



Введение в машинное обучение. Практический подход.

Арсений Горин

МГТУ, 21 ноября 2016

План курса

2 лекции, 1 практическое занятие (или скорее 1.5/1.5)

- Вводная лекция:
основные понятия и алгоритмы
- Примеры решения задач
с использованием языка python
и библиотеки scikit-learn
- Разбор задач
и дополнительный материал
(кластеризация, бустинг,
глубинное обучение...)



Вводная лекция. Содержание

1 Введение

- Основные понятия
- Виды систем МО и примеры задач

2 Модели обучения с учителем

- Задача регрессии
- Обобщение и перер обучение
- Задача классификации
- Обзор и сравнение некоторых классификаторов

3 Заключение

Содержание

1 Введение

- Основные понятия
- Виды систем МО и примеры задач

2 Модели обучения с учителем

- Задача регрессии
- Обобщение и перер обучение
- Задача классификации
- Обзор и сравнение некоторых классификаторов

3 Заключение

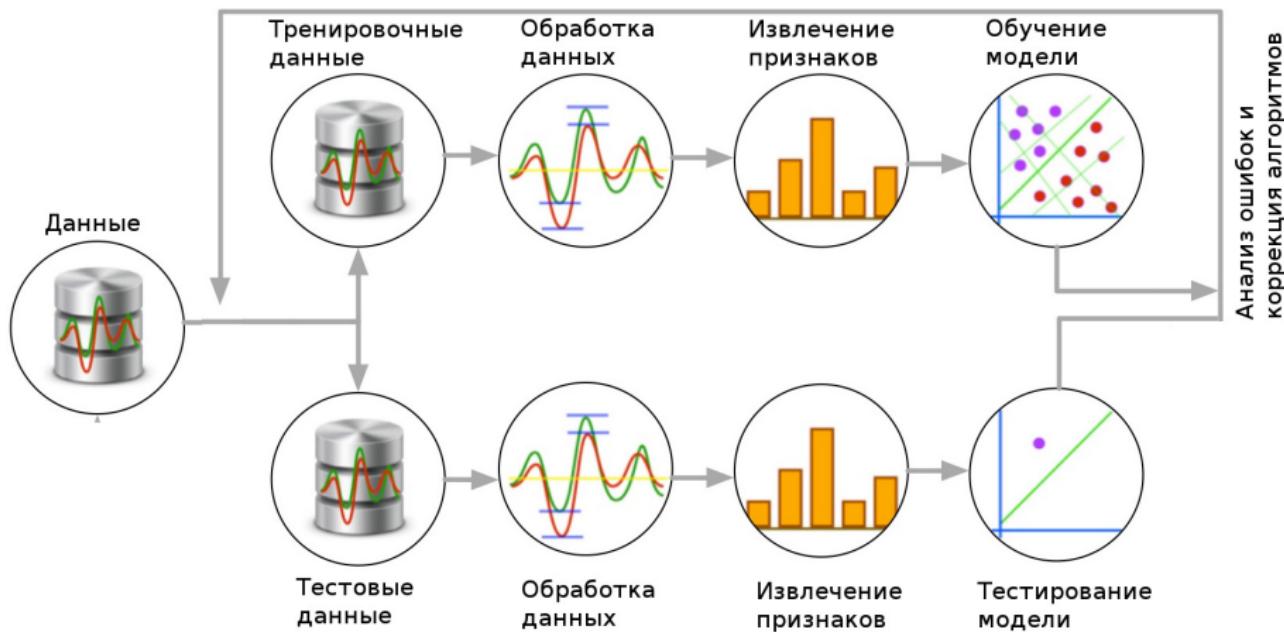
Что такое машинное обучение

Машинное обучение - это процесс, в результате которого машина (компьютер) способна показывать поведение, которое в нее не было явно заложено (запрограммировано). (Arthur Samuel, 1959)

*Компьютерная программа обучается на основе опыта E по отношению к некоторому классу задач T и меры качества P , если качество решения задач из T , измеренное на основе P , улучшается с приобретением опыта E .
(Tom Mitchell, 1997)*

- Алгоритмы искусственного интеллекта, способного к обучению
- Методы теории вероятности и статистики, линейной алгебры, оптимизации

Этапы разработки систем МО



Основные классы задач МО

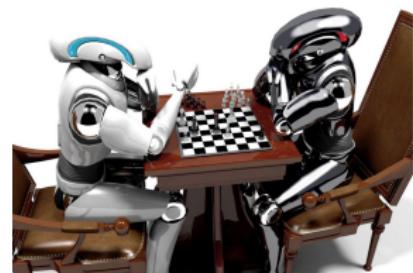
Обучение с учителем
(supervised learning)



Обучение без учителя
(unsupervised learning)



Обучение с подкреплением
(reinforcement learning)



Обучение с учителем (supervised training)

Формальное определение:

Дано:

- Множество объектов (objects) X и их признаков (features) F
- Подмножество объектов тренировки: $X_{train} \subset X$, $X_{train} = \{x_1, \dots, x_n\}$
- Подмножество ответов $Y_{train} = \{y(x_1), \dots, y(x_n)\}$
- Тестовое подмножество $X_{test} \subset X$, $X_{test} = \{x'_1, \dots, x'_m\}$

