

## Соревнование Rossmann Store Sales

Остапец Андрей  
(aostapec@mail.ru)

15 октября 2015 г.

# Содержание

- 1 Конкурсное задание
  - Условие
  - Данные
- 2 Краткий обзор нескольких алгоритмов машинного обучения

## Условие

**Задача:** предсказать дневные продажи лекарств для 1115 магазинов за 6-недельный период времени

**Признаки:**

- Описание магазина (идентификатор, тип)
- Информация о праздничных днях
- Описание ассортимента магазина
- Информация о ближайших конкурирующих магазинах
- Информация о промо-акциях

**Целевая переменная:** количество проданных товаров

**Метрика качества:** Root Mean Square Percentage Error (RMSPE)

# RMSPE

$$RMSPE = \sqrt{\frac{1}{n} \sum_{i=1}^n \left( \frac{y_i - \hat{y}_i}{y_i} \right)^2},$$

- $n$  - число объектов
- $y_i$  - истинное значение продаж в конкретном магазине в конкретный день
- $\hat{y}_i$  - предсказанное значение продаж в конкретном магазине в конкретный день

# RMSPE

$$RMSPE = \sqrt{\frac{1}{n} \sum_{i=1}^n \left( \frac{y_i - \hat{y}_i}{y_i} \right)^2},$$

- $n$  - число объектов
- $y_i$  - истинное значение продаж в конкретном магазине в конкретный день
- $\hat{y}_i$  - предсказанное значение продаж в конкретном магазине в конкретный день

Если  $y_i = 0$ , то данный объект не участвует в оценке.

## Leaderboard и форум

Таблица результатов:

- 1 Результаты в таблице вычисляются по 39% тестовой выборке
- 2 Опасно настраиваться только на leaderboard, появляется риск переобучения








На странице конкурса есть форум.

- Можно делиться интересными идеями или кодом
- Можно воспользоваться чужими идеями или кодом

# Scripts


**ROSSMANN** Rossmann Store Sales \$35,000 • 807 teams  
Wed 30 Sep 2015 Merger and 1st Submission Deadline  
Mon 14 Dec 2015 (2 months to go)

Hottest All Languages All Output Types






-  Exploratory Analysis Rossmann  
last run 4 days ago by [the1a](#)  
6 comments · 3 forks · 1207 views · Markdown
-  H2O Random Forest Example (0.1578)  
last run 3 days ago by [miandry](#)  
16 comments · 31 forks · 4811 views · R · Forked Script · +123 / -25 / -32
-  Random Forest Example (0.12579)  
last run 5 days ago by [Michael Pawlus](#)  
8 comments · 29 forks · 4398 views · R
-  predict\_sales\_with\_pandas.py  
last run 4 days ago by [dune\\_dweller](#)  
13 comments · 7 forks · 1684 views · Python · Forked Script · +4 / -6 / -9
-  XGBoost in python with RMSPE  
last run 4 days ago by [Paco](#)  
12 comments · 10 forks · 1491 views · Python
-  predict\_sales\_with\_pandas.py  
last run 7 days ago by [Roelint](#)  
2 comments · 7 forks · 1782 views · Python
-  XGB W/ RMSPE Eval  
last run 8 days ago by [the1a](#)

[New Script](#)  
[Feedback](#)

**Explore**  
Run one-click analyses, no local environment or data download needed



**Most Recent Scripts**

-  RossmannMain  
4 minutes ago · 0 votes
-  xGB\_Rossmann  
16 minutes ago · 0 votes
-  xGB\_Rossmann  
1 hour ago · 0 votes
-  xgb-1  
2 hours ago · 0 votes
-  XG boost lb 11

# Данные

- 1115 различных магазинов
- Обучение: период времени с 2013-01-01 по 2015-07-31
- Тест: период времени с 2015-08-01 по 2015-09-17
- Обучение: 1,017,209 объектов
- Тест: 41,088 объектов



