

# Кейс стади 1



Задача классификации. Поиск причины дефекта в металле. Тесты на согласие: поиск распределения. Проблема мультиколлинеарности. Проблема несбалансированности классов. Пример решения задачи классификации с помощью RandomForest. Метрики классификации: precision, recall, F1. Принцип минимальных компонент, PCA. Кросс-валидация. ROC-кривая. Предсказание цены на нефть за определённый период.

**Даниил Корбут**

Специалист по Анализу Данных



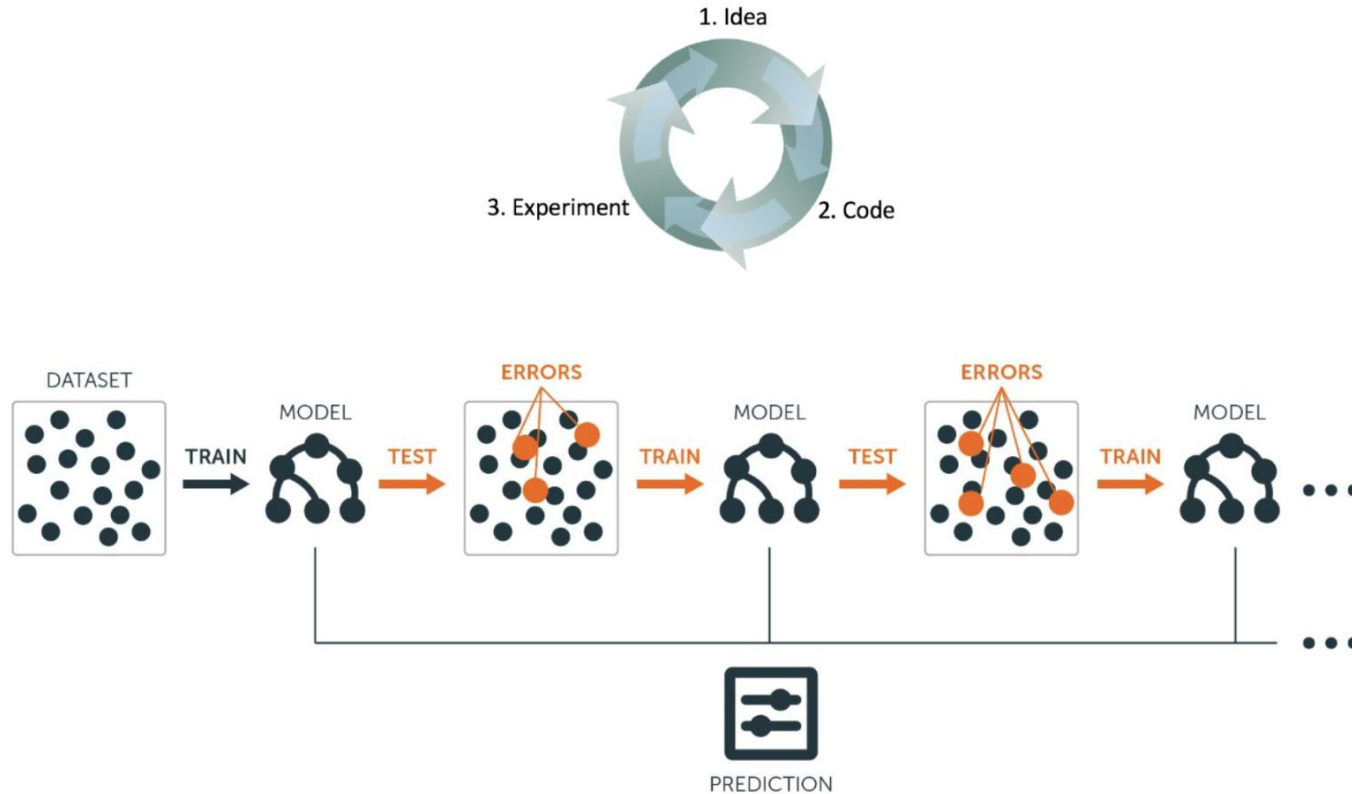
**Даниил Корбут**  
DL Researcher  
Insilico Medicine, Inc

Окончил бакалавриат ФИВТ  
МФТИ (Анализ данных) в 2018г  
Учусь на 2-м курсе  
магистратуры ФИВТ МФТИ  
Работал в Statsbot и Яндекс.  
Алиса.  
Сейчас в Insilico Medicine, Inc,  
занимаюсь генерацией  
активных молекул и  
исследованиями старения с  
помощью DL.