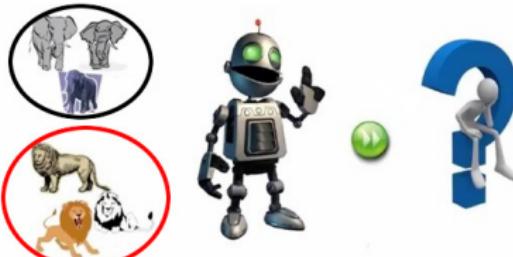
Найти:

- Алгоритм, или решающую функцию (decision function) $a : X \rightarrow Y$, где Y - множество всех ответов

Простым языком

По тренировочным примерам (объекты - ответы) научиться предсказывать ответы для всех объектов

Обучение с учителем. Пример классификации



объект	признаки				ответ (класс)
	цвет	высота	кол-во ног	хобот	
x_1	серый	3.5	4	да	слон
x_2	оранжевый	0.7	4	нет	лев
...
x_n	белый	0.6	3	нет	лев
x'_1	белый	3.1	4	да	?

Более реальные примеры:

- распознавание речи и изображений
- классификация спама и нелегальных банковских транзакций
- медицинский диагноз

Обучение с учителем. Пример регрессии



объект	признаки				ответ (размер уха)
	цвет	высота	кол-во ног	хобот	
x_1	серый	3.5	4	да	0.8
x_2	оранжевый	0.7	4	нет	0.2
...
x_n	белый	0.6	3	нет	0.3
x'_1	белый	3.1	4	да	?

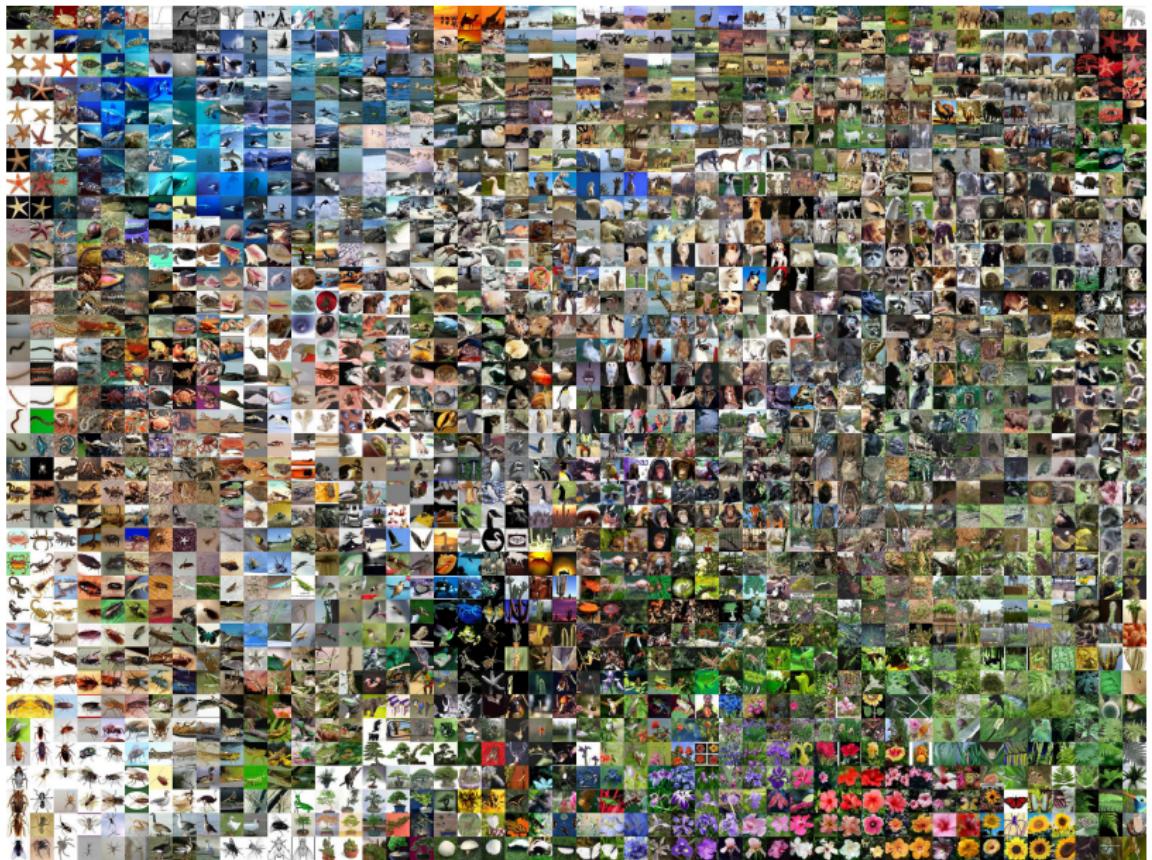
Более реальные примеры:

- оценка кредитоспособности, спроса
- прогнозирование (курс валют, стоимость ценных бумаг)

Обучение без учителя (unsupervised learning)

