Задание 2. Постановка задачи машинного обучения.
Метод решающих деревьев.

I. Постановка задачи обучения по прецедентам (обучение с учителем или контролируемое обучение) (подробней см. в [1, стр. 4–17], или [2])

Есть набор объектов $x_i \in X$, с некоторыми признаками f_j.

Например,

(1) в задаче медицинской диагностики объекты x_i – это пациенты, признаками f_j для них могут быть их симптомы, результаты обследований, возраст и т.п.

(2) в задаче определения спама объекты x_i – это письма, признаки f_j для них – слова, встречающиеся в тексте, адрес письма и др.

Для некоторых объектов известны ответы. Например, в (1) известны диагнозы для ряда пациентов, в (2) есть набор писем, помеченных спам/не спам. Это называется обучающая выборка. На ее основе, можно попытаться найти закономерности для предсказания ответа для других объектов (построить решающую функцию). Например, в (1) для других пациентов и в (2) для других писем предсказать ответ.

Математическая постановка:

X – множество объектов;

Y – множество ответов;

y^*: X → Y – неизвестная зависимость (target function) между ними.

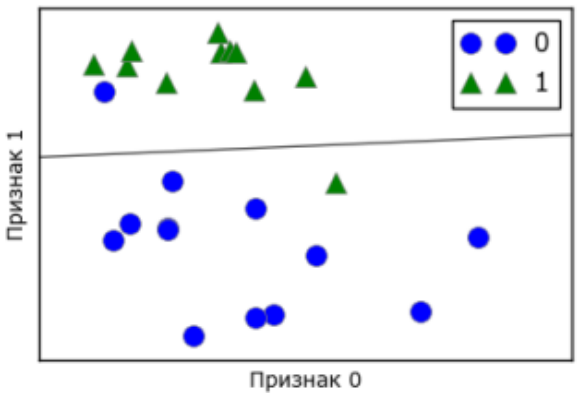
Дано:

$\{x_1, \dots, x_l\} \subset X$ – обучающая выборка (training sample)

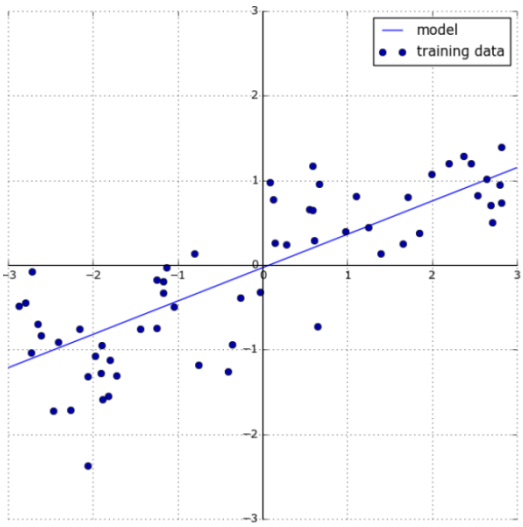
$y_i = y(x_i),\ i = 1, \dots l$ – известные ответы для обучающей выборки.

Найти:

a: X → Y – алгоритм, решающую функцию (decision function), приближающую функцию y^* на всем множестве X.



Например, данные и решающая функция, разделяющая их на 2 класса (задача классификации)



Или, данные и решающая функция, их приближающая
(задача восстановления регрессии)

Для решения задачи все признаки объекта должны быть сведены к числам. Признаки делятся на несколько типов: бинарные {0, 1}, номинальные признаки (конечное неупорядоченное множество), порядковые признаки (конечное упорядоченное множество), количественные признаки (непрерывное множество $\mathbb{R}$).