# Классическое Обучение

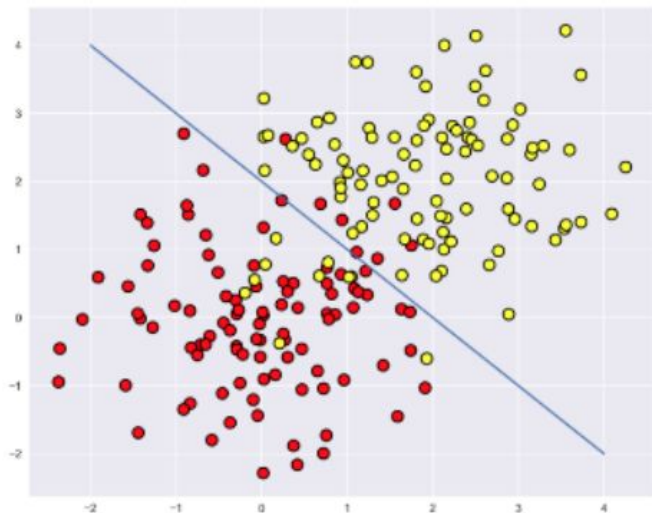


# Процесс разработки моделей



## Классификация

*Множество допустимых ответов конечно. Их называют метками классов (class label). Класс — это множество всех объектов с данным значением метки.*

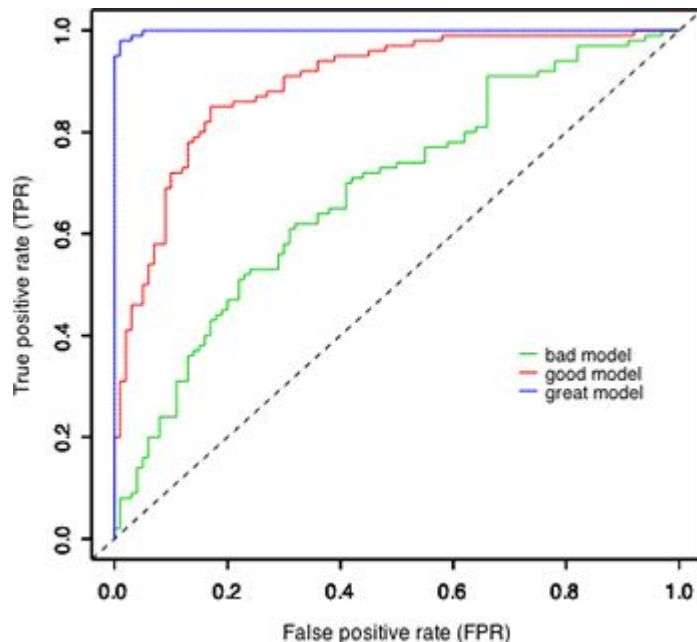


# Метрики классификации

		predicted condition		
		prediction positive	prediction negative	
true condition	total population			<b>Sensitivity</b>
	condition positive	True Positive (TP)	False Negative (FN) (Type II error)	<b>Recall =</b> $\frac{\sum TP}{\sum \text{condition positive}}$
	condition negative	False Positive (FP) (Type I error)	True Negative (TN)	<b>Specificity =</b> $\frac{\sum TN}{\sum \text{condition negative}}$
		<b>Precision =</b> $\frac{\sum TP}{\sum \text{prediction positive}}$		<b>F1 Score =</b> $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
		<b>Accuracy =</b> $\frac{\sum TP + \sum TN}{\sum \text{total population}}$		

# Метрики классификации

## Roc-AUC



## Log-Loss

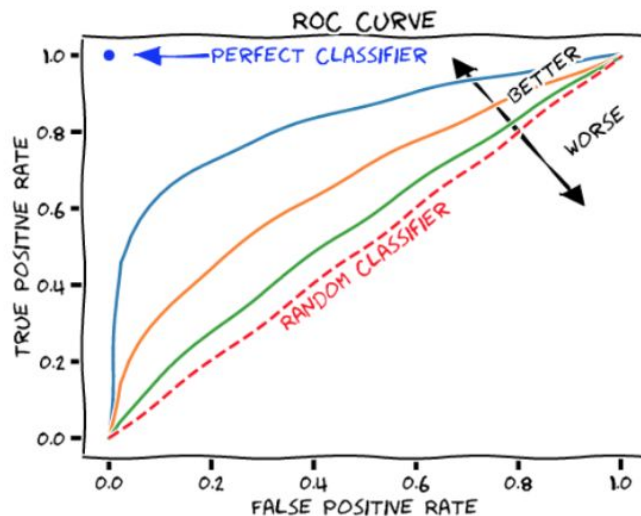
$$-(y \log(p) + (1 - y) \log(1 - p))$$

predicted→ real↓	Class_pos	Class_neg
Class_pos	TP	FN
Class_neg	FP	TN

$$\text{TPR (sensitivity)} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{FPR (1-specificity)} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

