Введение в Data Science
Занятие 10. Метод ансамблей

Николай Анохин Михаил Фирулик

17 мая 2014 г.

Метод ансамблей

Boosting

Bagging

Комбинирование моделей

Задача классификации

Пусть дана выборка, состоящая из обучающих объектов

$$\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N),$$

и соответствующих значений целевой переменной

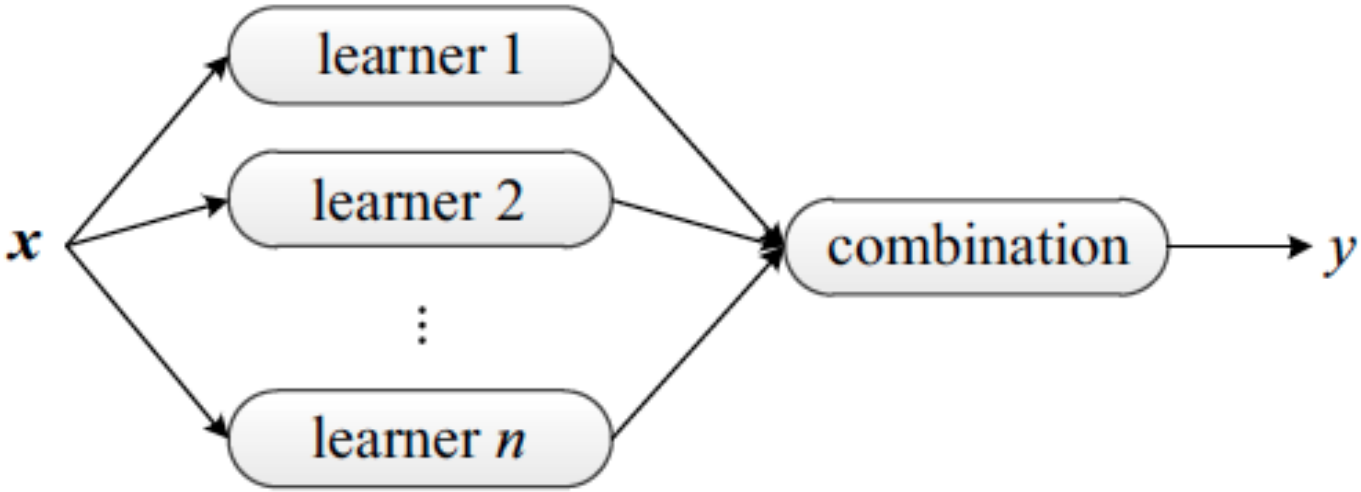
$$Y = (y_1, \ldots, y_N) = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)).$$

Требуется найти функцию $h(\mathbf{x})$, наилучшим образом приближающую $f(\mathbf{x})$, то есть точно предсказывающую значение целевой переменной для любых $\mathbf{x}$.

Недостатки одиночных моделей

- Statistical issue
  риск выбрать неправильную гипотезу из возможного набора, учитывая ограниченность выборки
- Computational issue
  риск получить локальный минимум в результате оптимизации
- Representational issue
  риск, что ни одна из моделей не окажется достаточно хорошей

Метод ансамблей



Идея

Построить несколько **базовых моделей** и правильным образом **скомбинировать** их для принятия решения. В идеале базовые модели должны быть максимально *точными* и при этом *разнообразными*.

Виды ансамблей

- комбинация классификаторов (combining classifiers)
  pattern recognition
- ансамбль слабых моделей (ensemble of weak learners)
  machine learning
- смесь экспертов (mixture of experts)
  neural networks

Стоит ли?

- Рекомендательные системы
  Победитель Netflx Prize $1M (первое и второе места)
- Компьютерное зрение
  AdaBoost with cascade – определение лиц на фото (или стыковка с МКС ☻)
- Медицинская диагностика
  Определение болезни на ранней стадии

Boosting

Пусть дан алгоритм обучения “слабой” модели – такой, которая только немного лучше случайности

Идея метода

Последовательно обучать слабые модели так, что каждая следующая модель “исправляет ошибки” предыдущих. Для предсказания используется комбинация из всех моделей последовательности.

