Типы задач

Рассмотрим типы задач, которые встречаются в машинном обучении.

- *Классификация.*

Рассмотрим следующую задачу. Мы работаем в банке и нам необходимо, что система определяла, сможет вовремя ли человек погасить кредит или нет. У нас о нём есть различная информация, так называемые признаки (features). Они бывают различных типов:

- Бинарные признаки: наличие телефона.
- Номинальные признаки: профессия, адрес проживания.
- Порядковые признаки: образование, занимаемая доложность. По сути, они похожи на номинальные признаки, только тут у значений признака имеется порядок (высшее образование ценится выше среднего и т.д.).
- Количественные признаки: зарплата, возраст, кол-во детей в семье и т.п.

Ответ же к данной задаче либо 0 (человек не выплатит кредит вовремя) и 1 (человек выплатит кредит вовремя). Т.е. в задаче классификации значение, которое мы хотим предсказать (далее *целевая перменная)*, принадлежит конечному множеству. В нашей задаче, например, это $\{0; 1\}$. Бывают множества не только из 2 классов. Например, задача диагностики болезни по симптомам. Тут классы - это список болезней.

Как решается данная задача? В таких задачах нам требуется *обучающая выборка.* Это история того, как в прошлом наши клиенты выплачивали кредиты. Мы знаем о них всё: где они работают, сколько получают и т.п. А также нам известно, смогли они выплатить кредит или нет. Знание того, что мы предсказываем (целевой переменной) относит задачу к задачам *обучения с учителем.*

Далее модель машинного обучения находит закономерности в этой выборке. Например, если человек безработный, то скорее всего, он не выплатит кредит вовремя и т.д. Она запоминает эти зависимости, и когда приходит черёд узнать, а сможет ли новый клиент заплатить за кредит вовремя, модель смотрит на эти зависимости и выдаёт ответ. Список новых клиентов называется *тестовой выборкой.* Главное её отличие от обучающей выборки заключается в том, что для элементов из тестовой выборки неизвестна целевая переменная (в нашем случае - это выплатит клиент кредит или нет).

- *Регрессия.*

Ещё одним типом задач является регрессия. Например, можно рассмотреть такую задачу: мы хотим по росту родителей определять насколько высоким

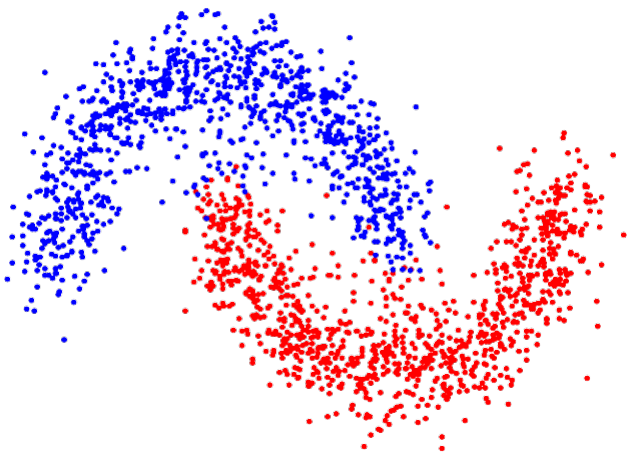
может быть их ребёнок. Действительно, как правило, есть зависимость, что у высоких родителей дети тоже имеют высокий рост и наоборот. Т.е. от классификации задача регрессии отличается тем, что тут целевая переменная - это вещественное число.

Обучающей выборкой в данной задаче будет набор троек: (рост родителя №1, рост родителя №2, рост ребёнка). Тестовой выборкой – набор двоек: (рост родителя №1, рост родителя №2).