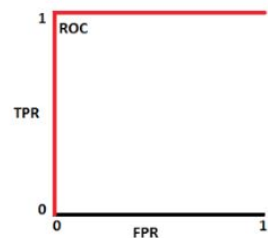
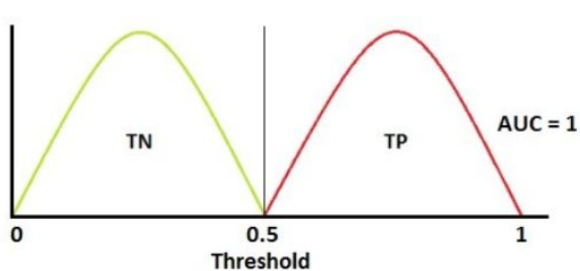
# Roc-curve



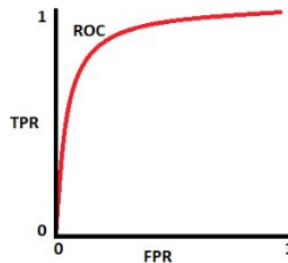
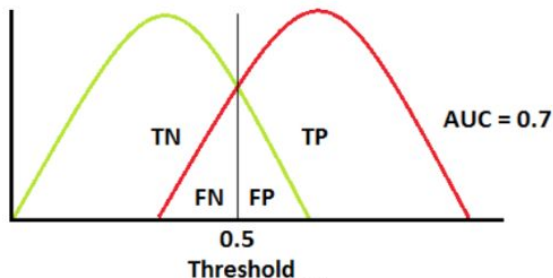
Позволяет определить порог,  
при котором мы будем отделять один класс от другого



# Roc-curve



Идеальная модель — порог 50%



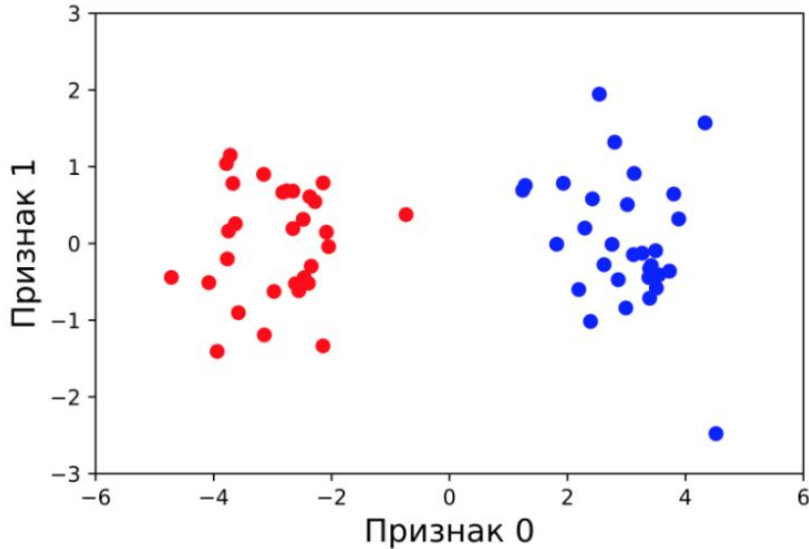
Модель с некоторыми ошибками — порог выбирается в зависимости от допускаемых ошибок

