# Машинное обучение в электронной коммерции – практика использования и подводные камни

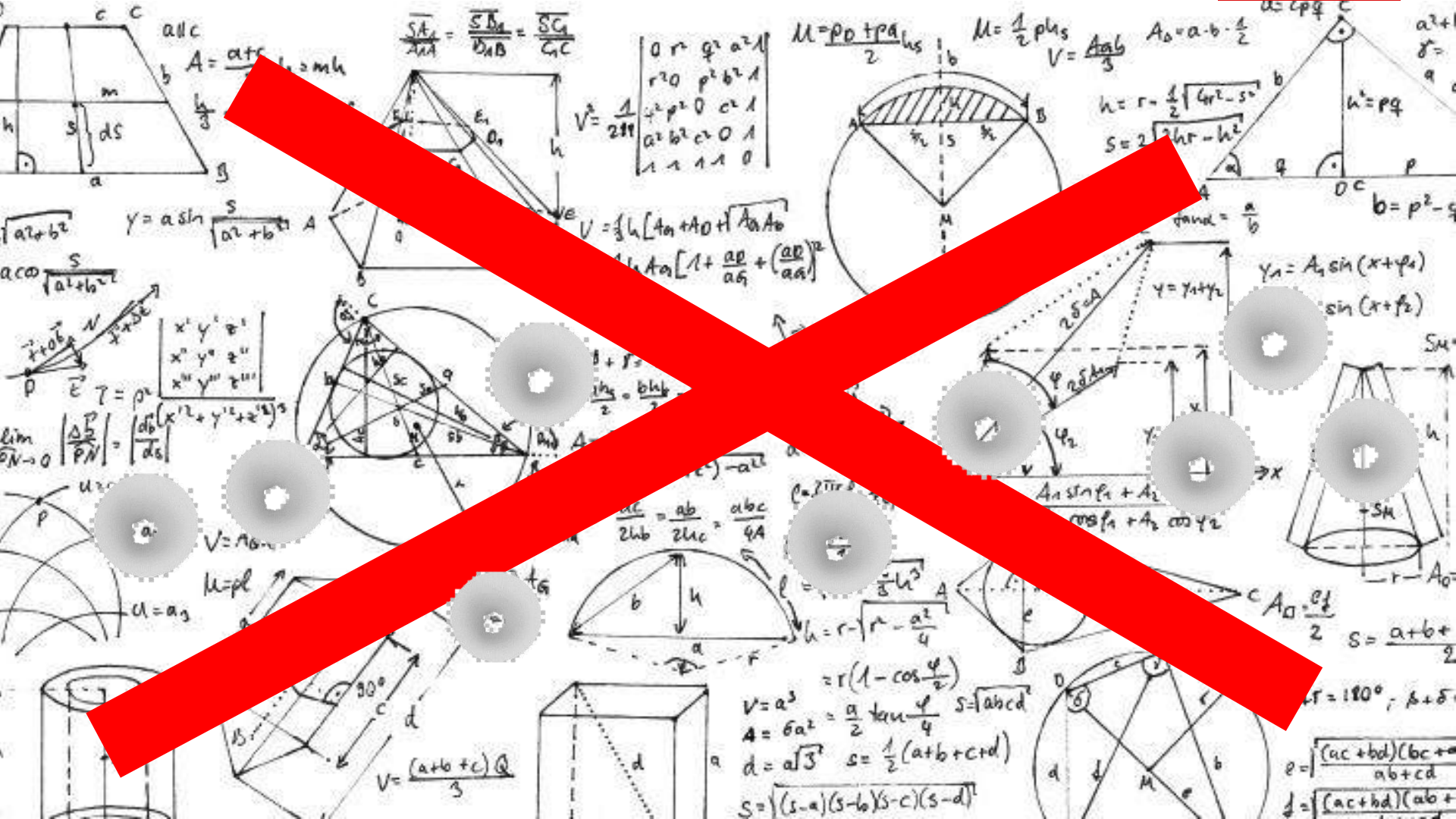
**Александр Сербул**
Руководитель направления


Карл, я открыл
страшную тайну
нейронных сетей
BLOCK
Это очень
интересно, пап!


Большая часть населения земли знают математику на уровне рабов в Египте. А для понимания нейросетей полезно помнить «вышку»

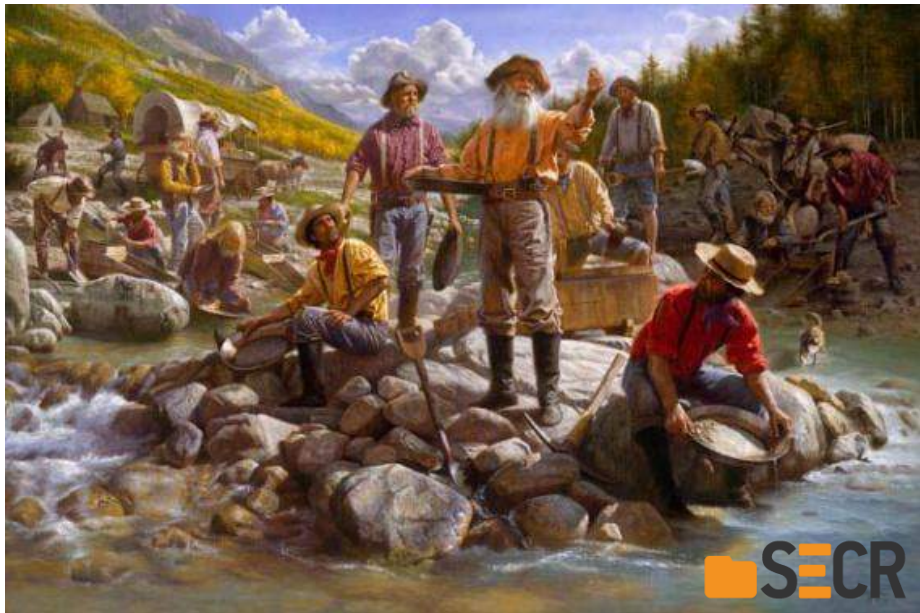
# О чем хочется поговорить

- Ввести в исторический контекст проблемы. Разобраться в причинах.
- Кратко вспомнить нужную теорию
- Перечислить актуальные, интересные бизнес-задачи в электронной коммерции и не только
- Рассмотреть популярные архитектуры нейронок и не только для решения бизнес-задач