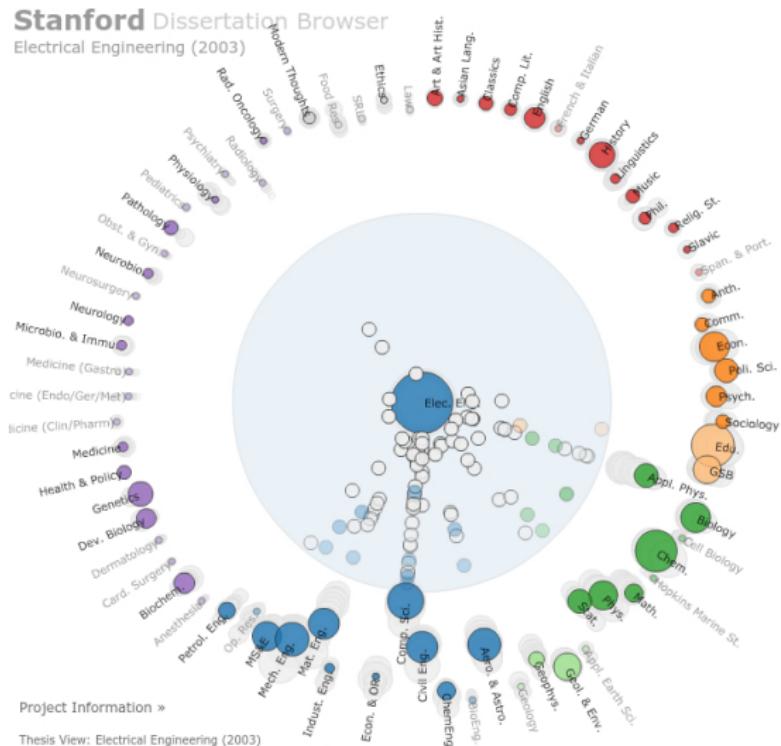
- Как и в случае с supervised learning, заданы объекты с некоторыми признаками
- Отсутствуют ответы
- Задача - найти некоторую скрытую структуру в неразмеченных данных
- Так как не известны ответы, сложно посчитать ошибку

Обучение без учителя. Пример (flickr t-SNE)



Обучение без учителя. Еще пример

Кластеризация диссертаций Stanford по словам, сочетаниям и цитированиям
(<http://nlp.stanford.edu/projects/dissertations/browser.html>)



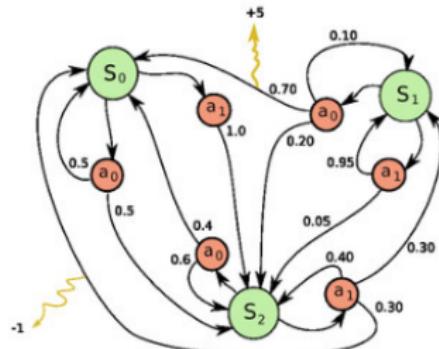
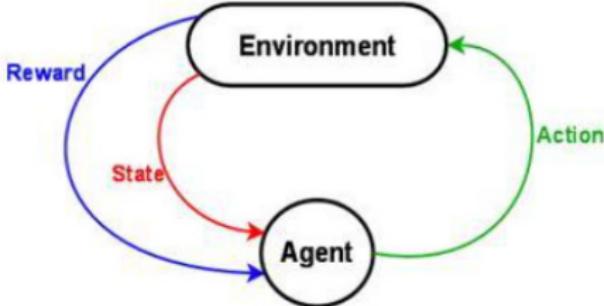
Обучение с подкреплением (reinforcement learning)

Задача заключается в обучении агента (agent) взаимодействию со средой (environment)

Марковский процесс принятия решений (MDP: Markov Decision Process)

- Конечное число состояний S
- Набор действий A_s , которые можно выполнить в каждом состоянии s
- Вероятности перехода из s_t в s_{t+1} действием a :
 $P_a(s, s') = P(s_{t+1} = s' | s_t = s, a_t = a)$
- Вознаграждение за переход из s в s' действием a : $R_a(s, s')$

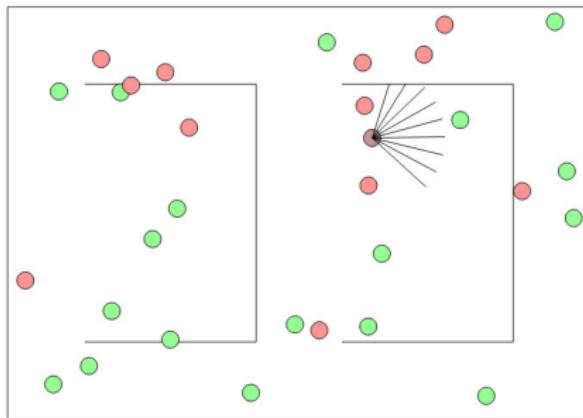
Задача - найти алгоритм, максимизируя суммарное вознаграждение.