Совокупность признаковых описаний всех объектов обучающей выборки X, записывают в виде таблицы размера $l \times$ n, где l – количество объектов (строк матрицы), а n – количество признаков (столбцов матрицы). Называют ее матрицей «объектов–признаков»:

$$\begin{pmatrix} f_1(x_1), \dots, f_n(x_1) \\ \dots \\ f_1(x_l), \dots, f_n(x_l) \end{pmatrix}$$

Ей соответствует вектор известных ответов:

$$\begin{pmatrix} y(x_1) \\ \dots \\ y(x_l) \end{pmatrix}$$

Т.е. каждой строке матрицы $(f_1(x_i), \dots, f_n(x_i))$, или каждому объекту x_i, ставится в соответствие известный ответ $y(x_i)$. Матрица объектов–признаков является стандартным и наиболее распространённым способом представления исходных данных. Обычно ее в программе обозначают X (матрица), а вектор ответов обычно обозначают y (вектор).

Классификация и восстановление регрессии

В зависимости от множества допустимых ответов y задачи обучения по прецедентам делятся на:

1) задачи **классификации** (если y – конечный набор), и тогда нужно по объекту x предсказывать к какому классу y он принадлежит.

2) задачи **восстановления регрессии** (если y – непрерывное множество), и тогда по x предсказывается ответ – число y.

Модель алгоритмов и метод обучения

Моделью алгоритмов называется параметрическое семейство отображений

$$A = \{g(x,\theta) \mid \theta \in \Theta\},$$

где $g(x,\theta)$ некоторая фиксированная функция от x, с набором параметров $\theta \in \Theta$. Θ – называется пространством параметров или пространством поиска (search space). Часто используются линейные модели:

$$g(x,\theta) = \sum_{j=1}^{n} \theta_j f_j(x) \text{ – для задач восстановления регрессии;}$$

$$g(x,\theta) = \text{sign} \sum_{j=1}^{n} \theta_j f_j(x) \text{ – для задач классификации.}$$

Т.е. $g(x,\theta)$ – это общий вид функции, с помощью которой надеются приблизить y^*: X → Y – неизвестную зависимость (target function). В качестве признаков могут выбираться не только исходные данные, но и функции от них. Например, приближение полиномом некоторой степени от признаков.

Процесс подбора оптимальных параметров θ модели по обучающей выборке (X, y) называют настройкой (fitting) или обучением (training, learning) алгоритма $a \in A$. Конкретный метод обучения (learning algorithm) совершает подбор параметров θ. После подбора параметров θ, функцию $a(x) = g(x,\theta)$ можно использовать для предсказания для других x значений y: $y = a(x)$.

Два этапа: (1) обучения и (2) применения (предсказания)

Итак, в задачах обучения по прецедентам чётко различаются два этапа:

1) На этапе обучения метод строит алгоритм a.

2) На этапе применения алгоритм a для новых объектов x выдаёт ответы $y = a(x)$.

Этап обучения наиболее сложен. Как правило, он сводится к поиску параметров модели, доставляющих оптимальное значение заданному функционалу качества.

Функционал качества

Функция потерь (loss function) – это неотрицательная функция $L(a, x)$, характеризующая величину ошибки алгоритма a на объекте x. Если $L(a, x) = 0$, то ошибки на объекте x нет.

Наиболее часто используются следующие функции потерь:

$L(a, x) = [a(x) \neq y^*(x)]$ – индикатор ошибки (= 0 или 1) для классификации;

$L(a, x) = |a(x) - y^*(x)|$– отклонение;

$L(a, x) = (a(x) - y^*(x))^2$ – квадратичное отклонение.

Функционал качества алгоритма a:

$$Q(a, X, y) = \frac{1}{l}\sum_{i=1}^{l} L(a, x_i).$$

Функционал Q называют также функционалом средних потерь или эмпирическим риском.