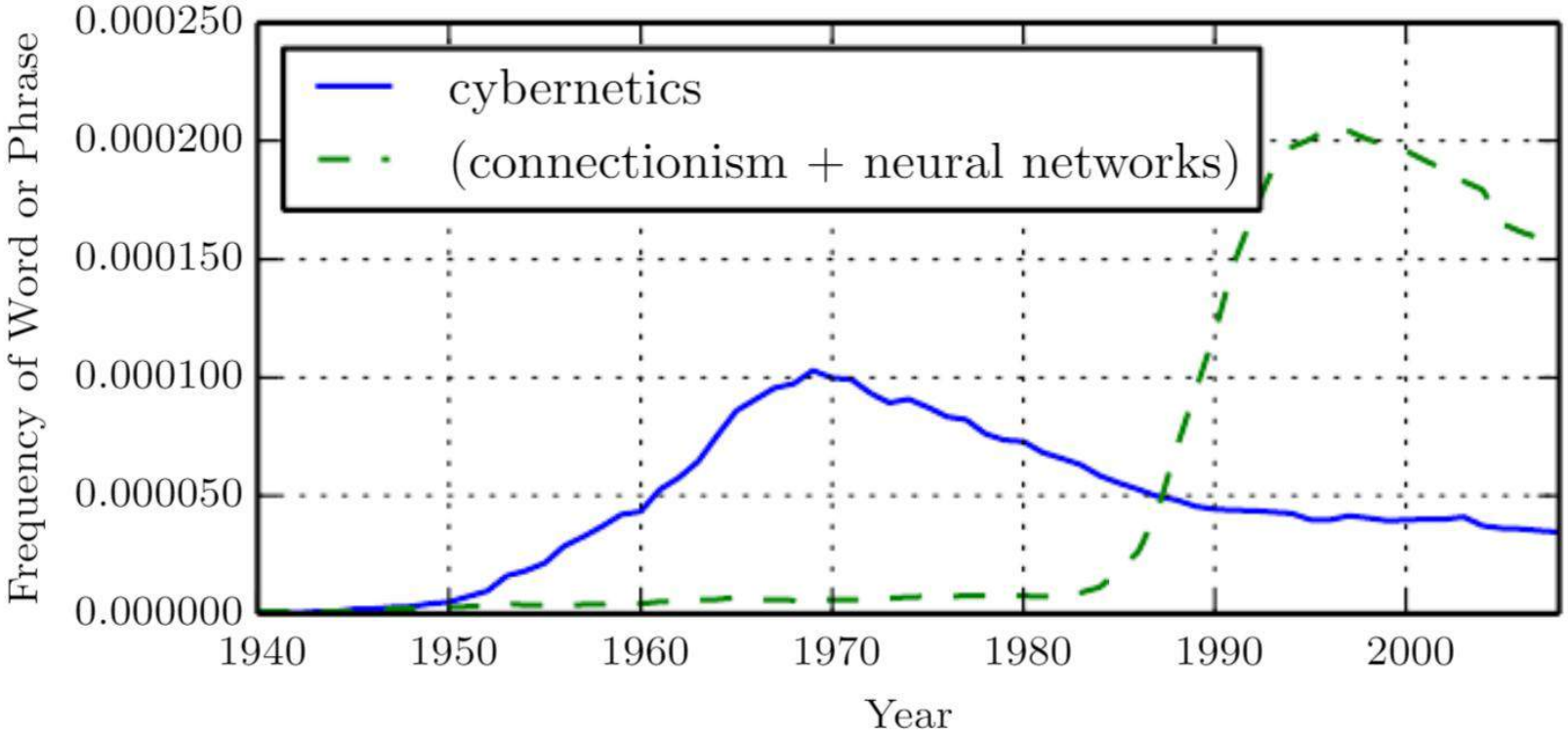


# «Золотая» лихорадка

- Была бигдата
- Теперь нейронки
- Завтра будет вторжение инопланетян ☺
- Что происходит, успеть или забИть?