Обучение с подкреплением. Примеры

ИИ, учащийся игре Atari:

<http://cs.stanford.edu/people/karpathy/convnetjs/demo/rldemo.html>



Еще известный пример: AlphaGo - ИИ, победивший чемпиона игры Go Lee Sedol (март 2016)

Содержание

1 Введение

- Основные понятия
- Виды систем МО и примеры задач

2 Модели обучения с учителем

- Задача регрессии
- Обобщение и перер обучение
- Задача классификации
- Обзор и сравнение некоторых классификаторов

3 Заключение

Обучение с учителем. Обзор

Обучение (training, learning):

$$\begin{pmatrix} f_1(x_1) & \dots & f_k(x_1) \\ \vdots & \vdots & \vdots \\ f_1(x_n) & \dots & f_k(x_n) \end{pmatrix} \rightarrow \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Получаем некоторый метод μ , который строит алгоритм $a = \mu(X_{train})$ по выборке (X_{train}, Y_{train})

Применение (testing):

$$\begin{pmatrix} f_1(x'_1) & \dots & f_k(x'_1) \\ \vdots & \vdots & \vdots \\ f_1(x'_m) & \dots & f_k(x'_m) \end{pmatrix} \rightarrow \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix}$$

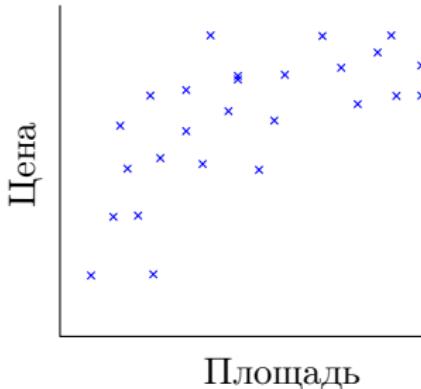
Используем алгоритм для предсказания (prediction) ответов для каждого объекта тестовой выборки X_{test}

Пример регрессии

- В задаче регрессии множество ответов - числа
- Числа известны для обучающей выборки, но не известны для тестовой



Игрушечный пример: оценка стоимости жилья, зная его площадь



- Зная множество пар (площадь, цена), как примерно оценить стоимость новой квартиры?
- Как определить признаки?
- Какие тут вообще могут быть проблемы?

Регрессия. Продолжение.

Решим задачу методом наименьших квадратов.

Линейная модель:

- $g(x, \Theta) = \sum_{j=1}^n \Theta_j f_j(x)$
- Θ - **параметры модели** (веса, коэффициенты)
- **Признаки** - некоторые функции от x , например $\{1, x, x^2, e^x, \sin x, \ln x\}$

Задача

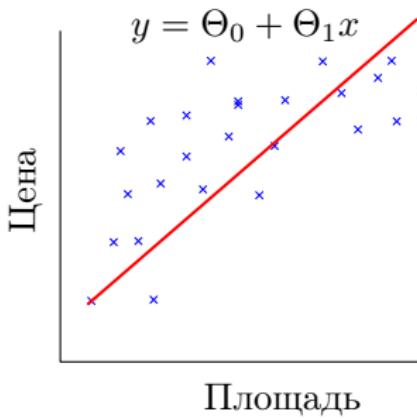
- минимизация функционала $\mu(X_{train}) = \arg \min_{\Theta} \sum_{i=1}^n (g(x_i, \Theta) - y_i)^2$
- $(g(x_i, \Theta) - y_i)^2$ называется **функцией потерь** (loss function) или ошибкой

А в чем проблема-то?

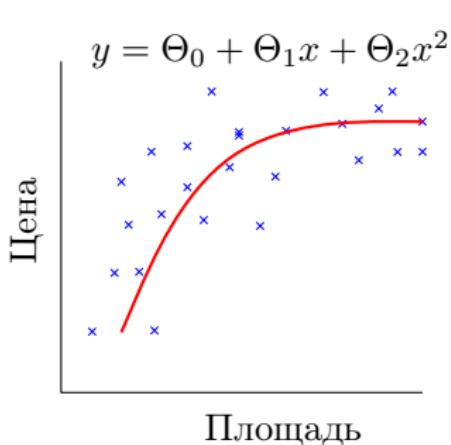
- Как оптимально выбрать Θ
- Модель должна работать не только на тренировочных, но и на новых (тестовых) данных - проблема обобщения (generalization problem)

Проблема (недо|пере)обучения

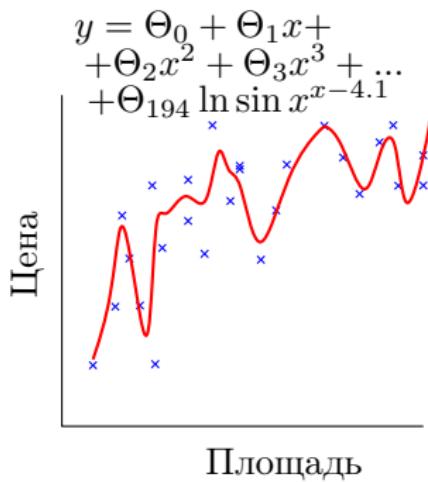
Модель слишком проста
(underfit)



Хорошая модель



Модель переобучена
(overfit)