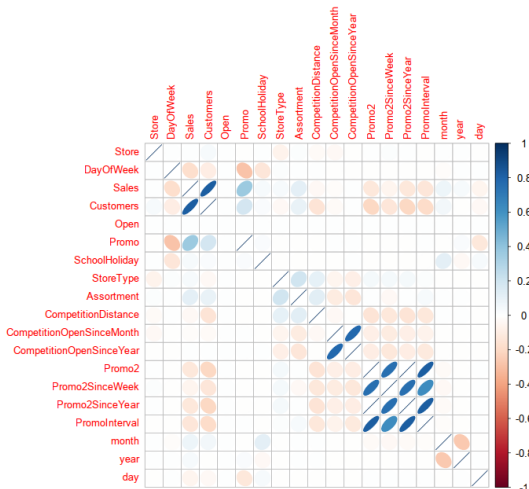
## Признаки

- Store - уникальный идентификатор магазина
- Date - день, в который производятся продажи
- Sales - число проданных товаров за день (целевая переменная)
- Customers - количество покупателей в данный день (дано только для тренировочной выборки)
- Open - открыт магазин или нет: 0 = закрыт, 1 = открыт
- StateHoliday - наличие государственного праздника в этот день (a = public holiday, b = Easter holiday, c = Christmas, 0 = None)
- SchoolHoliday - наличие школьных каникул
- StoreType - тип магазина (a, b, c, d)
- Assortment - выбор товаров в магазине (a = basic, b = extra, c = extended)

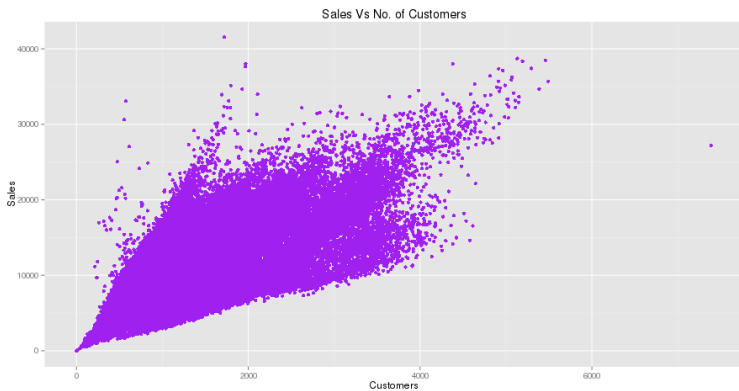
## Признаки

- CompetitionDistance - расстояние до ближайшего конкурента
- CompetitionOpenSince[Month/Year] - когда был открыт ближайший конкурент?
- Promo - индикатор промо-акции в этот день
- Promo2 - наличие Promo2 акции (более продолжительная и действующая на ряд магазинов)
- Promo2Since[Year/Week] - когда магазин начал участвовать в Promo2 акции?
- PromoInterval - в какие месяцы каждый год в этом магазине идет Promo2?