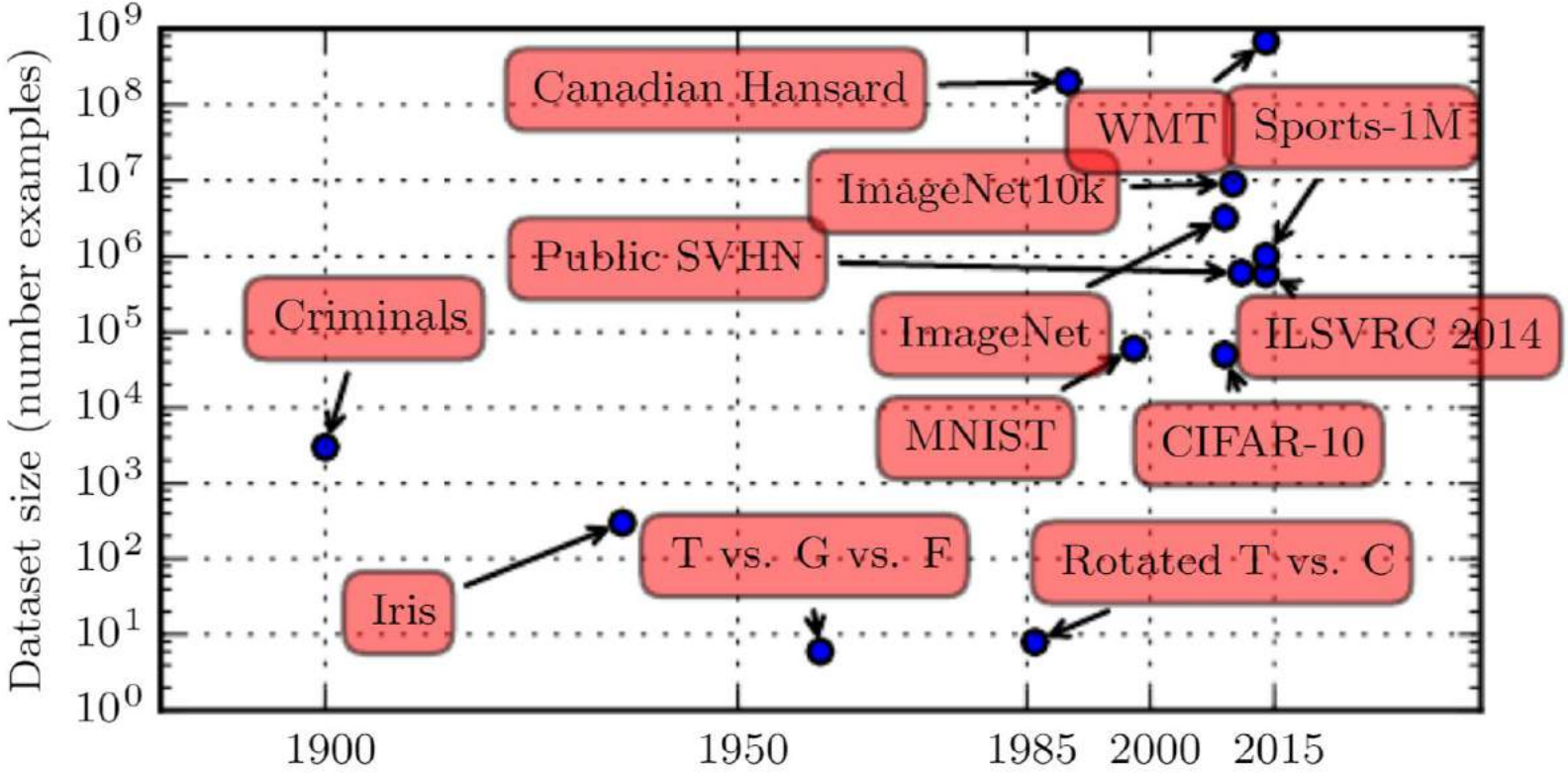


# Третья волна…



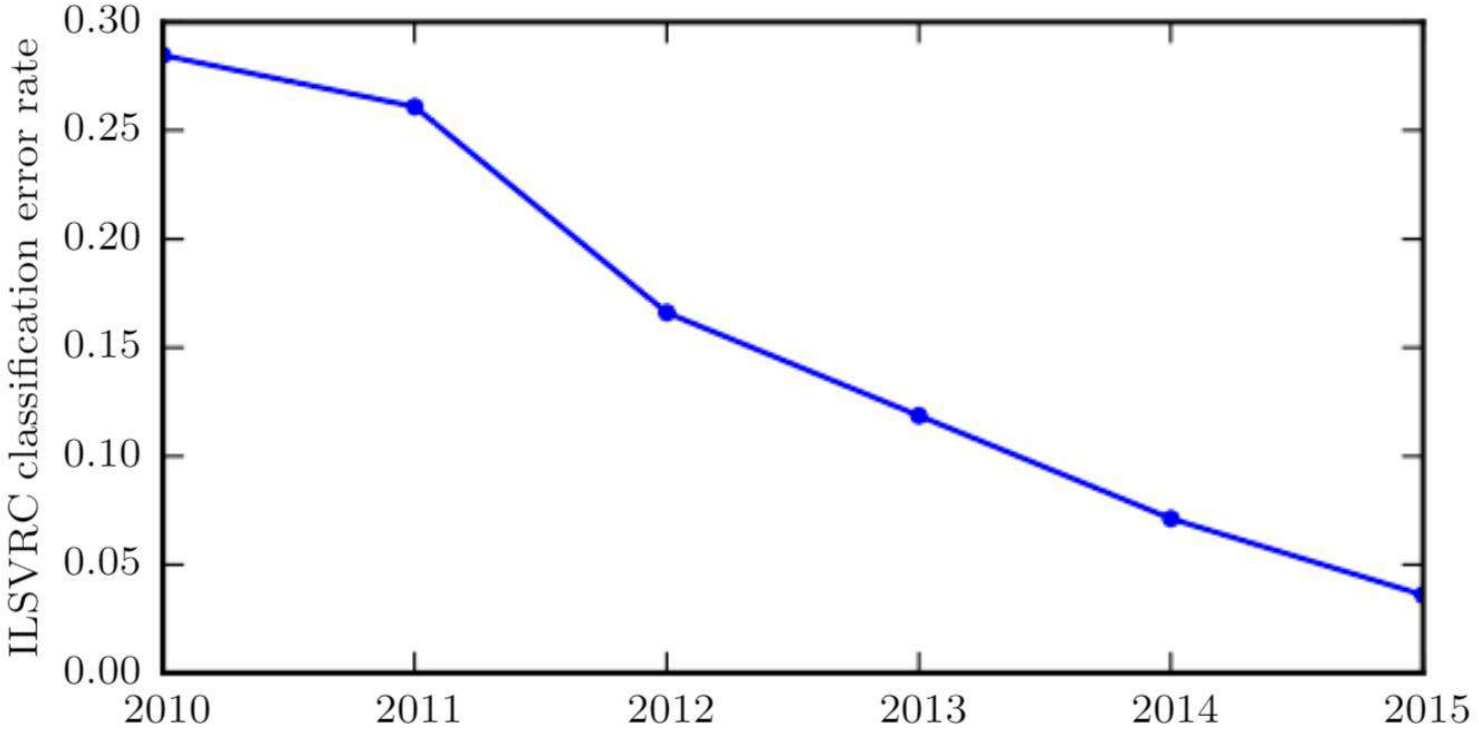

www.deeplearningbook.org

# Датасеты становятся больше…



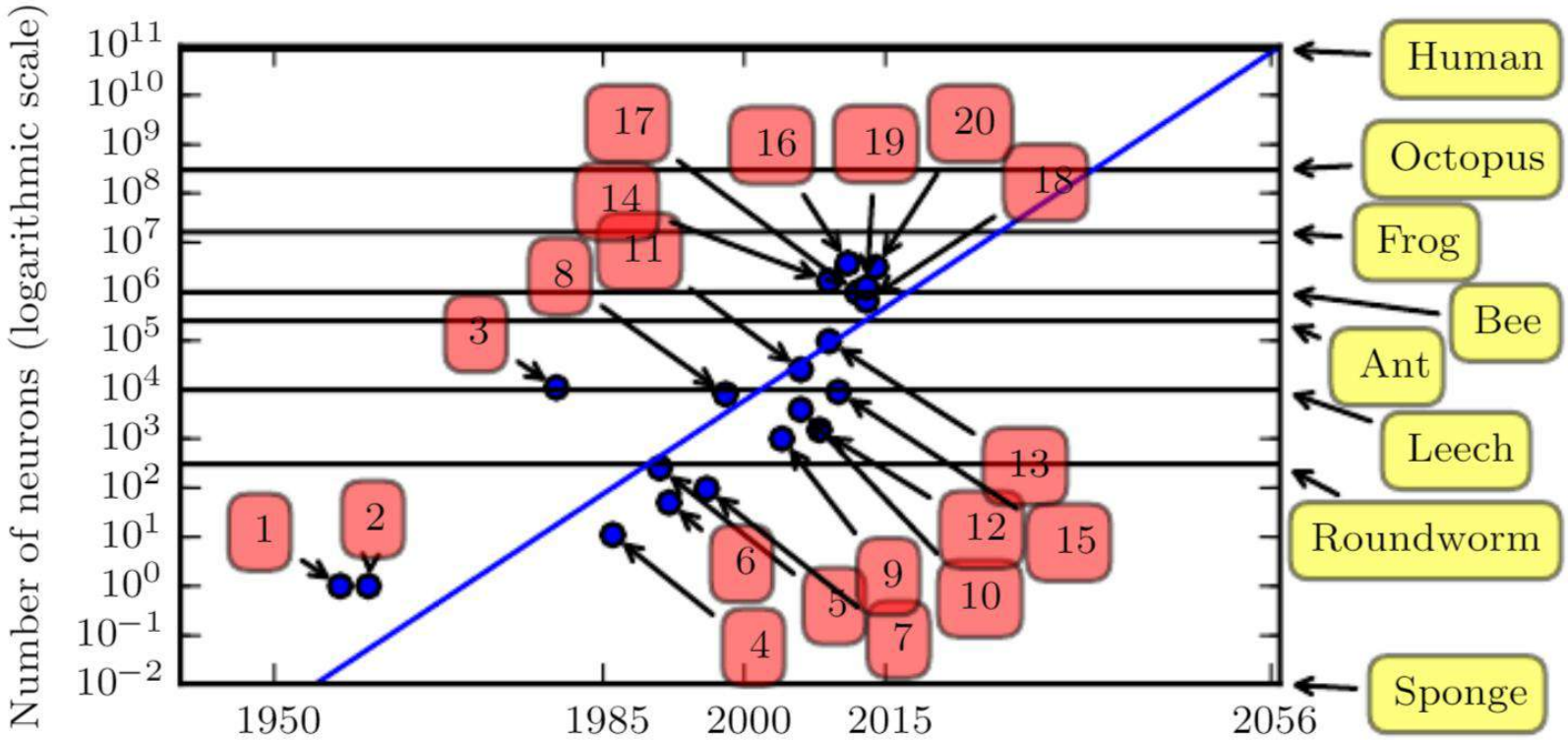

www.deeplearningbook.org

# Нейронки гораздо точнее…




www.deeplearningbook.org

# Нейронки становятся больше…




www.deeplearningbook.org

# Бигдата и нейронки – созданы друг для друга

- Машины опорных векторов
- Факторизация слоев
- Нелинейность

# А если данных все таки собрано мало?

- Сколько нужно данных?
- У большинства компаний – «маленькая» бигдата
- Простые классические алгоритимы: naïve bayes, logistic regression, support vector machine (SVM), decision tree, gbt/random forest (https://tech.yandex.ru/catboost/)

# Подтянулись GPU и железо

- Универсальные GPU
- CUDA
- Работа с тензорами
- Диски, кластера: Spark, Hadoop/HDFS, Amazon s3
- Языки: Scala

# Парад бесплатных фреймворков

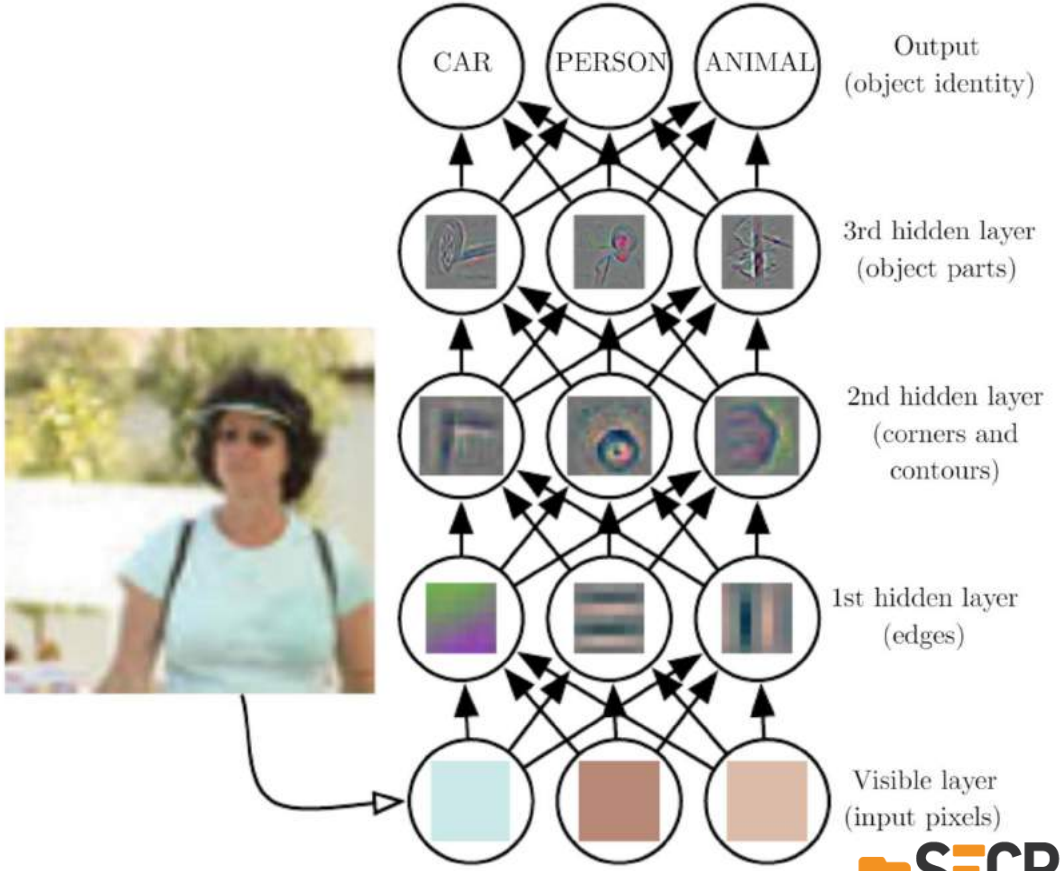
- TensorFlow (Google)
- Torch
- Theano
- Keras
- Deeplearning4j
- CNTK (Microsoft)
- DSSTNE (Amazon)
- Caffe