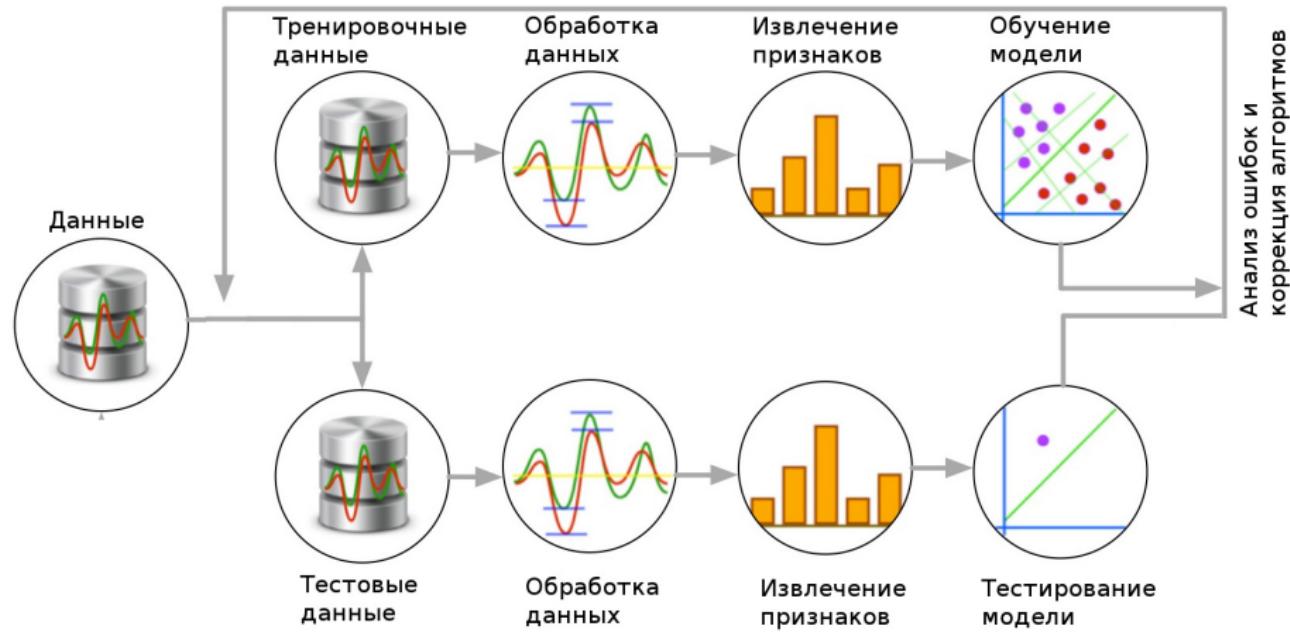


Переобучение

- Сильно снижает точность на тестовых данных
- Определяется методом анализа на независимых данных
- Много способов борьбы (ограничения на параметры - регуляризация)
- Наверно самая серьезная проблема для сложных нелинейных моделей (например, многослойных нейросетей)

Этапы разработки систем МО. Повтор

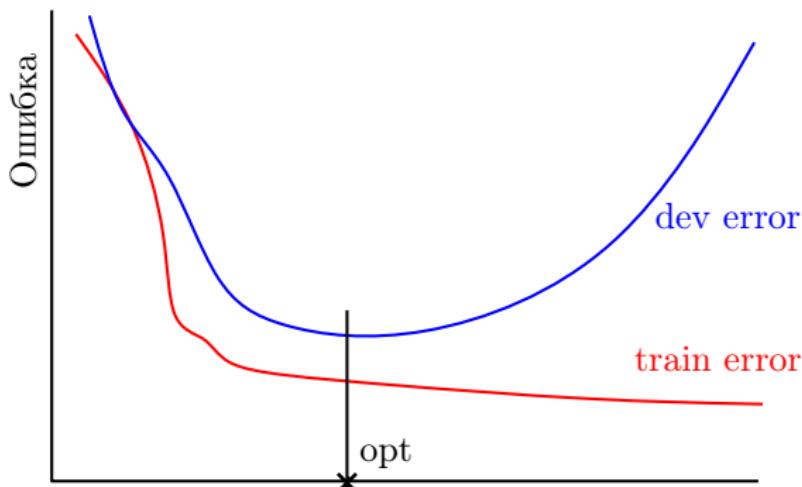
В идеале желательно иметь не только тестовые, но и так называемые контрольные данные (**hold-out**)



Еще о подготовке экспериментов

Данные:

- Обучающая выборка (train) - оценка параметров модели
- Настроечная выборка (dev) - для подбора параметров, контролирующих сложность модели
- Тестовая выборка (test, eval) - данные, отражающие реальную точность модели



Сложность модели / число параметров / время обучения

Классификация

- Основное отличие от регрессии - дискретность ответов $Y = \{1, \dots, M\}$
- Частный случай - бинарная классификация, когда $Y = \{-1, 1\}$ (например, классификация спама)
- Реже и сложнее если классы могут пересекаться



Несмотря на теоретические различия, алгоритмы регрессии и классификации часто похожи

