слое в виде массива,наибольшее число которого указывает направление движения.

Далее реализуется алгоритм обратного распространения ошибки. Используется вектор значений предикторов и правильный ответ , а также активационнай функции нейронов и целевоа функции ошибки.

Сначала считается значение ошибки на выходном слое. Затем ошибка распространяется на предыдущие слои в обратном порядке. Важно что вес каждой связи нужно менять дважды,ведь у каждой связи есть исходный перцептрон и конечный.

Обучение нейросети происходит до тех пор пока движение становится невозможным, и эти "заезды"повтаряются необходимое число раз для получение нужных весов.

5. Использованные источники

1. machinelearning.ru — описание задачи классификации, перцептрона Розенблатта и других методов машинного обучения;
2. stepik.org — практический курс по нейронным сетям от Института биоинформатики;
3. — статьи на хабре по нейронным сетям.