# Вендоры скупают ученых

- Facebook (Yann LeCun)
- Baidu (Andrew Ng, уже правда уходит, достали тупить ☺ )
- Google (Ian Goodfellow)
- SalesForce (Richard Socher)
- openai.com …

# Как работает нейронка?

- Все просто – почти как наш мозг ☺
- Вспомните школьные годы – и все станет понятно

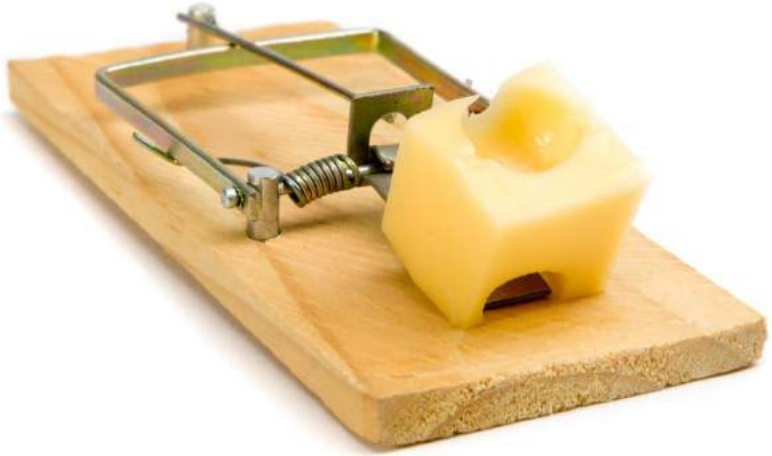



www.deeplearningbook.org

# Кейсы применения нейронок и алгоритмов машинного обучения

- Предсказание следующего действия (RNN, …)
- Кластеризация (autoencoders)
- Кто из клиентов уйдет, кто из сотрудников уволится (churn rate: FFN, CNN)
- Сколько стоит квартирка (regression)
- Анализ причин (InfoGANs)
- Персонализация

# Где же подвох? ☺

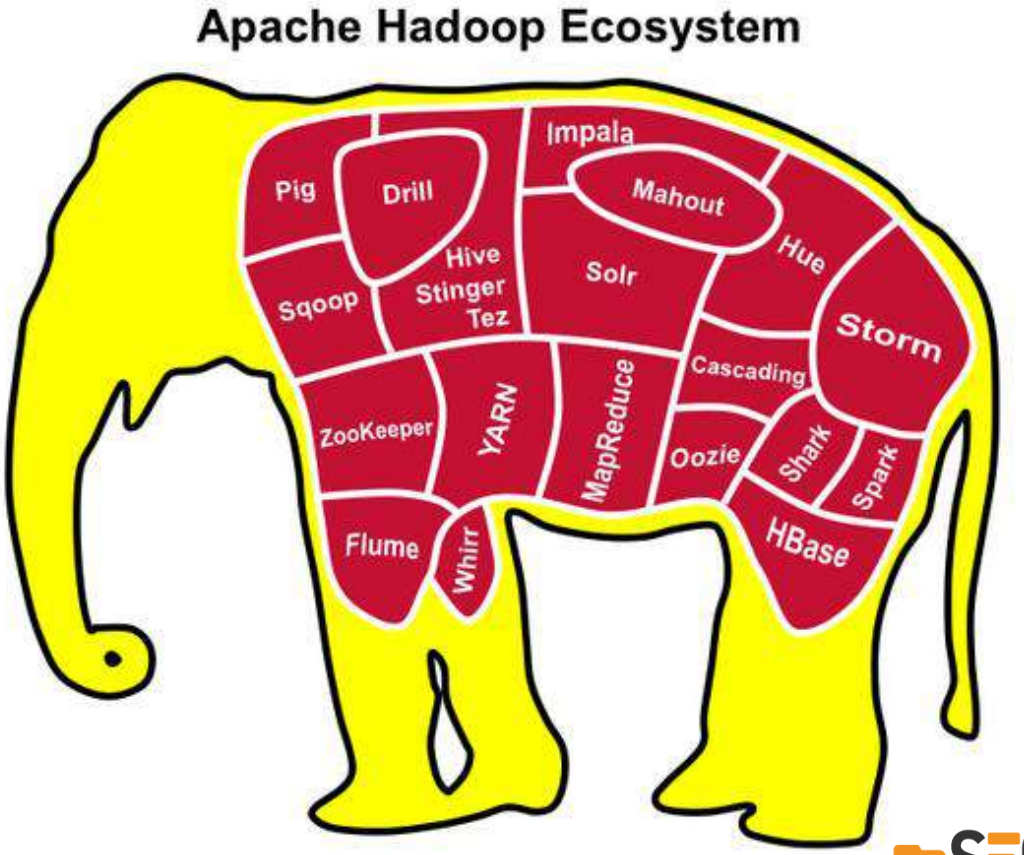


# А они то есть!!!

# И не один. ☺

# Подвох 1

- Нужна бигдата
- Только конкретная, ваша, а не общедоступная
- Сможете собрать/купить?

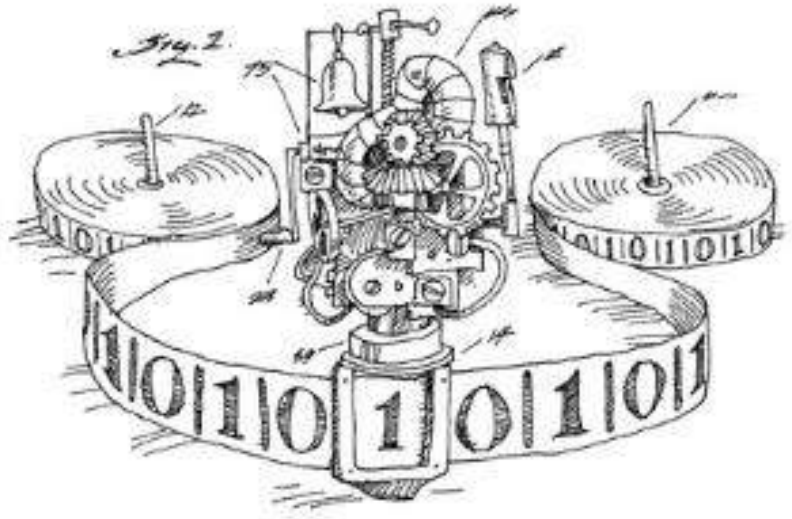


# Подвох 2 – семантический разрыв

- Классификация
- Регрессия
- Кластеризация
- Анализ скрытых факторов в ином измерении

- Как увеличить прибыль?
- Как удержать клиента?
- Как предложить самое нужное?