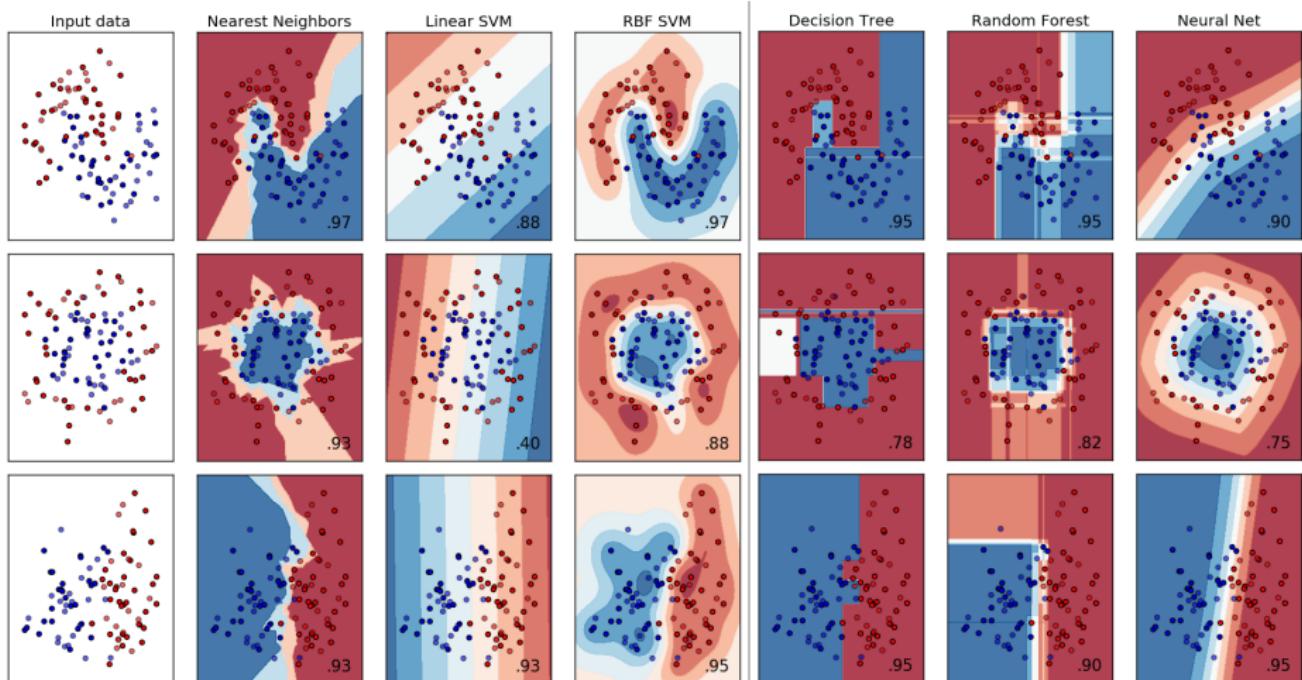
	Регрессия (МНК)	Классификация
функция потерь (loss)	$(a(x) - y(x))^2$	$[a(x) \neq y(x)]$
линейная модель	$g(x, \Theta) = \sum_{j=1}^n \Theta_j f_j(x)$	$g(x, \Theta) = \text{sign} \sum_{j=1}^n \Theta_j f_j(x)$

Популярные модели классификации

- Градиентные
 - Линейный классификатор (linear classifier)
 - Нейронные сети (neural networks)
- Метрические
 - Метод опорных векторов (support vector machine)
 - Метод ближайших соседей (nearest neighbors)
- Логические
 - Решающие деревья (decision trees)
 - Случайный лес (random forest)
- Байесовские
 - Наивный Байес (naive Bayes)
 - Смеси вероятностных распределений (mixture models)

Выбор классификатора

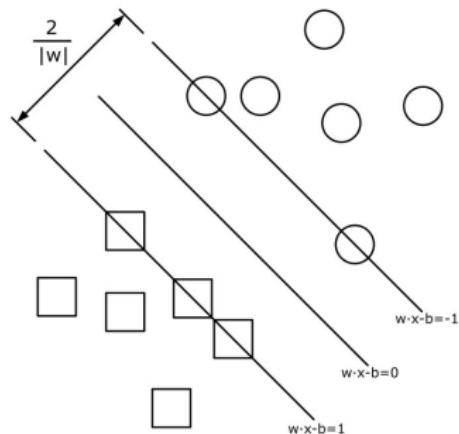
- Задача не всегда тривиальная. Учитывают специфику данных, скорость работы, использование памяти.
- Ни одно из последних соревнований по МО kaggle не было выиграно без использования комбинации нескольких моделей



Метод опорных векторов (support vector machine)

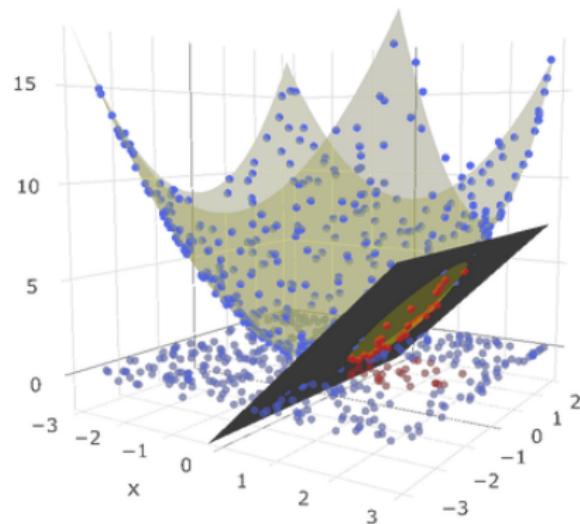
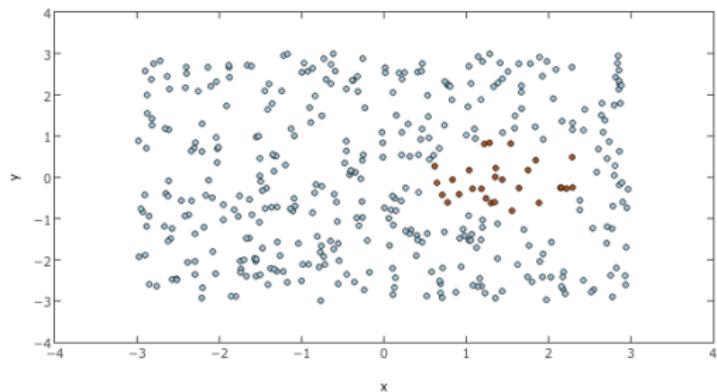
SVM - одна из лучших (а многие считают, что и самая лучшая)
“off-the-shelf” модель обучения с учителем (Andrew Ng)