AdaBoost

ada_boost($\mathbf{X}$, Y, T):
  инициализируем $\mathcal{D}_1 = 1/m$
  for $t = 1, \ldots, T$:
    обучаем модель $h_t(\mathbf{x}) = \mathcal{L}(\mathbf{X}, Y)$,
      принимая во внимание распределение $\mathcal{D}_t$
    вычисляем ошибку $h_t(\mathbf{x})$: $\epsilon_t = P_{\mathbf{x} \sim \mathcal{D}_t}(h_t(\mathbf{x}) \neq f(\mathbf{x}))$
    if $\epsilon_t > error\_rdm$:
      break
    вычисляем вес $h_t(\mathbf{x})$: $a_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$
    новое распределение: $\mathcal{D}_{t+1}(\mathbf{x}) = \frac{\mathcal{D}_t(\mathbf{x})}{Z_t} \exp(-a_t f(\mathbf{x}) h_t(\mathbf{x}))$
  return $H(\mathbf{x}) = \mathrm{sign} \sum_{t=1}^{T} a_t h_t(\mathbf{x})$

Свойства AdaBoost

- Минимизирует экспоненциальную ошибку (exponential loss)

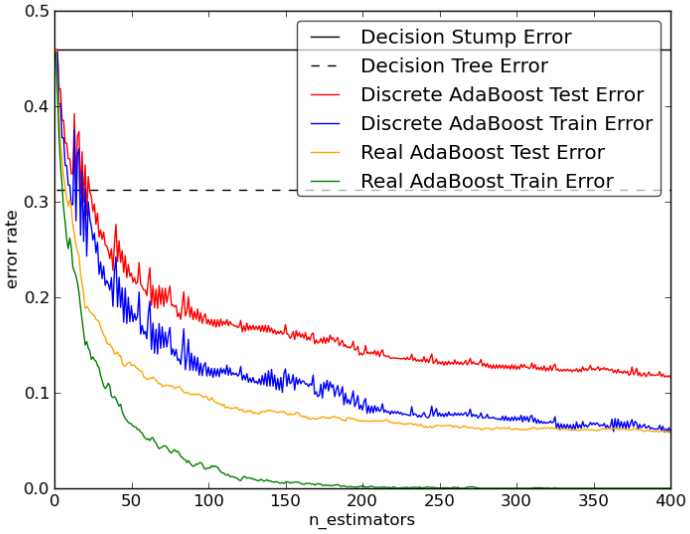
$$L_{exp}(h|\mathcal{D}) = E_{\mathbf{x}\sim\mathcal{D}}[e^{-f(\mathbf{x})h(\mathbf{x})}]$$

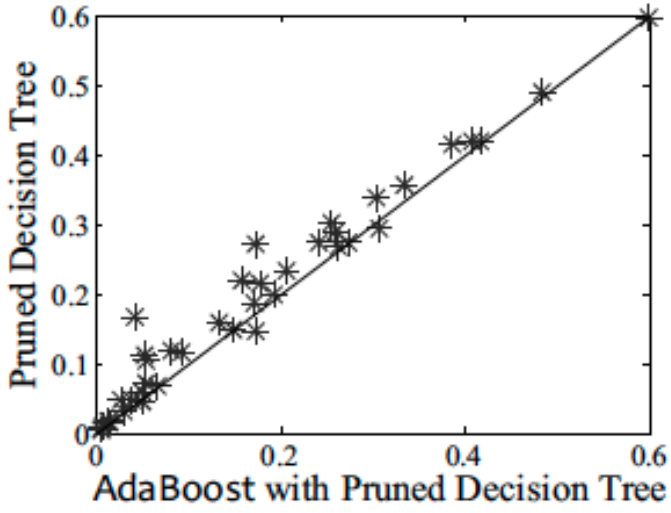
- Требует обучения модели с учетом распределения
  Варианты: re-weighting или re-sampling
- Ошибка классификации

$$\epsilon_{\mathcal{D}} \leq \epsilon_{\mathbf{x}} + O\left(\sqrt{\frac{dT}{N}}\right)$$