# Давать ли кредит?



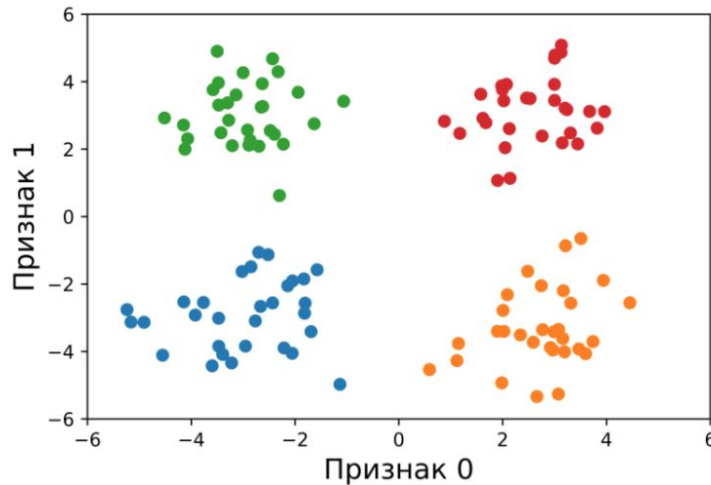
Дерево Решений

# Решающие деревья



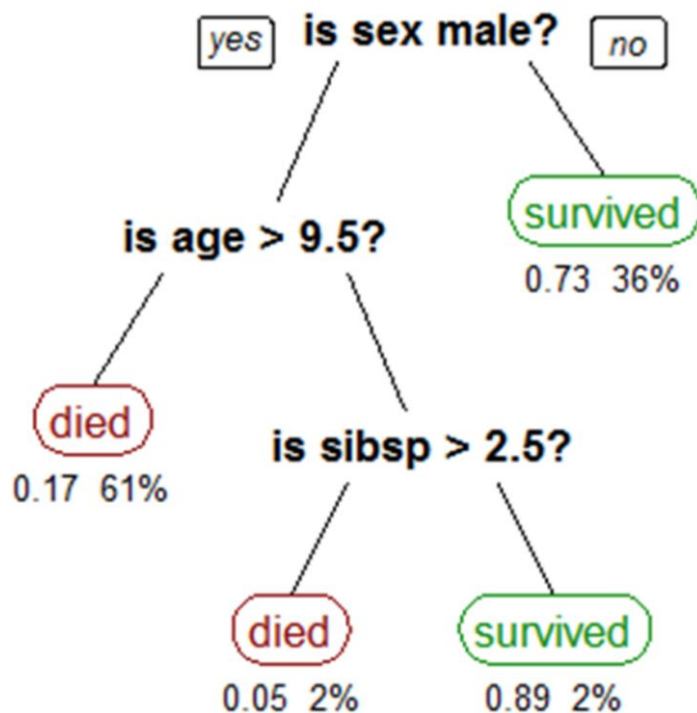
```
def classify(X):  
    if X[0] < 0:  
        return "red"  
    else:  
        return "blue"
```

# Решающие деревья

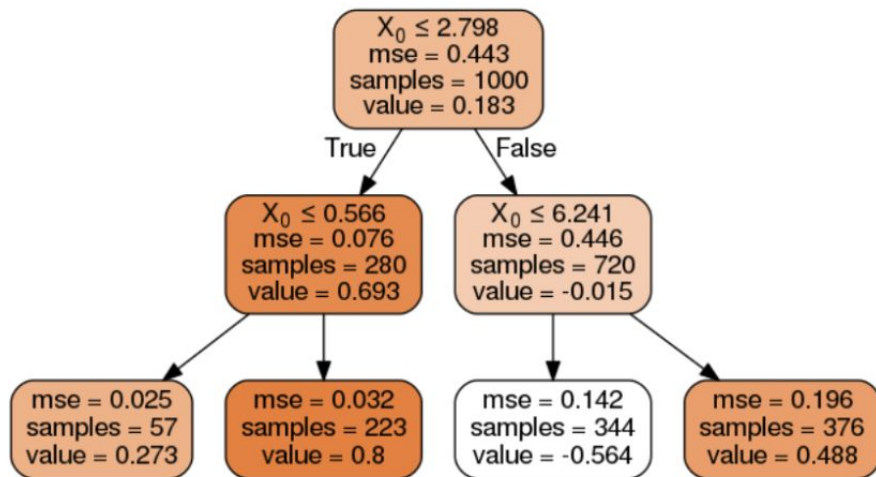
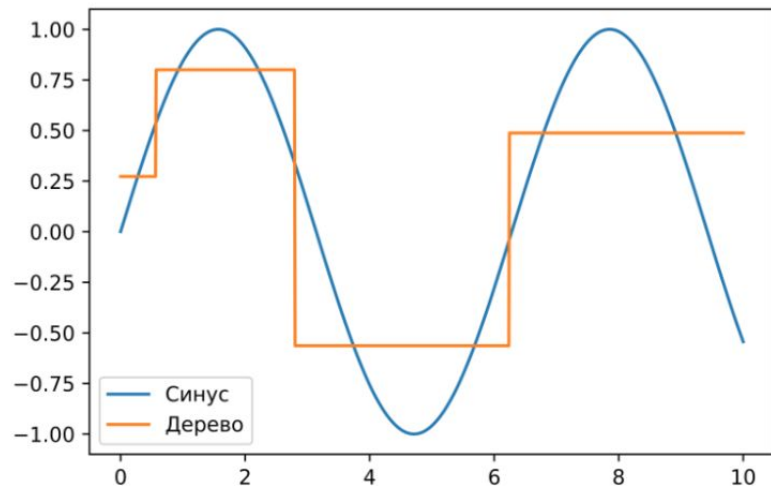


```
def classify(X):  
    if X[0] < 0:  
        if X[1] < 0:  
            return "blue"  
        else:  
            return "green"  
    else:  
        if X[1] > 0:  
            return "red"  
        else:  
            return "orange"
```