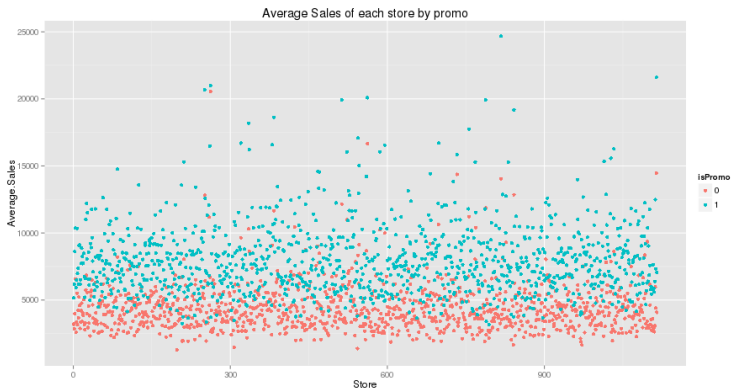
# Корреляционная матрица признаков



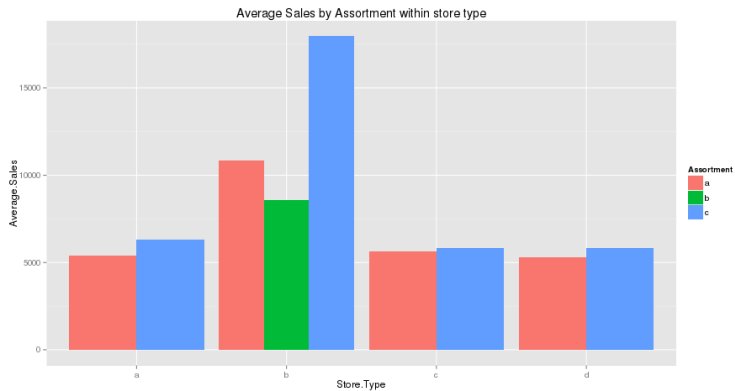
# Продажи и клиенты



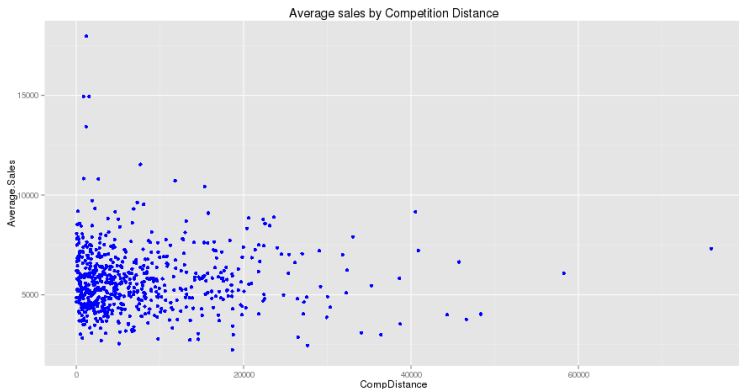
# Промо акции



# Тип и ассортимент

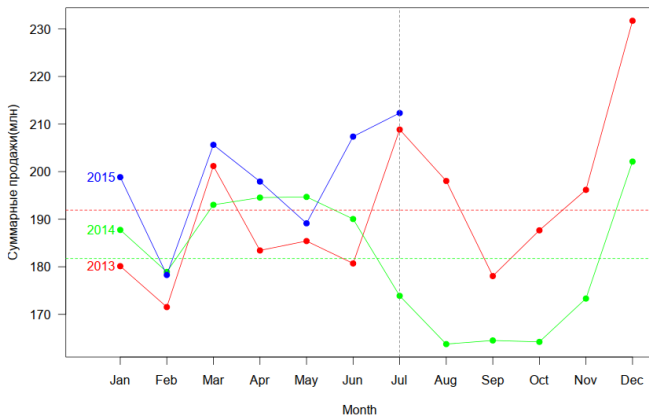


# Расстояние до ближайшего конкурента



# Визуализация продаж

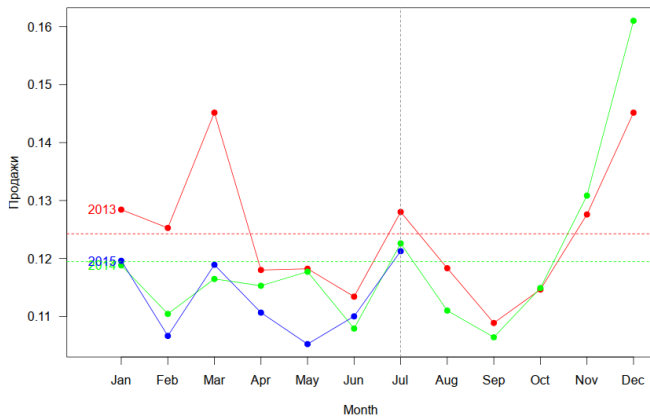
Продажи по годам





# Визуализация продаж

Продажи по годам (Store 1)



# Основные алгоритмы машинного обучения

Краткий обзор нескольких алгоритмов машинного обучения

# Naive Bayes

**Что он делает?** Наивный байесовский классификатор – это семейство алгоритмов классификации, которые принимают одно допущение:

Каждый параметр классифицируемых данных рассматривается независимо от других параметров класса.

**Что означает слово «независимо»?** 2 параметра называются независимыми, когда значение одного параметра не оказывает влияния на второй.

**Почему метод называется наивным?** Предположение, что все параметры набора данных независимы – это довольно наивное предположение. Обычно так не бывает (пульс, уровень холестерина, вес, рост и почтовый индекс)

# Naive Bayes

$$P(\text{Class } A | \text{Feature 1, Feature 2}) = \frac{P(\text{Feature 1} | \text{Class } A) \cdot P(\text{Feature 2} | \text{Class } A) \cdot P(\text{Class } A)}{P(\text{Feature 1}) \cdot P(\text{Feature 2})}$$

# Naive Bayes

Таблица : Описание фруктов

Class	Long	Sweet	Yellow	Total
Banana	400	350	450	500
Orange	0	150	300	300
Other	100	150	50	300
Total	500	650	800	1000