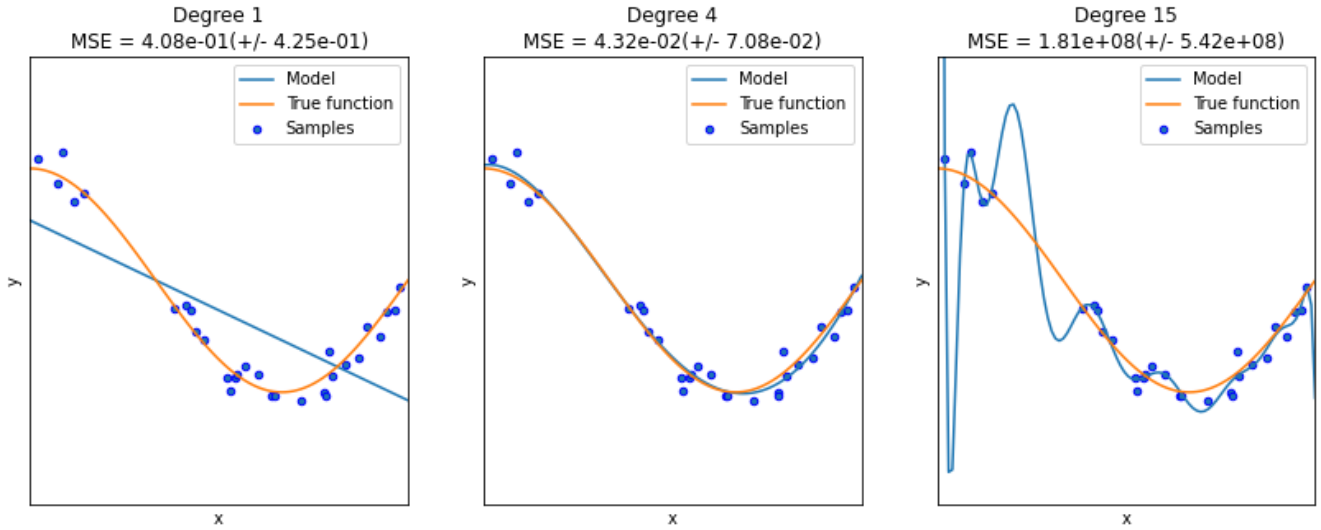
Классический метод обучения, называемый минимизацией эмпирического риска (empirical risk minimization, ERM), заключается в том, чтобы найти в заданной модели A алгоритм a, доставляющий минимальное значение функционалу качества Q на заданной обучающей выборке:

$$\mu = \arg\min_{a} Q(a, X, y).$$

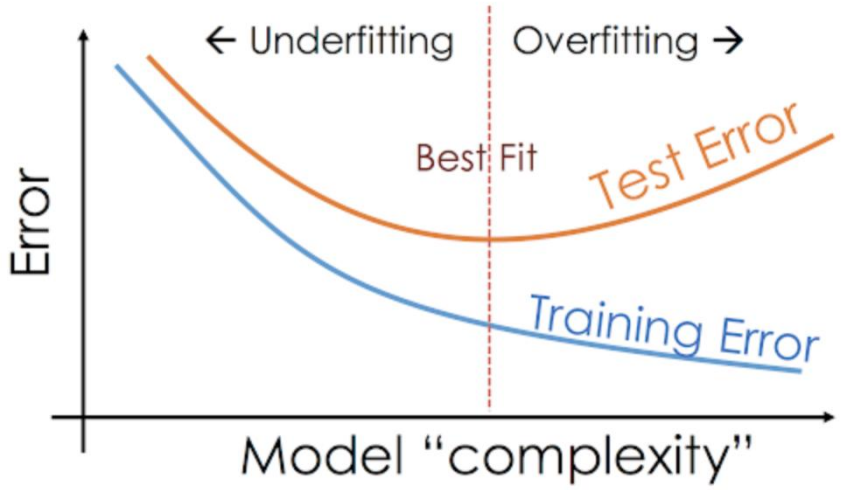
Проблема переобучения и понятие обобщающей способности

Если минимум достигается на обучающей выборке, это ещё не гарантирует, что метод будет хорошо приближать целевую зависимость $y^*(x)$ на произвольной контрольной выборке. Метод может из-за большого количества параметров начать хорошо приближать конкретные точки обучающей выборки, а не общую закономерность в данных. И для других данных давать существенно худший результат. Когда качество работы алгоритма на новых объектах, не вошедших в состав обучения, оказывается существенно хуже, чем на обучающей выборке, говорят об эффекте переобучения (overtraining) или переподгонки (overfitting). При решении практических задач с этим явлением приходится сталкиваться очень часто.