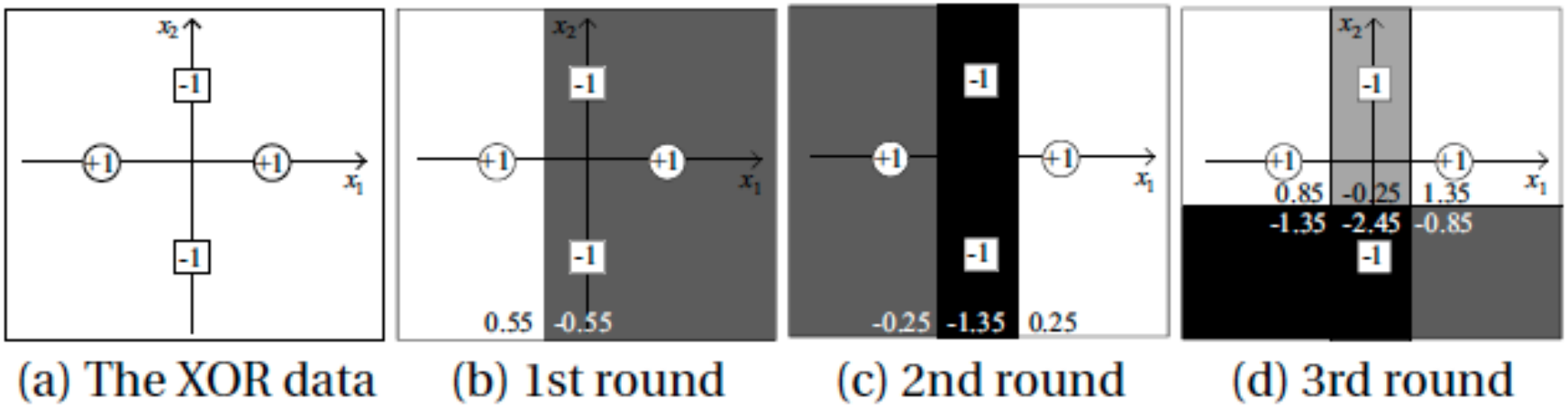
(d отражает "сложность" классификатора)

AdaBoost: тесты





Пример



(a) The XOR data (b) 1st round (c) 2nd round (d) 3rd round

$$h_1(\mathbf{x}) = \begin{cases} +1, & \text{если } x_1 > -0.5 \\ -1, & \text{иначе} \end{cases}$$

$$h_2(\mathbf{x}) = \begin{cases} -1, & \text{если } x_1 > -0.5 \\ +1, & \text{иначе} \end{cases}$$

$$h_3(\mathbf{x}) = \begin{cases} +1, & \text{если } x_1 > +0.5 \\ -1, & \text{иначе} \end{cases}$$

$$h_4(\mathbf{x}) = \begin{cases} -1, & \text{если } x_1 > +0.5 \\ +1, & \text{иначе} \end{cases}$$

$$h_5(\mathbf{x}) = \begin{cases} +1, & \text{если } x_2 > -0.5 \\ -1, & \text{иначе} \end{cases}$$

$$h_6(\mathbf{x}) = \begin{cases} -1, & \text{если } x_2 > -0.5 \\ +1, & \text{иначе} \end{cases}$$

$$h_7(\mathbf{x}) = \begin{cases} +1, & \text{если } x_2 > +0.5 \\ -1, & \text{иначе} \end{cases}$$