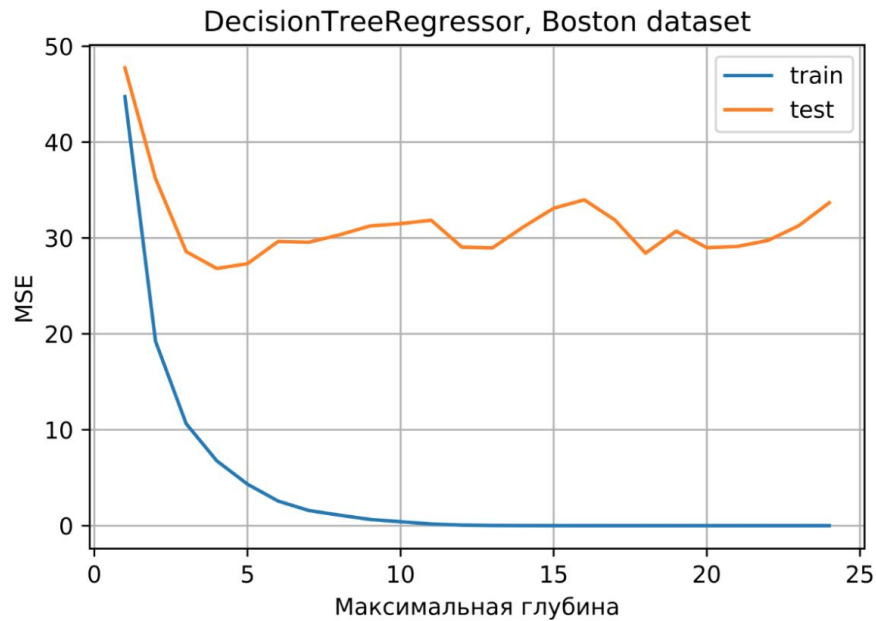
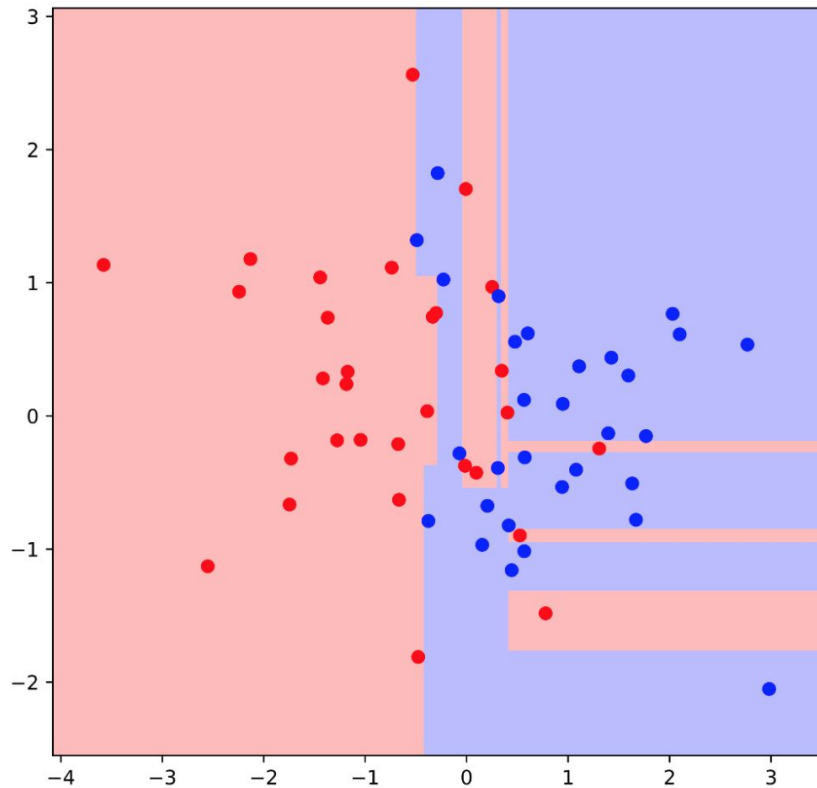
# Решающие деревья



# Решающие деревья



# Решающие деревья



# Решающие деревья

**Индекс неоднородности** - величина, оценивающая неоднородность выборки

Для задачи регрессии

$$\text{MSE: } H(Y) = \frac{1}{|Y|} \sum_{i=1}^{|Y|} (y_i - \bar{y})^2$$

$$\bar{y} = \frac{1}{|Y|} \sum_{i=1}^{|Y|} y_i$$



## Решающие деревья

Для классификации ( $P_i$  - доля класса  $i$  в  $X$ ,  $L$  - число классов):

- Энтропия:  $H(X) = - \sum_{i=1}^L P_i \log P_i$
- Джини:  $H(X) = \sum_{i=1}^L P_i(1 - P_i)$
- Misclassification:  $H(X) = 1 - \max_{1..L} P_i$

Замечание: нужно считать, что  $P_i \log(P_i) = 0$  при  $P_i = 0$

## Решающие деревья

Уменьшение среднего индекса неоднородности при разбиении:  $I(Q, f, v) = H(Q) - \frac{|L|}{|Q|} H(L) - \frac{|R|}{|Q|} H(R)$   
 $Q$  - выборка,  $f$  - признак,  $v$  - порог,  $L$  и  $R$  - соответствующие им разбиения выборки  $Q$  на две части.