На следующих рисунках демонстрируется эффект недообучения / переобучения. На первом рисунке выбрана слишком простая функция для приближения закономерности в данных, модель недообучена (Underfitting), на втором рисунке выбрано оптимальное количество параметров и начальная закономерность приближается хорошо, на третьем рисунке модель переобучена (Overfitting): для точек обучающей выборки достигается практически идеальный результат, а для других данных модель будет давать результаты хуже.



Как борются с переобучением? Обычно выборку с известными данными и ответами для них (X, y) разбивают на части. Например, часть выборки используют при обучении (например, 75% случайно выбранных строк X) – обучающая выборка; а часть используют для проверки метода (25% оставшихся строк) – тестовая выборка. Важно обучаться на одних данных, а проверять метод на других!



И выбирать оптимальное количество параметров по лучшему результату на тестовой выборке.

Библиотека Scikit-Learn содержит множество методов обучения. Перед их использованием, их нужно подключить из библиотеки. Например, как в задании ниже:

```
from sklearn import tree
```

Один и тот же метод обычно представлен в двух вариантах: для задач классификации (например, класс DecisionTreeClassifier) и для задач регрессии (класс DecisionTreeRegressor). Они имеют гиперпараметры, которые задают свойства метода обучения и устанавливаются в момент создания объекта – экземпляра класса:

```
decision_tree = tree.DecisionTreeClassifier(random_state = 0, max_depth = 2)
```

Значения обычно установлены по умолчанию в самые часто используемые, поэтому большую их часть можно не менять. Метод fit созданного объекта запускает метод обучения и возвращает решающую функцию:

```
decision_tree = decision_tree.fit(X, y)
```

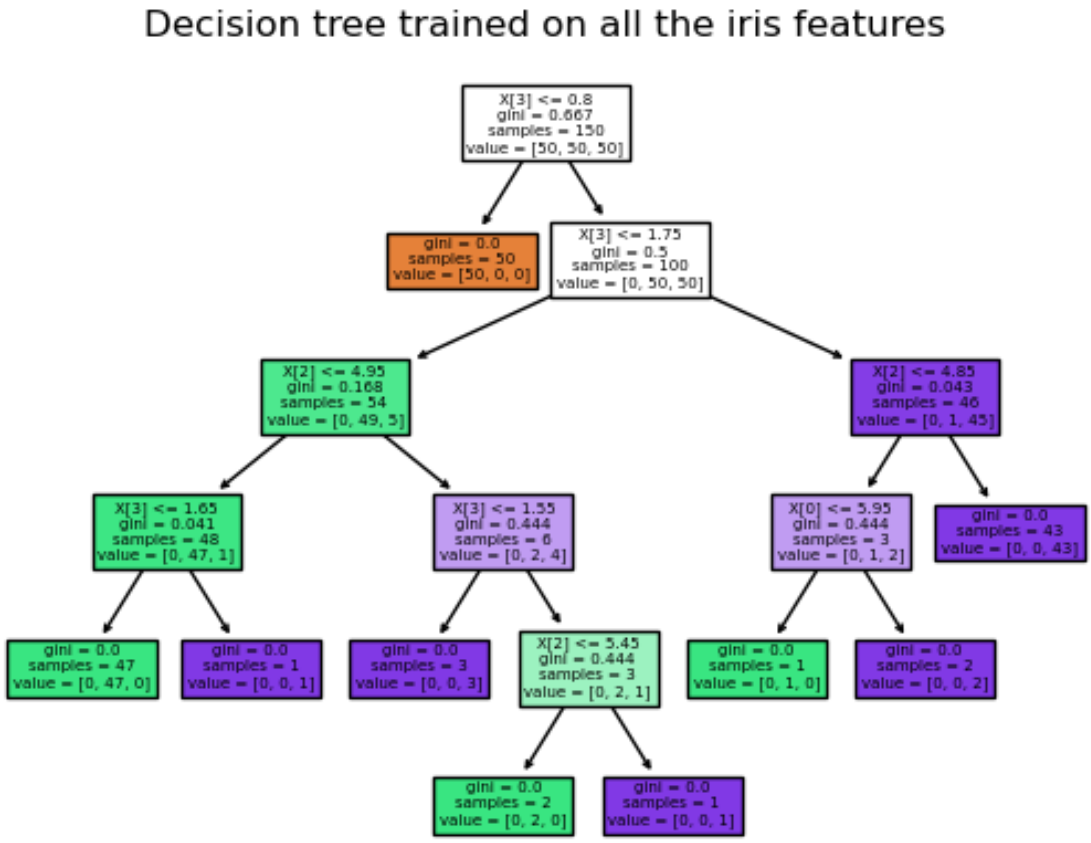
После этого, метод predict может использоваться для предсказания.