# Машина Тьюринга и … GTA




**Нужно создавать новые абстракции, нужны «нейронные» программисты, менеджеры и прОдукты**

# Подвох 3 – всем тут все понятно? ☺



To see that the weight scaling rule is exact, we can simplify $\tilde{P}_{\text{ensemble}}$:

$$\tilde{P}_{\text{ensemble}}(\mathrm{y} = y \mid \mathbf{v}) = \sqrt[2^n]{\prod_{\boldsymbol{d} \in \{0,1\}^n} P(\mathrm{y} = y \mid \mathbf{v}; \boldsymbol{d})} \tag{7.60}$$

$$= \sqrt[2^n]{\prod_{\boldsymbol{d} \in \{0,1\}^n} \text{softmax}\left(\boldsymbol{W}^\top (\boldsymbol{d} \odot \mathbf{v}) + \boldsymbol{b}\right)_y} \tag{7.61}$$

$$= \sqrt[2^n]{\prod_{\boldsymbol{d} \in \{0,1\}^n} \frac{\exp\left(\boldsymbol{W}^\top_{y,:}(\boldsymbol{d} \odot \mathbf{v}) + b_y\right)}{\sum_{y'} \exp\left(\boldsymbol{W}^\top_{y',:}(\boldsymbol{d} \odot \mathbf{v}) + b_{y'}\right)}} \tag{7.62}$$

$$= \frac{\sqrt[2^n]{\prod_{\boldsymbol{d} \in \{0,1\}^n} \exp\left(\boldsymbol{W}^\top_{y,:}(\boldsymbol{d} \odot \mathbf{v}) + b_y\right)}}{\sqrt[2^n]{\prod_{\boldsymbol{d} \in \{0,1\}^n} \sum_{y'} \exp\left(\boldsymbol{W}^\top_{y',:}(\boldsymbol{d} \odot \mathbf{v}) + b_{y'}\right)}} \tag{7.63}$$

# Подвох 3 – а тут? ☺



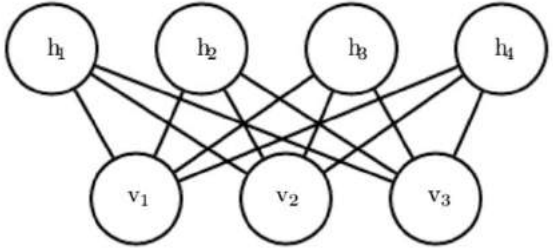



Figure 16.14: An RBM drawn as a Markov network.

and

$$p(\mathbf{v} \mid \mathbf{h}) = \Pi_i p(\mathrm{v}_i \mid \mathbf{h}). \tag{16.12}$$

The individual conditionals are simple to compute as well. For the binary RBM we obtain:

$$P(\mathrm{h}_i = 1 \mid \mathbf{v}) = \sigma\left(\mathbf{v}^\top \boldsymbol{W}_{:,i} + b_i\right), \tag{16.13}$$

$$P(\mathrm{h}_i = 0 \mid \mathbf{v}) = 1 - \sigma\left(\mathbf{v}^\top \boldsymbol{W}_{:,i} + b_i\right). \tag{16.14}$$

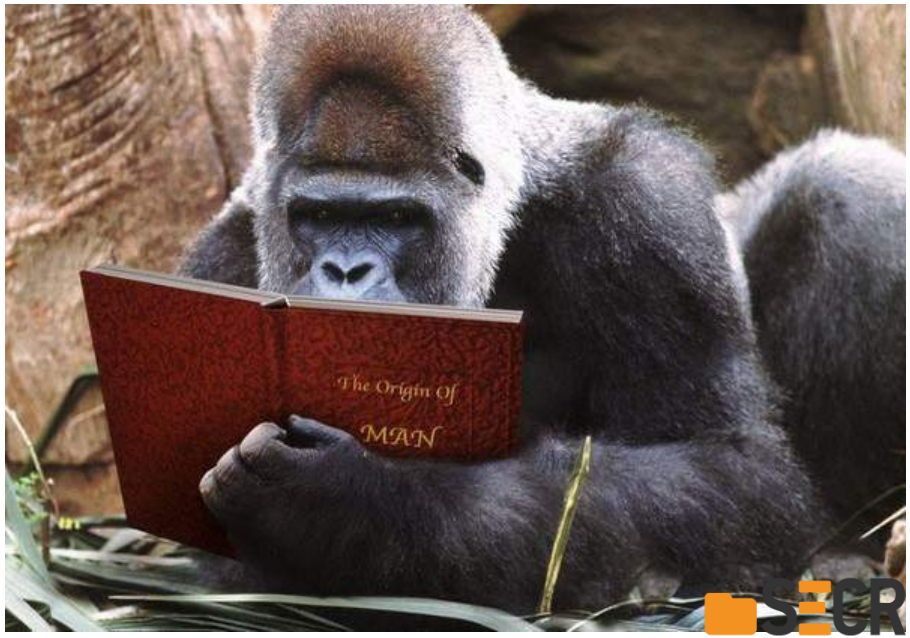
Together these properties allow for efficient **block Gibbs** sampling, which alter-

# Подвох 3 – нужно долго учиться

- Хорошая матподготовка выше среднего
- Уметь писать код
- Исследовательский дух, много читать
- Опыт и интуиция

# Подвох 4 – никаких гарантий

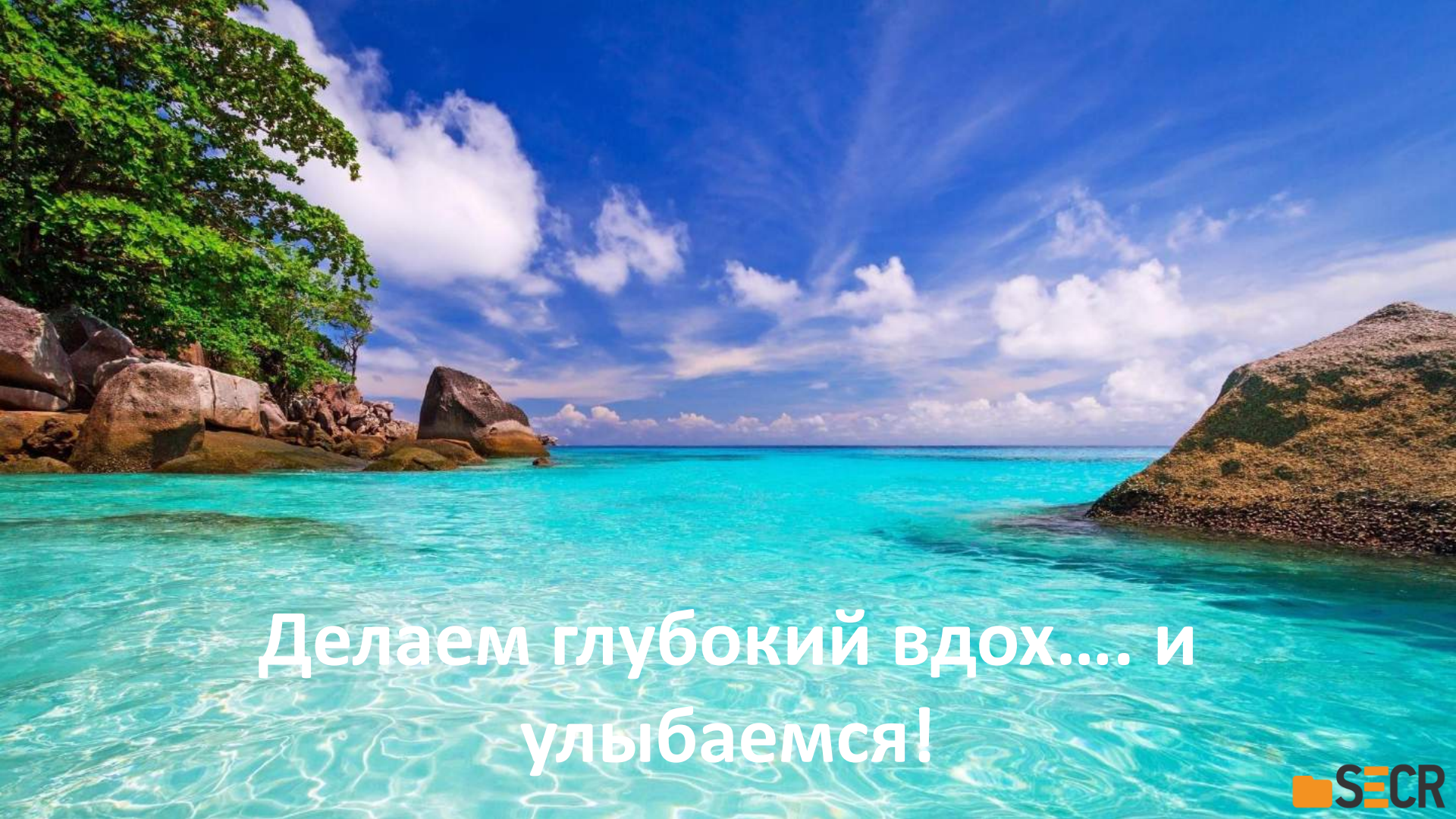
- В интернете - работает
- На ваших данных – нет
- Где ошибка? В данных, в модели, в коэффициентах, в коде, в голове??