- Будем строить дерево от корня (стартовая вершина) к листьям (вершины, из которых некуда идти)
- В начале в стартовой вершине лежит вся выборка

# Решающие деревья

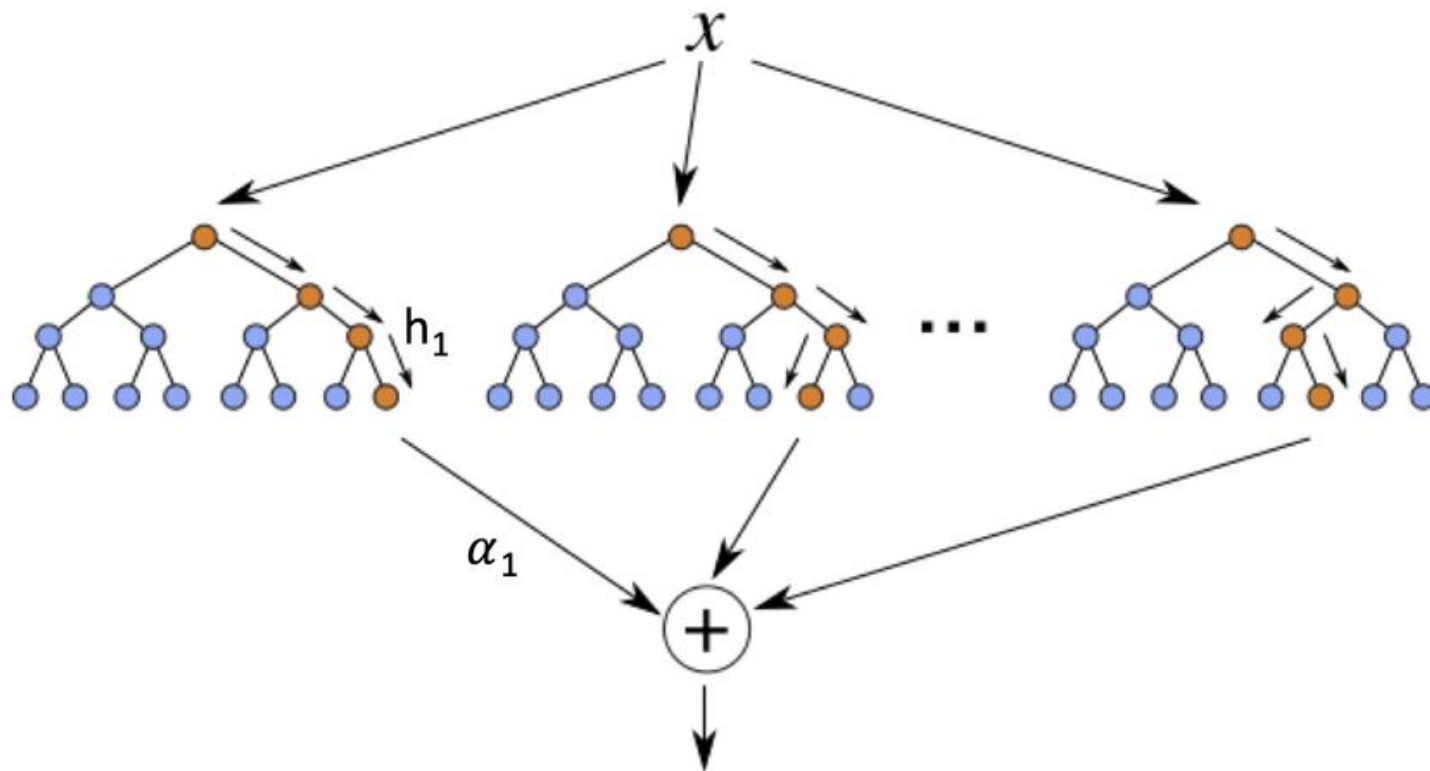
- Если в текущей вершине выполнен критерий останова - ничего не делаем в этой вершине.
- Выбрать  $f$  и  $v$  так, чтобы  $I(Q, f, v)$  было максимально, например, перебрав все признаки и пороги.
- Разделим данную выборку на  $L$  и  $R$  согласно выбранным  $f$  и  $v$ , создадим двух потомков текущей вершины и положим в них  $L$  и  $R$  соответственно.
- Повторим для каждой дочерней вершины.