$P(\text{Banana} | \text{Long}, \text{Sweet}, \text{Yellow}) = ?$

- $P(\text{Long} | \text{Banana}) = 400/500 = 0.8$
- $P(\text{Sweet} | \text{Banana}) = 350/500 = 0.7$
- $P(\text{Yellow} | \text{Banana}) = 450/500 = 0.9$
- $P(\text{Banana}) = 500/1000 = 0.5$

## Naive Bayes

- Числитель для  $P(\text{Banana}|\text{Long, Sweet, Yellow}) = 0.8 \cdot 0.7 \cdot 0.9 \cdot 0.5 = 0.252$
- Числитель для  $P(\text{Orange}|\text{Long, Sweet, Yellow}) = 0$
- Числитель для  $P(\text{Other}|\text{Long, Sweet, Yellow}) = 0.01875$

Наивный байесовский алгоритм классифицирует этот длинный, сладкий и желтый фрукт как банан

## Код на R

```
library(e1071)

x <- cbind(xtrain , ytrain)

# Fitting model
fit <- naiveBayes(ytrain ~ . , data = x)
summary(fit)

#Predict Output
predicted <- predict(fit , xtest)
```

## В данной задаче...

Классификация:

- Выросли или упали продажи по сравнению с прошлым годом?
- Будут ли хоть продажи выше порога(например, 0)?
- ...



## В данной задаче...

Классификация:

- Выросли или упали продажи по сравнению с прошлым годом?
- Будут ли хоть продажи выше порога(например, 0)?
- ...

# kNN

**Что он делает?** Алгоритм kNN (k-Nearest Neighbors) не строит в явном виде никакую классификационную модель. Вместо этого он просто сохраняет размеченные тренировочные данные. Когда появляется новые неразмеченные данные, kNN проходит по 2 базовым шагам:

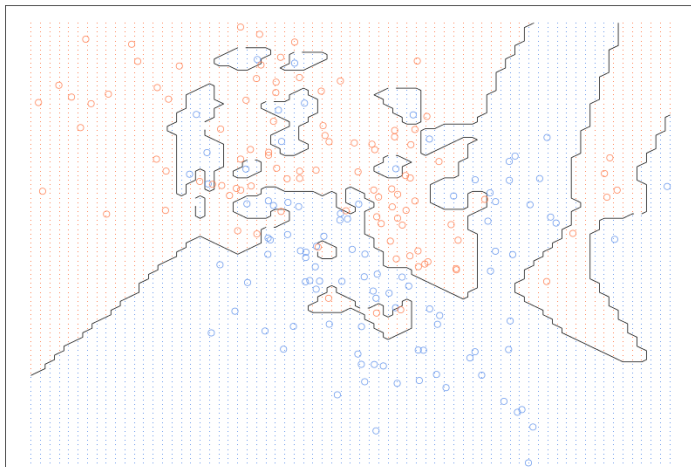
- Сначала он ищет  $k$  ближайших соседей
- Затем, используя данные о классах этих соседей, kNN решает, как лучше классифицировать новые данные.

# kNN

Если мера сходства введена достаточно удачно, то оказывается, что *схожим объектам*, как правило, соответствуют *схожие ответы*