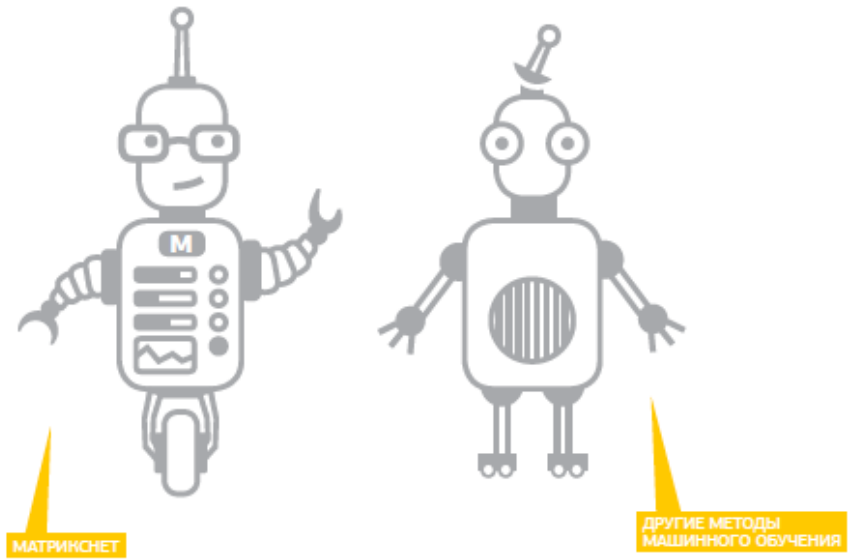
$$h_8(\mathbf{x}) = \begin{cases} -1, & \text{если } x_2 > +0.5 \\ +1, & \text{иначе} \end{cases}$$

Еще boosting

- LogitBoost

$$L_{log}(h|\mathcal{D}) = E_{\mathbf{x}\sim\mathcal{D}}\left[\ln\left(1 + e^{-2f(\mathbf{x})h(\mathbf{x})}\right)\right]$$

- GradientBoosting
  оптимизация произвольной функции потерь + регуляризация



AdaBoost. Итоги

+ Высокая точность
+ Почти не переобучается
— Трудно параллелизовать
— Чувствителен к шуму

Bagging

Bagging = Bootstrap + Aggregating

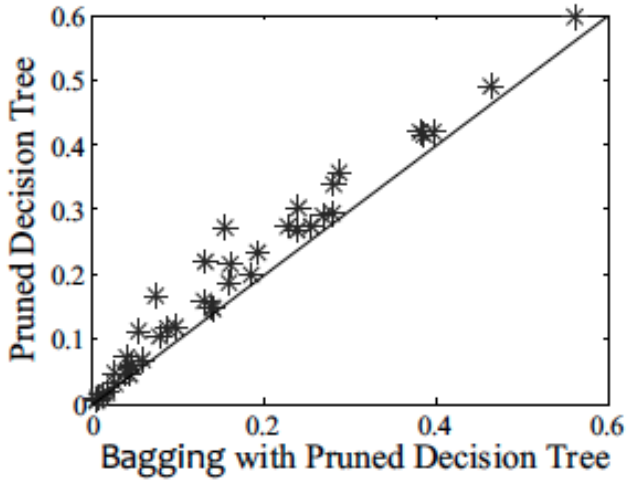
Идея метода

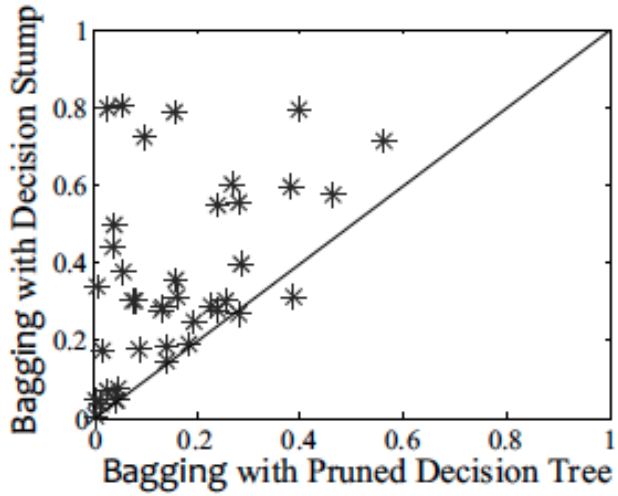
Обучить несколько независимых моделей на основании случайно выбранных (bootstrap) подмножеств объектов из обучающей выборки. Классификация производится на основании результата голосования (aggregating) моделей.