# Решающие деревья

Таким образом, конкретный метод построения решающего дерева определяется:

1. Видом предикатов в вершинах
2. Критерием информативности
3. Критерием останова
4. Методом обработки пропущенных значений
5. Методом стрижки
6. Работой с категориальными признаками: можно создавать по потомку для каждого значения категориального признака, а можно кодировать средним значением переменной среди элементов данного класса

# Ансамбли деревьев



# Ансамбли деревьев

## **Bootstrap:**

Пусть дана выборка  $X$  из  $n$  объектов. Выберем несколько раз, например  $n$ , равновероятно случайный объект из выборки  $X$  (выбор с повторениями). Выборку составленную из этих объектов назовём bootstrap-выборкой.

## **Пример:**

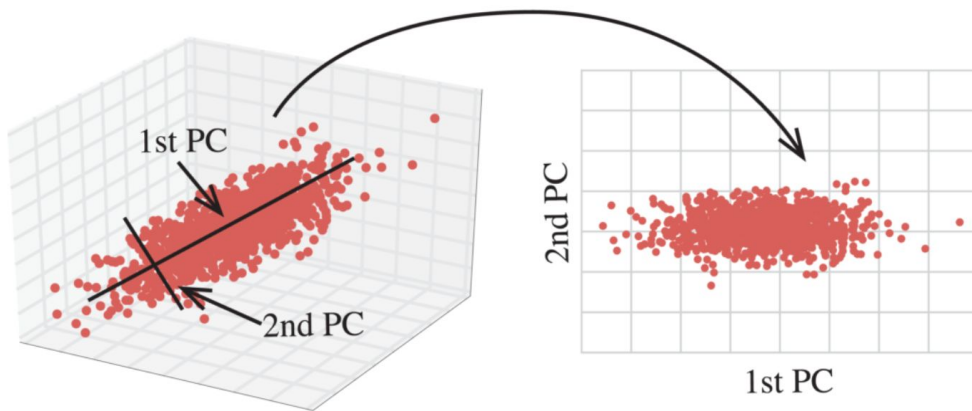
Из  $[1,2,3]$  могут получиться выборки  $[1,2,2]$ ,  $[3,1,2]$ ,  $[3,3,2]$  и тд

# Ансамбли деревьев

Получили итоговый алгоритм для построения **случайного леса (Random Forest)** с  $k$  деревьями:

- Сгенерируем  $k$  bootstrap-подвыборок исходного датасета
- Обучим на каждой выборке своё дерево, но при построении дерева в каждом узле при поиске лучшего разбиения признака используем не все, а  $m$  случайных признаков, и ищем разбиение только по ним ( $m=n^{0.5}$  для классификации,  $m=n/3$  для регрессии)
- Ответ всего алгоритма - класс, за который проголосовало наибольшее количество кандидатов либо среднее значение для классификации и регрессии соответственно

# Dimensionality Reduction: PCA



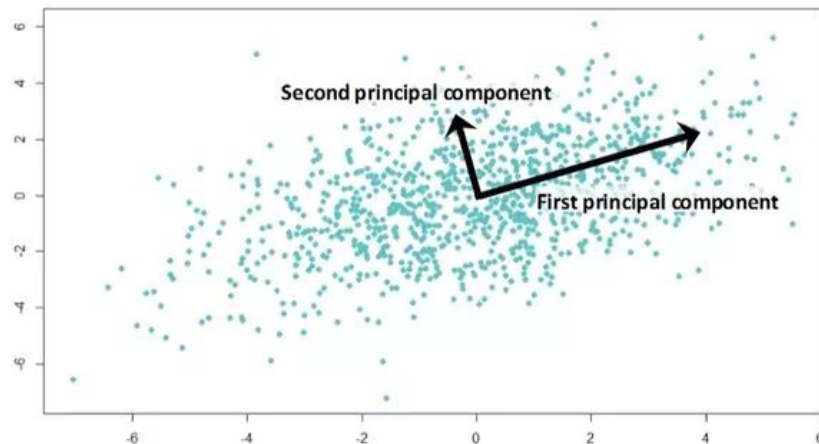
Один из основных способов уменьшить размерность данных, потеряв наименьшее количество информации.





# Dimensionality Reduction: PCA

- 1) Вычисляем матрицу ковариаций признаков
- 2) Находим собственные вектора матрицы ковариаций
- 3) Первые k векторов соответствующих k максимальным собственным значениям - компоненты нашего разложения

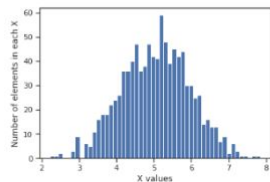


<https://medium.com/@sadatnazrul/the-dos-and-donts-of-principal-component-analysis-7c2e9dc8cc48>

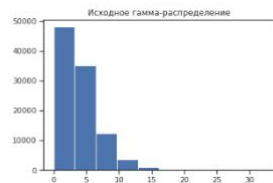
Плюсы: возможность регуляции получаемой размерности (добавлении компонент по одной в зависимости от объяснённой дисперсии); скорость алгоритма; интерпретация