- *Кластеризация.*

В предыдущих задачах в обучающих выборках нам было известна целевая переменная: выплатит ли человек кредит вовремя в задаче классификации, а также рост ребёнка в задаче регрессии. Однако, она не всегда известна. Например, мы провели социологический опрос, у нас есть много ответов на вопросы и нам хотелось бы сгруппировать их по поведению. Мы заранее не знаем, сколько таких кластеров получится, и что они из себя представляют. Это мы можем узнать только после того, как появятся сами кластеры.

Ещё простой и пробирочной задачей кластеризации является кластеризация точек на плоскости:



Алгоритм KNN для задачи классификации

Рассмотрим один из самых простых алгоритмов для задачи классификации KNN – k ближайших соседей. Пусть перед нами стоит задача бинарной классификации – 0 или 1. Рассмотрим ситуацию, когда все признаки – вещественные числа. В дальнейшем, при изучении курса вы сумеете переводить номинальные и порядковые признаки в числа. Пусть у нас имеется обучающая выборка $X_{train} = \{x_1, ..., x_N\}$, где каждый x_i – это вектор из

m значений, который соответствует объекту из m признаков. Для каждого объекта из обучающей выборки x_i известна целевая переменная $y_i \in \{0; 1\}$. А также пусть имеется тестовая выборка $X_{test} = \{x_{N+1}, ..., x_{N+M}\}$. Для них неизвестна целевая переменная, это мы должны выяснить сами.

Сам алгоритм:

1. Для каждого из объектов тестовой выборки x_{N+i}
2. Находим k ближайших к нему соседей из обучающей выборки X_{train}. Ближайших в смысле Евклидового расстояния:

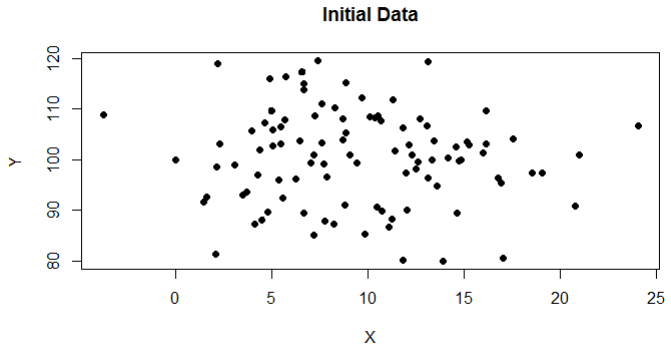
$$\rho(x; y) = \sqrt{(x_1 - y_1)^2 + ... + (x_m - y_m)^2}$$

3. Если среди этих ближайших соседей больше нулей, то для объекта x_{N+i} выдаём ответ – 0, иначе – 1.

Вот и весь алгоритм. Его можно немного модифицировать – использовать вместо Евклидового расстояния другие метрики. К сожалению, алгоритм не часто используется на практике, но неплохо подходит для обучения.

Алгоритм k-means для задачи кластеризации

Рассмотрим теперь задачу кластеризации и алгоритм k-means для её решения. Пусть в качестве объектов у нас Nточек на плоскости: $\{(x_1, y_1), ..., (x_N, y_N)\}$:



Мы хотим разбить эти точки на $k = 3$ кластера.

1. Выбираем $k = 3$ случайные точки из этого множетсва. Говорим, что они теперь являются центрами наших кластеров.
2. Для каждой из оставшихся точек смотрим, к какому из центров она ближе и определяем её в этот кластер.
3. У нас получилось разбить точки на 3 кластера. Естественно это не оптимальное разбиение. Найдём новые центры кластеров, а именно посчитаем центры масс каждого кластера. Например, если точки $\{(x_{i_1}, y_{i_1}),$

... , $(x_{i_n}, y_{i_n})\}$ попали в один кластер, то его новый центр масс будет иметь координаты:

$$x_M = \frac{x_{i_1} + ... + x_{i_n}}{n}$$

$$y_M = \frac{y_{i_1} + ... + y_{i_n}}{n}$$

4. Далее с этими новыми центарми масс переходим к шагу 2. Продолжаем до тех пор, пока кластеры перестанут меняться.