Bagging

bagging($\mathbf{X}$, Y, T):
  for $t = 1, \ldots, T$:
    генерируем bootstrap-распределение $\mathcal{D}_{bs}$
    обучаем модель $h_t(\mathbf{x}) = \mathcal{L}(\mathbf{X}, Y)$,
      принимая во внимание распределение $\mathcal{D}_{bs}$
  return $H(\mathbf{x}) = \arg\max_{y \in Y} \sum_{t=1}^{T} \mathbf{I}(h_t(\mathbf{x}) = y)$

Bagging: тесты

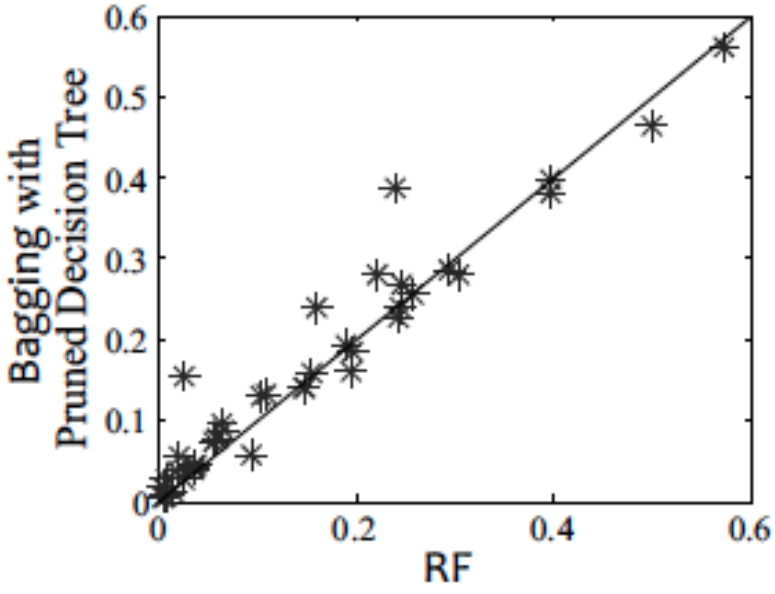


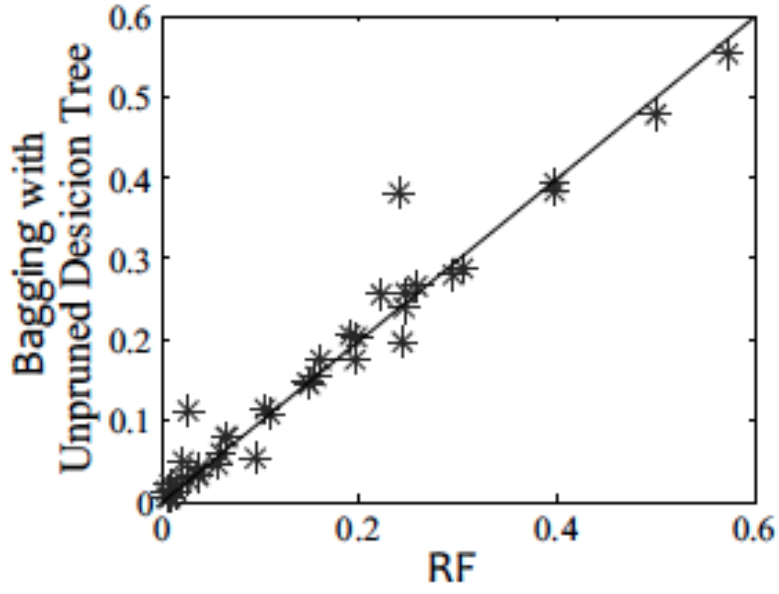


Random Forest

random_tree($\mathbf{X}$, Y, K):
  N - узел дерева для $\mathbf{X}$
  if все $\mathbf{x} \in \mathbf{X}$ одного класса:
    return N
  $\mathcal{F}$ - случайно выбираем K признаков
  $f \in \mathcal{F}$ - признак, наилучшим образом разделяющий $\mathbf{X}$
  $N_l = random\_tree(X_l^f, Y_l^f, K)$