# Подвох 5: полная цепочка - сложна

- Сбор данных
- Фильтрация, валидация
- Обучение модели
- Раздача предсказаний
- Контроль качества


Рис. 6. Внешний вид и основные составляющие школьного микроскопа


Делаем глубокий вдох…. и
улыбаемся!

# Ражнирование товаров (Google Play)

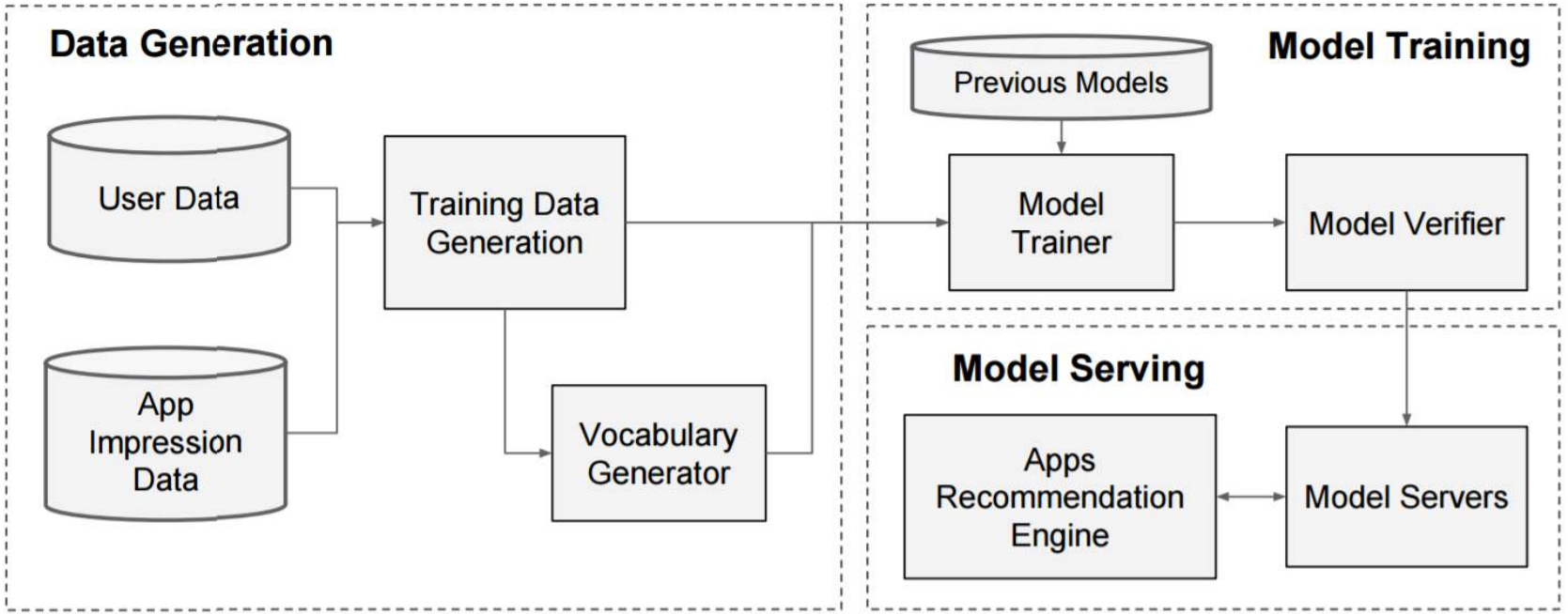



Figure 3: Apps recommendation pipeline overview.

arxiv.org/abs/1606.07792

# Ражнирование товаров (Google Play)

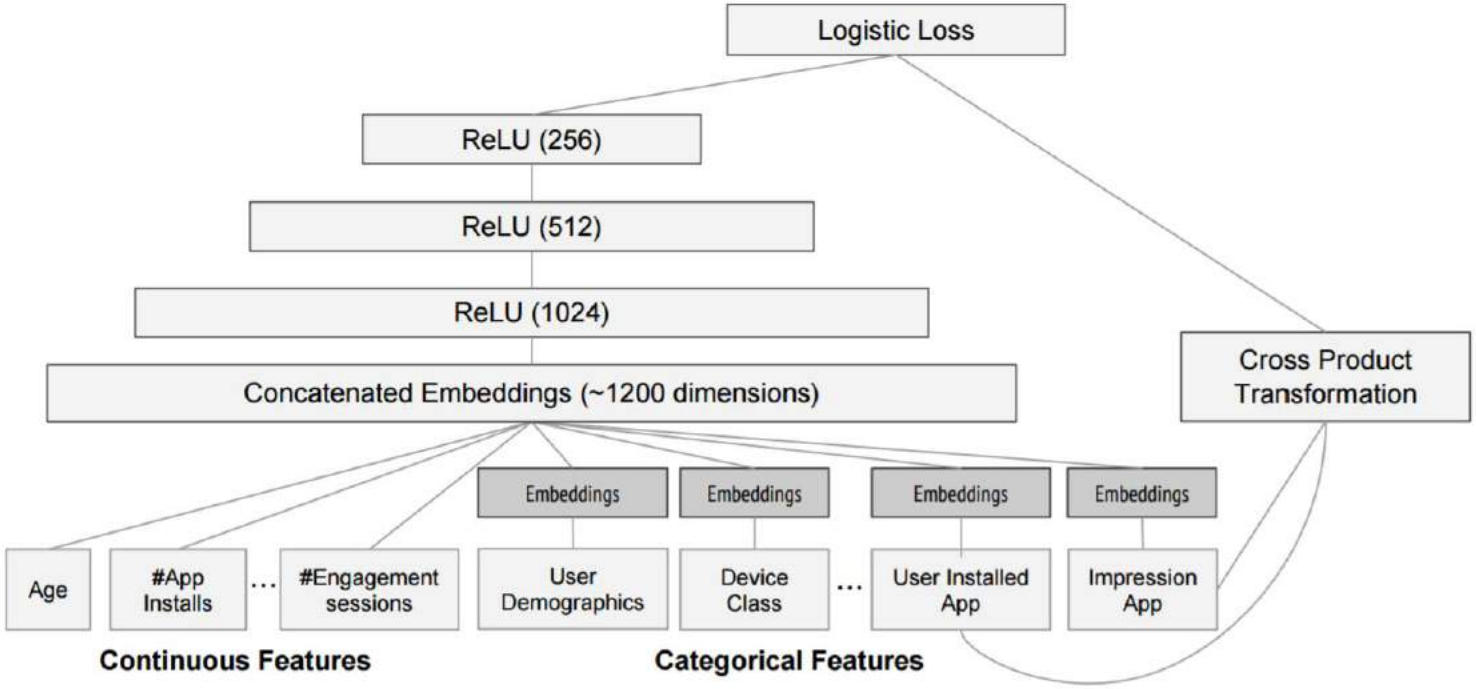



Figure 4: Wide & Deep model structure for apps recommendation.

arxiv.org/abs/1606.07792

# Ражнирование товаров (Google Play)

$$P(Y=1|\mathbf{x}) = \sigma(\mathbf{w}_{wide}^T[\mathbf{x}, \phi(\mathbf{x})] + \mathbf{w}_{deep}^T a^{(l_f)} + b)$$

- Собирается все что есть…
- Засовывается в нейронку
- Нейронка предсказывает вероятность клика/покупки приложения – для каждого приложения из отобранных
- Приложения сортируются и отображаются. Все! ☺