Библиотека **Scikit-Learn** содержит кроме самих методов машинного обучения, многое, что понадобится при решении задач машинного обучения, например, разбиение на обучающую/тестовую выборки, применение кросс-валидации, нормализация признаков, вычисление различных мер качества классификации и т.д.

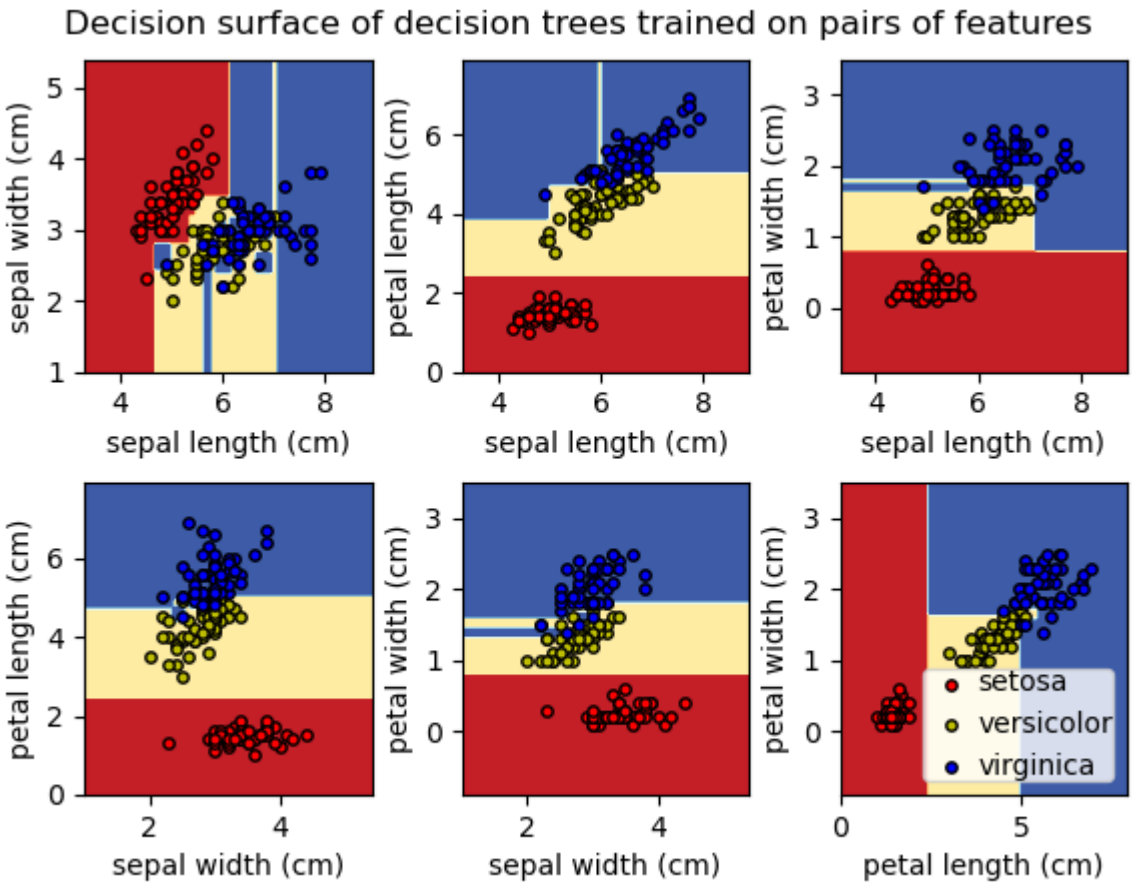
II. Метод решающих деревьев
(подробней см. [3, стр. 124–139], [4, стр. 223–239])
(документация: https://scikit-learn.org/stable/modules/tree.html scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html)