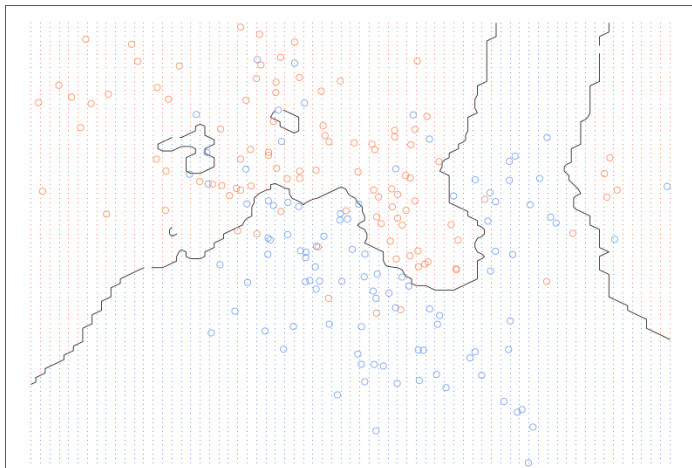
# 1-NN

1-nearest neighbour



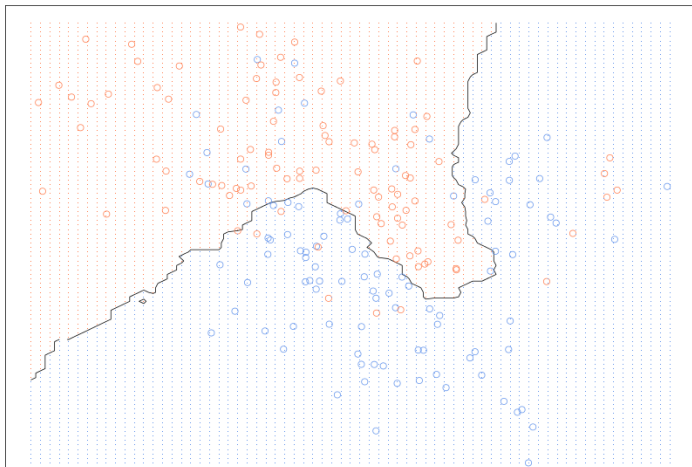
# 5-NN

5-nearest neighbours



# 15-NN

15-nearest neighbours



# Метрика

Вопрос: какие примеры метрик Вы знаете?

## Определение класса

Как определить класс классифицируемого объекта?

- Принять за правильное решение простое большинство. К какому классу относится наиболее количество соседей, туда и определяют точку данных.
- Раздать веса в зависимости от расстояния до объекта. При увеличении дистанции вес становится все меньше и меньше.



## Плюсы

Плюсы:

- Легок в понимании
- Легко реализуем
- При «хорошем» выборе дистанционной метрики, kNN может показывать достаточно точные результаты

## Минусы

Минусы:

- Ресурсозатратный на большом наборе данных
- Нужна нормализация признаков. Признаки с большим количеством значений могут оказывать влияние на дистанционную метрику, по отношению к признакам с меньшим количеством значений
- Выбор правильной дистанционной метрики очень важен для точности kNN

## Код на R

```
library(knn)

x <- cbind(x_train, y_train)

# Fitting model
fit <- knn(y_train ~ ., data = x, k=5)
summary(fit)

#Predict Output
predicted <- predict(fit, x_test)
```

# Линейная регрессия

Что алгоритм делает? Обучается линейная модель:

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^p x_j w_j = \mathbf{x}^T \mathbf{w}, \mathbf{x} = (x_1, \dots, x_p)$$

По вектору входов  $\mathbf{x}^T = (x_1, \dots, x_p)$  мы предсказываем выход  $y$ :

$$\hat{y}(\mathbf{x}) = \hat{w}_0 + \sum_{j=1}^p x_j \hat{w}_j = \mathbf{x}^T \hat{\mathbf{w}}$$

# Линейная регрессия

- Как найти оптимальные параметры  $\hat{\mathbf{w}}$  по тренировочным данным вида  $(x_i, y_i)_{i=1}^N$ ?
- Метод наименьших квадратов: будем минимизировать

$$RSS(\mathbf{w}) = \sum_{i=1}^N (y_i - x_i^T \mathbf{w})^2$$

- Как минимизировать?

## Метод наименьших квадратов

- Можно на самом деле решить задачу точно – записать как

$$RSS(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w}),$$

где  $\mathbf{X}$  -матрица размера  $N \times p$ , продифференцировать по  $\mathbf{w}$ , получится

$$\hat{\mathbf{w}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y},$$

если матрица  $\mathbf{X}^T \mathbf{X}$  невырожденная.

- Замечание:  $(\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T$  называется *псевдообратной матрицей Мура–Пенроуза (Moore–Penrose pseudo-inverse)* матрицы  $\mathbf{X}$ ; это обобщение понятия обратной матрицы на неквадратные матрицы.

## Код на R

```
x <- cbind(x_train, y_train)

# Train the model using the training set
linear <- lm(y_train ~ ., data = x)
summary(linear)

#Predict Output
predicted <- predict(linear, x_test)
```

## Решающее дерево

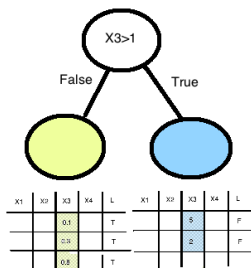
**Что алгоритм делает?** Строит дерево, где каждая вершина представляет собой точку разбиения данных. Вершина определяет некоторый простой критерий по которому мы делим данные на части.