  $N_r = random\_tree(X_r^f, Y_r^f, K)$
  добавляем N_l и N_r как детей к N
  return N

Random Forest: тесты

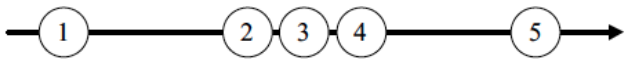




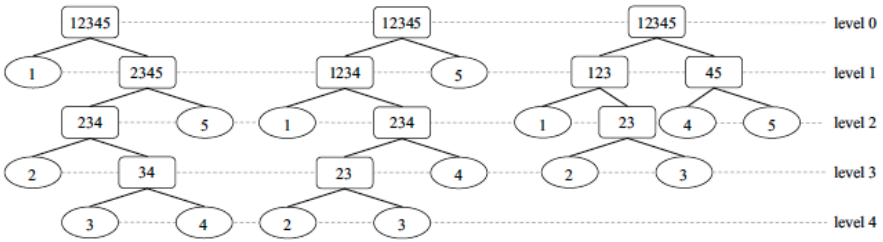
Модификации Random Forest

- VR-Tree
  В каждом узле с вероятностью α просиходит случайный выбор признака
- Density estimation
  Польностью случайное дерево
- Anomaly Detection
  Польностью случайное дерево с ограничением по глубине
  SCiForest

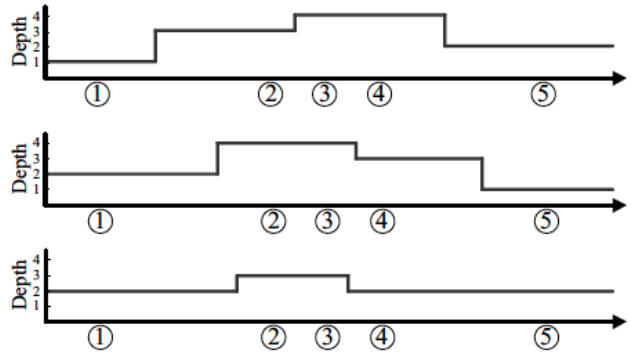
Density Estimation



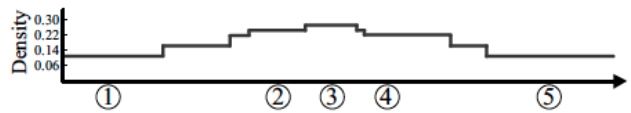
(a) Five one-dimensional data points



(b) Three completely random trees



(c) The depth of leaves on the one-dimensional data, each corresponding to a random tree in the sub-figure (b)