Метод решающих деревьев является одним из методов машинного обучения. Он является простым для понимания и интерпретации результатов. Построенное решающее дерево можно применять и без применения компьютера к новым данным.

Обычно в пакетах (и в библиотеке Scikit-Learn) применяются бинарные деревья. Область данных разбивается гиперповерхностями вида x[i] <= число.



Постепенно спускаясь от корня дерева к листьям и выбирая ветви в зависимости от ответа на вопрос x[i] <= число, мы относим выбранную точку к определенному классу.



Для того чтобы разделить узлы в самых информативных признаках, применяются 3 разных критерия (по умолчанию в Scikit-Learn установлен критерий Джини: criterion= “gini”).

Недостатком решающих деревьев является их переобучение при отсутствии ограничений на деревья. Обычно ограничивают высоту деревьев (max_depth). Также можно ограничить другие параметры (min_samples_split, min_samples_leaf, , min_weight_fraction_leaf, max_features, max_leaf_nodes). После построения дерева, можно оценить важность признаков (feature_importances), т.е. выделить те признаки, которые больше всего участвовали при построении дерева.

Метод решающих деревьев часто не дает самый лучший ответ по сравнению с другими методами, но он является основой для одного из самых популярных методов машинного обучения – метод случайного леса.

Задания

I. Прочитайте пункт “I. Постановка задачи обучения по прецедентам” (и/или рекомендованную литературу к нему) и ответьте на вопросы:

1) Что такое объекты, признаки, ответы? Приведите пример.
2) Какие данные содержатся в матрице «объектов–признаков»?
3) Что такое модель алгоритмов и метод обучения?
4) Какие два этапа можно выделить в задачах обучения?
5) Что такое переобучение? Как с ним бороться?

II. Прочитайте пункт “II. Метод решающих деревьев” и рекомендованную литературу к нему, и ответьте на вопросы:

1) Как строится решающее дерево?
2) Какие критерии разбиения используются при построении деревьев?
3) Какие преимущества и недостатки метода?

III. Выполните задание из в файла "2_statement-importance.pdf"

Литература

[1] К.В. Воронцов Математические методы обучения по прецедентам (теория обучения машин). 141 с. (Voron-ML-1.pdf)

[2] Машинное обучение (курс лекций, К.В.Воронцов) (machinelearning.ru) пункт 2.1 Основные понятия и примеры прикладных задач (запись лекции: https://www.youtube.com/watch?v=xccjt6lOoow)

[3] Рашка Себастьян, Мирджалили Вахид Python и машинное обучение: машинное и глубокое обучение с использованием Python, scikit-learn и TensorFlow 2, 3-е изд.: Пер. с англ. СПб. : ООО "Диалектика", 2020. 848 с.

[4] Жерон, Орельен. Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow: концепции, инструменты и техники для создания интеллектуальных систем. Пер. с англ. СпБ.: ООО Альфа-книга: 2018. 688 с.