- линейный классификатор
- при обучении выбираются опорные точки, ищется разделяющая полоса максимальной ширины

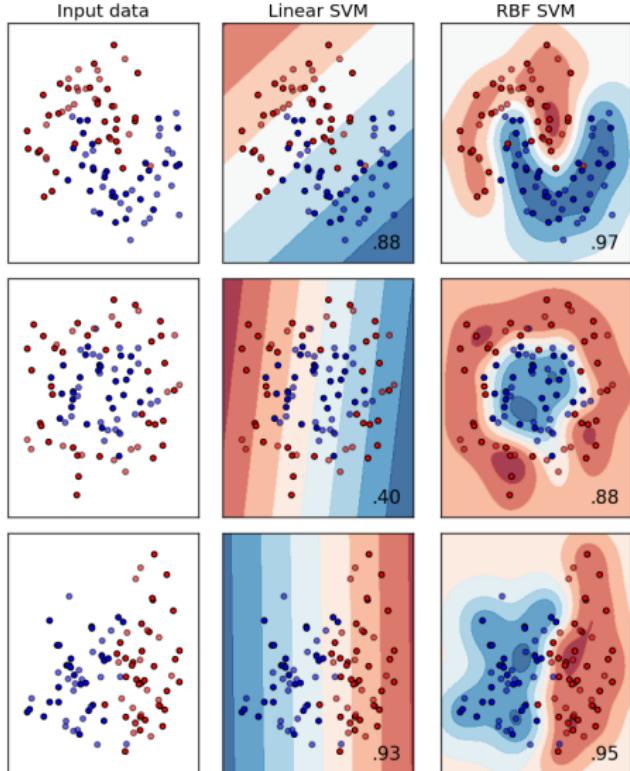


Метод опорных векторов (support vector machine)

- Если классы не разделяются линейно, используют нелинейное преобразование признаков (ядро, kernel trick)
- Идея - перейти в такое пространство признаков, в котором классы разделяются линейно



Метод опорных векторов (support vector machine)



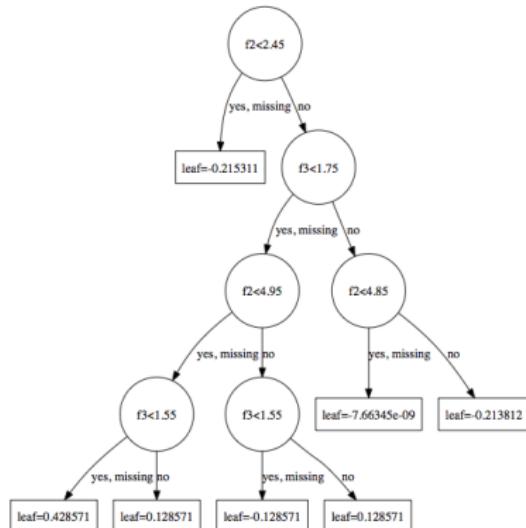
- + Устойчивость решения (обучения)
- + “Уверенная” классификация благодаря максимизации разделяющей поверхности
- Неустойчивость к шумовым выбросам (могут оказаться опорными точками)
- Для нелинейной классификации не ясно, как выбрать соответствующее пространство

Решающие деревья (Decision trees)

Логический классификатор, строящий в каждой вершине дерева предикаты

$R : X \rightarrow 0, 1$, так что:

- R записывается на естественном языке
- R зависит от подмножества признаков
- Признаки выбираются в каждой вершине дерева по принципу максимизации информативности одного из классов (например, энтропия)
- На листьях дерева - ответы (один ответ зачастую на нескольких листьях)

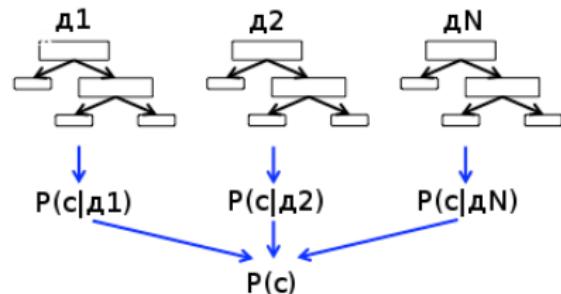


Решающие деревья (Decision trees)