Таким образом, в каждой вершине дерева мы разбиваем данные на несколько частей согласно значению одного из параметров. Дальше для каждой части мы повторяем операцию и в итоге получаем дерево.



# Решающее дерево

X1	X2	X3	X4	L
		0.1		T
		5		F
		0.3		T
		2		F
		0.8		T



## Вопросы

- Как выбирается параметр по которому происходит разбиение?
- Как выбирается пороговое значения параметра?
- Когда алгоритм должен остановиться?

## Код на R

```
library(rpart)

x <- cbind(x_train, y_train)

# grow tree
fit <- rpart(y_train ~ ., data = x)
summary(fit)

#Predict Output
predicted <- predict(fit, x_test)
```

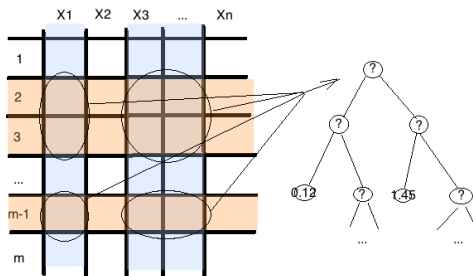
## Случайный лес

**Что алгоритм делает?** Random forest (случайный лес) - ансамбль решающих деревьев.

Особенности:

- Нечувствительность к любым монотонным преобразованиям значений признаков
- Существует методы оценивания значимости (Feature Importance) отдельных признаков в модели
- Высокая параллелизуемость и масштабируемость

# Случайный лес

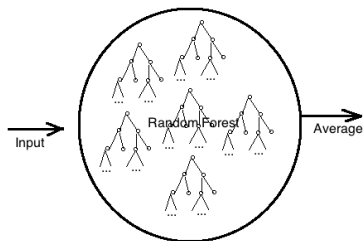


## Алгоритм обучения

Пусть обучающая выборка состоит из  $N$  примеров, размерность пространства признаков равна  $M$ , и задан параметр  $m$ . Все деревья комитета строятся независимо друг от друга по следующей процедуре:

- Генерируется случайная подвыборка с повторением размером  $N$  из обучающей выборки. (Сколько уникальных объектов туда попадут?)
- Строится решающее дерево, классифицирующее примеры данной подвыборки, причём в ходе создания очередного узла дерева выбирается признак, на основе которого производится разбиение, не из всех  $M$  признаков, а лишь из  $m$  случайно выбранных.
- Дерево строится до полного исчерпания подвыборки.

## Получение ответа



## Код на R

```
library(randomForest)

x <- cbind(x_train, y_train)

# Fitting model
fit <- randomForest(Species ~ ., x, ntree=500)
summary(fit)

#Predict Output
predicted <- predict(fit, x_test)
```

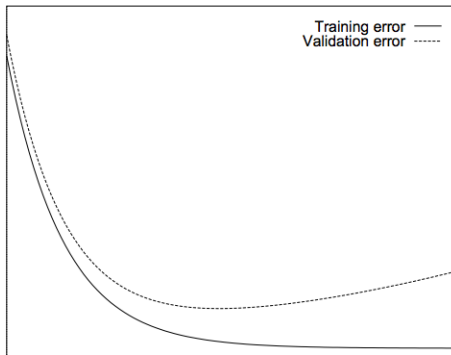


## Бустинг

**Что алгоритм делает?** Бустинг - это процедура последовательного построения композиции алгоритмов машинного обучения, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов. Бустинг представляет собой жадный алгоритм построения композиции алгоритмов.

**Вопрос:** возможно ли, имея множество плохих (незначительно отличающихся от случайных) алгоритмов обучения, получить хороший?

## Зависимость от числа деревьев



## Композиция алгоритмов

- бэггинг (bagging) - усреднение нескольких однотипных моделей
- бустинг (boosting) - построение цепочки моделей, дополняющих друг друга
- блэндинг (blending) - смешивание классификаторов (как правило, линейная комбинация)
- стэкинг (stacking) - построение классификатора над другими классификаторами

## Задание

**Дедлайн:** 28 октября 23:59

**Задание:** Попробовать по крайней мере 3 различных алгоритма машинного обучения и по крайней мере 3 различных признаков пространства. Прислать отчет на почту [aostapec@mail.ru](mailto:aostapec@mail.ru).