Минусы: линейность, предположение об ортогональности

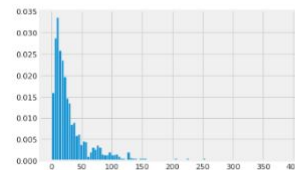
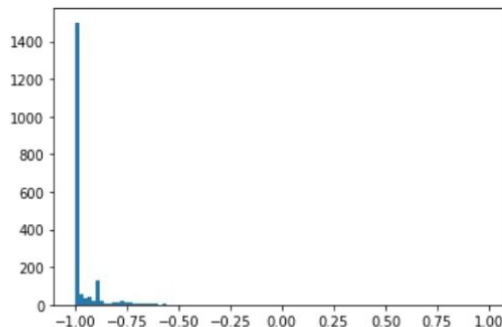
# Тест на согласие: какое это распределение?



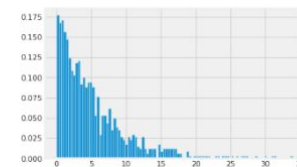
нормальное



гамма



экспоненциальное



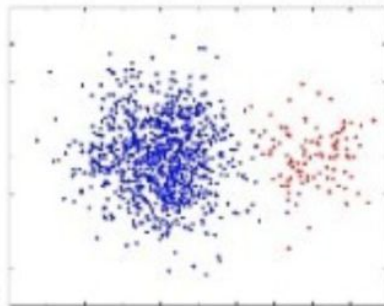
экспоненциальное

<https://mc.ai/15-statistical-hypothesis-tests-in-python-cheat-sheet/>

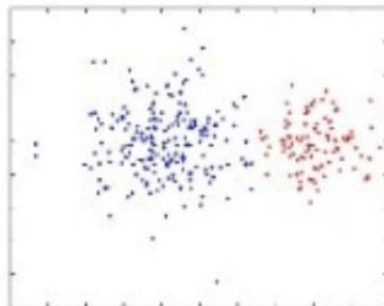
# Проблема несбалансированности классов

**Sampling:** Rebalancing the dataset

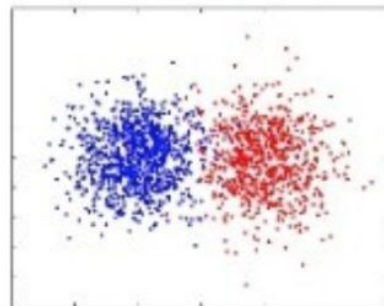
Imbalanced Data



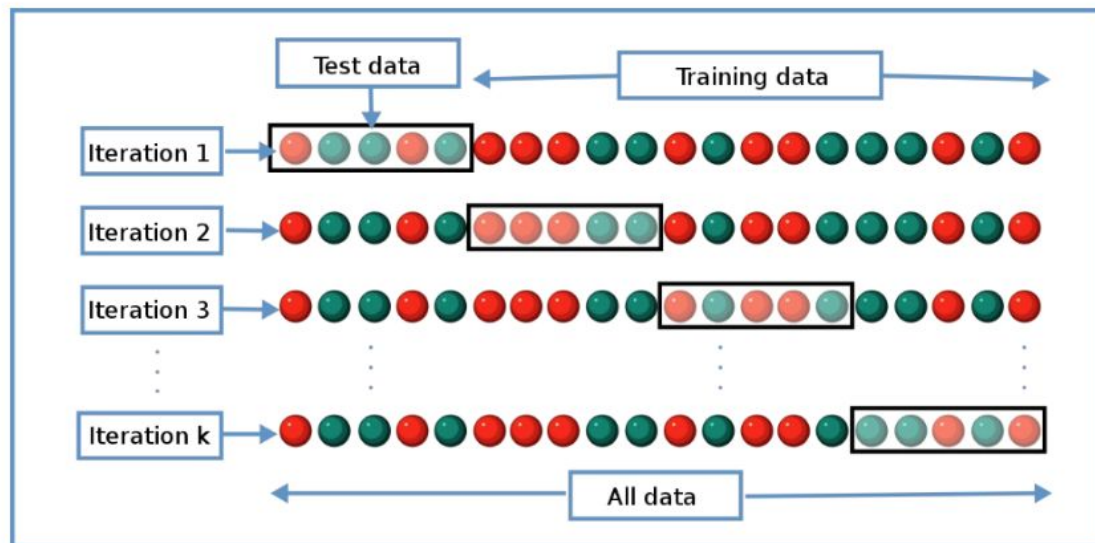
Under-sampling



Over-sampling



# Кросс-валидация



Оцениваем модель на нескольких тестовых данных

**Спасибо за внимание!**