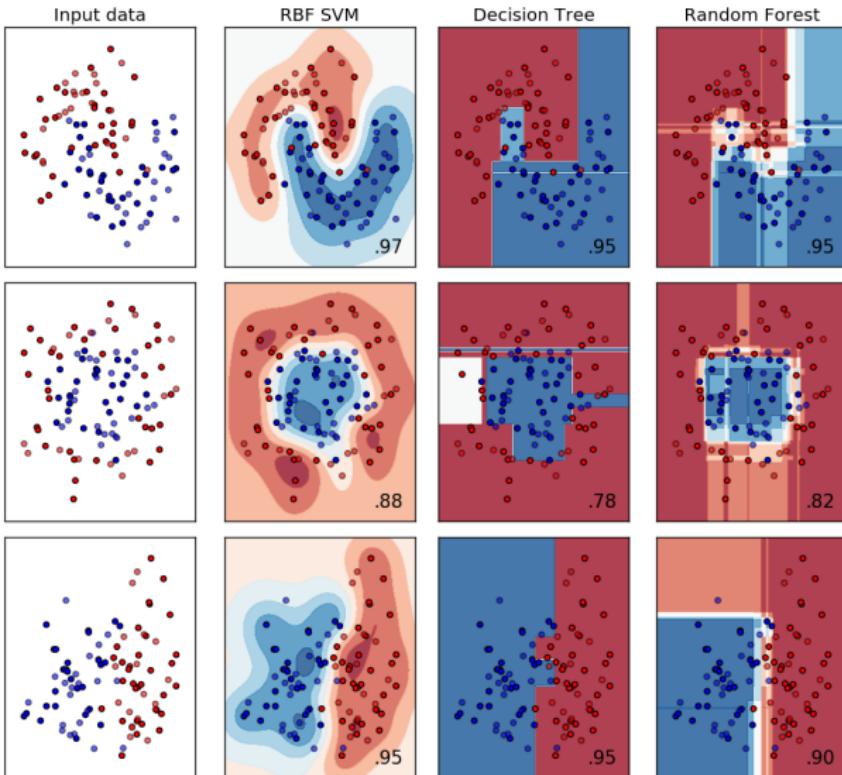
- + Интерпретируемый результат (можно использовать в рисковых системах, например в банковском секторе, или в медицине)
- + Не чувствительны к пропускам в данных и нормализации
- + Высокая скорость работы ⇒ часто используются как базовые классификаторы в каскаде
- С ростом глубины дерева уменьшается надежность оценок (мало объектов чтобы посчитать свойства)
- Чувствительны к шуму

Случайный лес деревьев (Random Forest)

- Случайный лес - пример популярного метода MO bagging
- Формируется N случайных подмножеств обучающей выборки
- На каждом подмножестве тренируется отдельное решающее дерево
- Для предсказания так же используются все деревья, результат выбирается голосованием или взвешиванием
- В практических задачах целесообразно использовать 10-1000 деревьев



Деревья и лес



Содержание

1 Введение

- Основные понятия
- Виды систем МО и примеры задач

2 Модели обучения с учителем

- Задача регрессии
- Обобщение и перер обучение
- Задача классификации
- Обзор и сравнение некоторых классификаторов

3 Заключение

Заключение

- МО - бурно развивающаяся и используемая на практике сегодня наука
- Машины могут учиться
 - По размеченным данным (supervised training)
 - По неразмеченным данным (unsupervised training)
 - Играть в рамках заданных правил (reinforcement learning)
- Еще они могут
 - Работать с вещественными числами (regression)
 - Работать с абстрактными классами данных (classification)
 - Недоучиваться и переобучаться
- Наши задачи:
 - Иметь общее понимание того, на каких принципах работают те или иные алгоритмы (линейный классификатор, SVM, логические классификаторы)
 - Исследовать, ставить себе задачи и решать их
 - Не забывать этого парня



Полезные ссылки

- <http://www.machinelearning.ru> - наверно самый полный ресурс о МО в рунете
- <https://yandexdataschool.ru/edu-process/courses/machine-learning> (сильно теоретический курс К.Воронцова, ШАД)
- <https://www.coursera.org/learn/machine-learning> - отличный курс от Andrew Ng, Stanford
- <http://www.dataschool.io/machine-learning-with-scikit-learn> - теория и практика МО в пакете scikit-learn
- <https://www.kaggle.com> - задачи и соревнования по МО

На следующее занятие

- Установить VirtualBox (<https://www.virtualbox.org/wiki/Downloads>)
- Скачать у меня образ xubuntu
- В VirtualBox выполнить “Machine → add”, добавить машину и запустить ее
- Запустить файл RUNME на рабочем столе. Должно быть так

The screenshot shows a Jupyter Notebook interface. At the top, there's a toolbar with File, Edit, View, Insert, Cell, Kernel, Help, and a Python 2 tab. Below the toolbar is a toolbar with icons for file operations like Open, Save, and Print, along with buttons for Cell, Kernel, and CellToolbar.

The main area contains two code cells:

- Agenda**
 - What is the famous iris dataset, and how does it relate to machine learning?
 - How do we load the iris dataset into scikit-learn?
 - How do we describe a dataset using machine learning terminology?
 - What are scikit-learn's four key requirements for working with data?
- Introducing the iris dataset**

Below this title is an image of an Iris flower with a coordinate system overlaid on its petals, indicating the points of measurement for the dataset.

- Если что-то пошло не так, пишите gorinars@yandex.ru