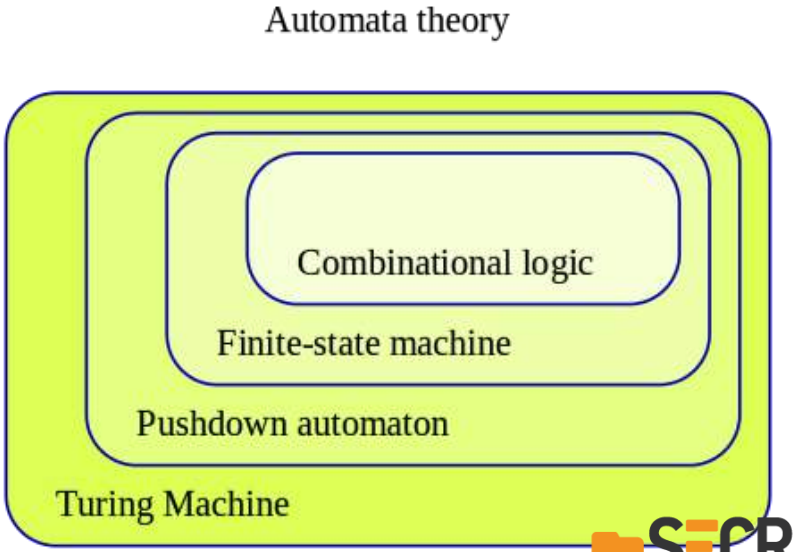
arxiv.org/abs/1606.07792

# Ну что, нырнем поглубже? Может заболеть голова ☺

# Абстрактные знания и фундаментальная наука

- **Логика, реляционная алгебра**
- Дискретная математика, теория графов, **теория автоматов, комбинаторика**, теория кодирования
- Теория алгоритмов
- Линейная алгебра
- Интегральное и дифф. исчисление
- Теория вероятностей
- Теория оптимизации и численные методы

**времени на это практически нет*



# Восьмая проблема Гильберта и другие штучки

• До сих пор неясно распределение простых чисел (Гипотеза Римана)

• Эффективные алгоритмы нередко находят методом «тыка», многие мало изучены

• Нейронные сети не должны … сходиться, но сходятся. И плохо-плохо изучены.

Наука только открывает ящик Пандоры!

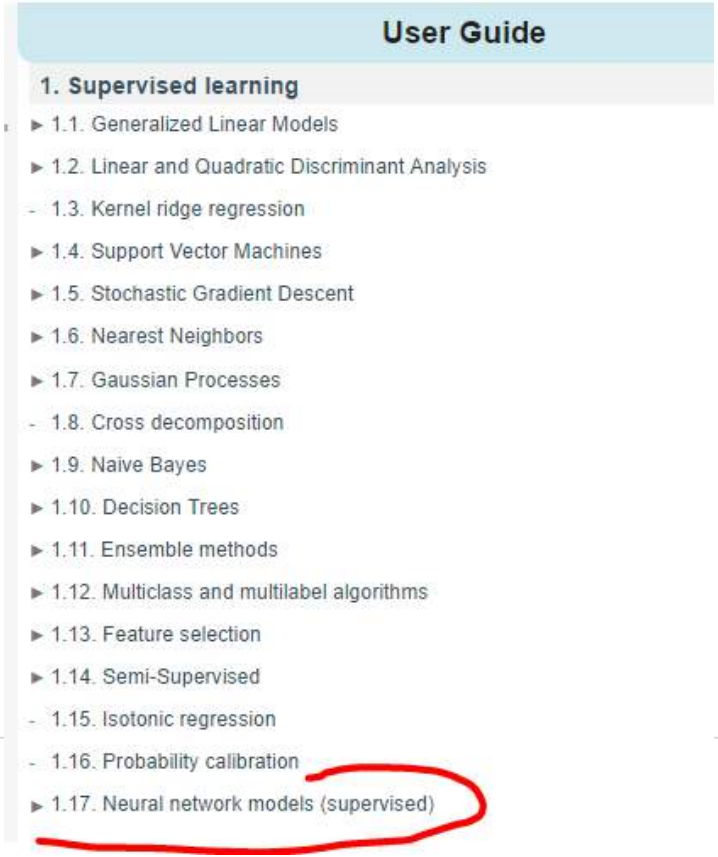
# Когда заканчивается наука, «начинается машинное обучение»

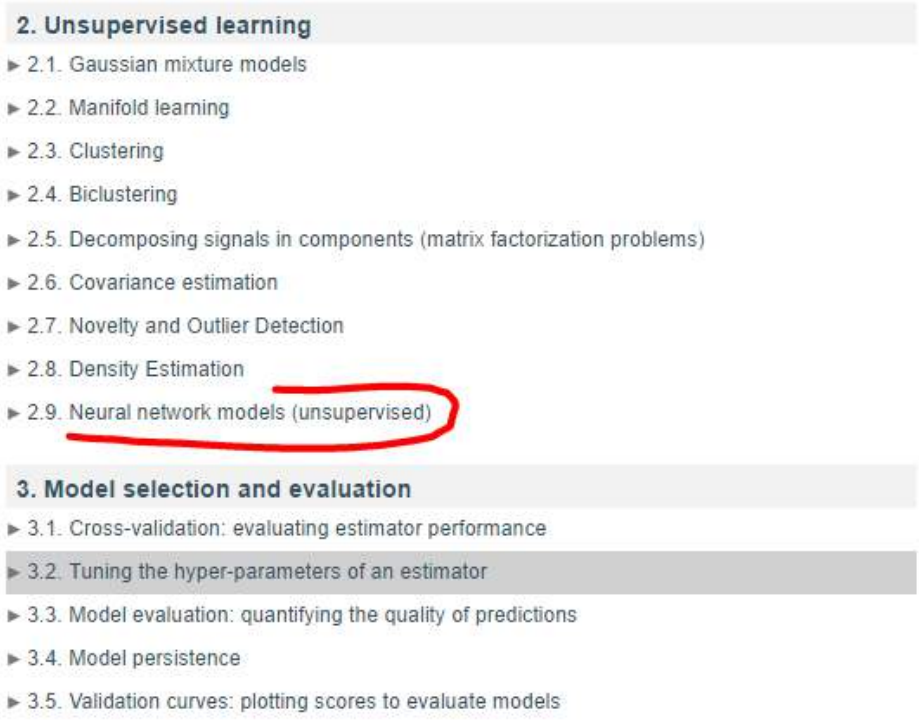
- Четкая кластеризация: K-means (EM)
- Нечеткая кластеризация: Latent dirichlet allocation
- Модели Маркова
- Google Page Rank
- Monte Carlo алгоритмы
- Las Vegas алгоритмы (в т.ч. «обезьянья сортировка»)

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i)^2$$

где $k$ — число кластеров, $S_i$ — полученные кластеры, $i = 1, 2, \ldots, k$ и $\mu_i$ — центры масс векторов $x_j \in S_i$.

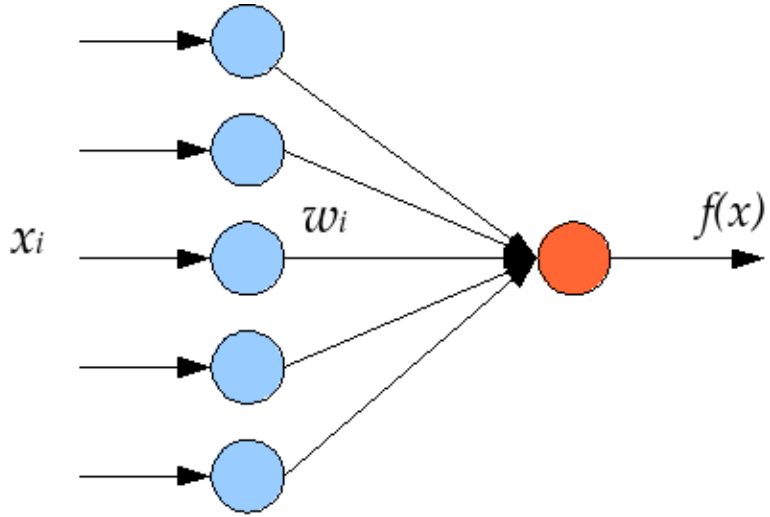
# Машинное обучение и … где-то в конце, нейронки (scikit-learn)



# Рассмотрим кусочек нейронки - нейрон

- Линейная регрессия
- Логистическая регрессия
- Сигмоид
- Здравствуй, линейная алгебра! Чмоки чмоки