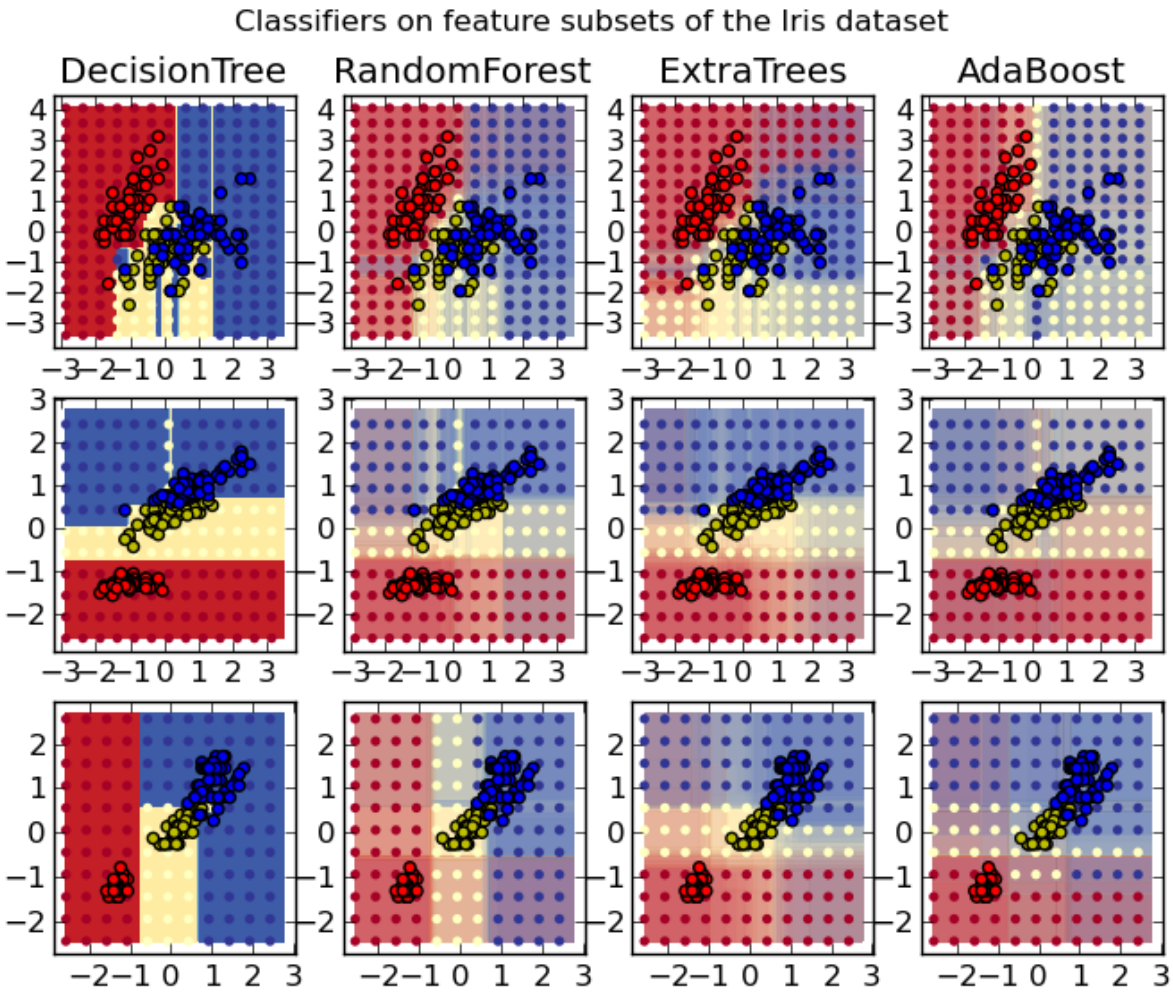


(d) The density estimation result

Random Forest. Итоги

+ Высокая точность
+ Мало переобучения
+ Легко параллелится
— Медленный без параллелизма

Сравнение методов



Зоопарк комбинаций

Задача регрессии

- averaging
- weighted averaging

Задача классификации

- majority voting
- plurality voting
- weighted voting
- soft voting

Кроме того: stacking

Выводы

Задача. Распознавание цифр

Дана обучающая выборка с картинками 8x8, на каждой из картинок изображена рукописная цифра.

```
$ python digits.py -s 25
```

1. для алгоритма AdaBoost построить график зависимости *train_error* и *test_error* от T
2. для алгоритма RandomForest построить график зависимости *train_error* и *test_error* от размера леса
3. реализовать простейший голосующий ансамбль и исследовать зависимость его точности от вида и количества базовых моделей