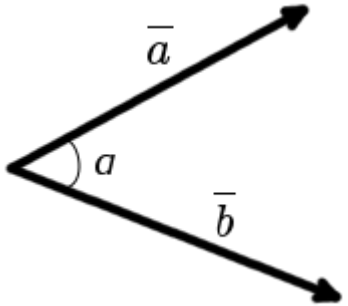

$$f(x) = \text{sign}(\sum_{}^{n} w_i x_i - \theta)$$

$$f(x,b) = b_1 x_1 + b_2 x_2 + \ldots + b_k x_k = \sum_{j=1}^{k} b_j x_j = x^T b,$$

где $x^T = (x_1, x_2, \ldots, x_k)$ — вектор регрессоров, $b = (b_1, b_2, \ldots, b_k)^T$ — вектор-столбец параметров (коэффициентов)

# Вектор, косинус угла между векторами

- Вектор – точка в N-мерном пространстве. Размер вектора.
- Косинус угла между векторами




Если же векторы заданы в пространстве, то есть $\bar{a} = (a_x; a_y; a_z)$ и $\bar{b} = (b_x; b_y; b_z)$, то косинус угла вычисляется по формуле

$$\cos\phi = \frac{(\bar{a}, \bar{b})}{|\bar{a}| \cdot |\bar{b}|} = \frac{a_x \cdot b_x + a_y \cdot b_y + a_z \cdot b_z}{\sqrt{a_x^2 + a_y^2 + a_z^2}\sqrt{b_x^2 + b_y^2 + b_z^2}}$$

# Уравнение плоскости через точку и нормаль

- Плоскость: косинус угла между нормалью и MP = 0
- Если угол меньше 90, косинус >0, иначе – косинус <0.

$$\cos\varphi = 0,$$

$$\frac{\bar{a}\cdot\bar{b}}{|\bar{a}|\cdot|\bar{b}|} = 0 \Leftrightarrow \bar{a}\cdot\bar{b} = 0$$

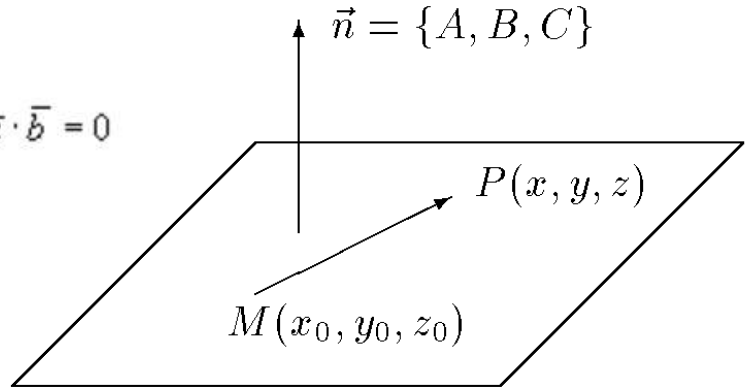



Рис. 1

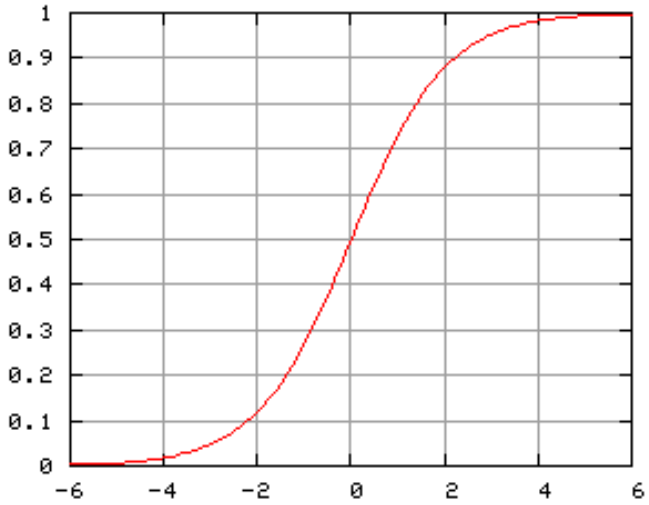
***Уравнение плоскости, проходящей через точку, перпендикулярно вектору нормали***

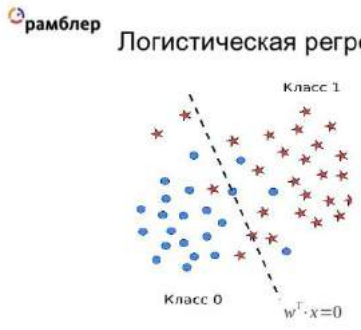
Чтобы составить уравнение плоскости, зная координаты точки плоскости $M(x_0, y_0, z_0)$ и вектора нормали плоскости $\bar{n} = \{A; B; C\}$ можно использовать следующую формулу.

$$A(x - x_0) + B(y - y_0) + C(z - z_0) = 0$$

# Сигмоид, логистическая регрессия

- Зачем нужен сигмоид?
- Визуализация
- Нелинейная активация, виды



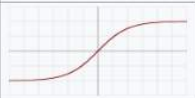
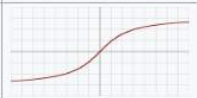
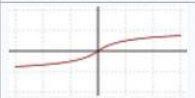




$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

$$f(x, b) = b_1 x_1 + b_2 x_2 + \ldots + b_k x_k = \sum_{j=1}^{k} b_j x_j = x^T b,$$

где $x^T = (x_1, x_2, \ldots, x_k)$ — вектор регрессоров, $b = (b_1, b_2, \ldots, b_k)^T$ — вектор-столбец параметров (коэффициентов)

# Другие функции активации

| Name | Plot | Equation | Derivative (with respect to $x$) | Range | Order of continuity |
|---|---|---|---|---|---|
| Identity |  | $f(x) = x$ | $f'(x) = 1$ | $(-\infty, \infty)$ | $C^\infty$ |
| Binary step |  | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$ | $\{0, 1\}$ | $C^{-1}$ |
| Logistic (a.k.a. Soft step) |  | $f(x) = \frac{1}{1 + e^{-x}}$ | $f'(x) = f(x)(1 - f(x))$ | $(0, 1)$ | $C^\infty$ |
| TanH |  | $f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$ | $f'(x) = 1 - f(x)^2$ | $(-1, 1)$ | $C^\infty$ |
| ArcTan |  | $f(x) = \tan^{-1}(x)$ | $f'(x) = \frac{1}{x^2 + 1}$ | $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$ | $C^\infty$ |
| Softsign [7][8] |  | $f(x) = \frac{x}{1 + \lvert x \rvert}$ | $f'(x) = \frac{1}{(1 + \lvert x \rvert)^2}$ | $(-1, 1)$ | $C^1$ |
| Rectified linear unit (ReLU)[9] |  | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ | $[0, \infty)$ | $C^0$ |

# Активация нейронки, матрицы

Here's how we calculate the total net input for $h_1$:

$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$

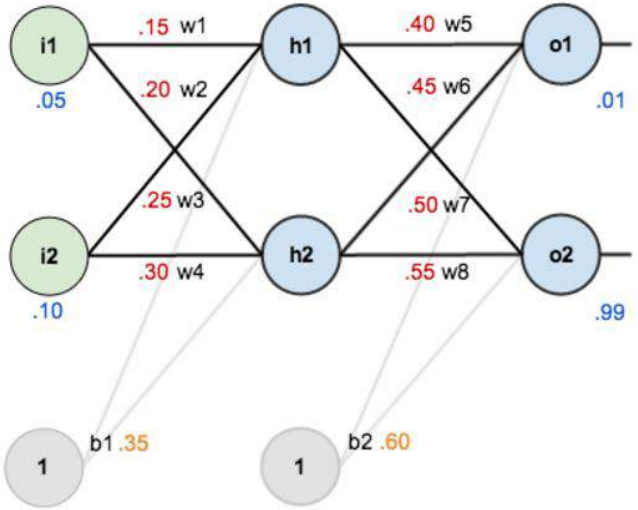
$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

We then squash it using the logistic function to get the output of $h_1$:

$$out_{h1} = \frac{1}{1+e^{-net_{h1}}} = \frac{1}{1+e^{-0.3775}} = 0.593269992$$

Carrying out the same process for $h_2$ we get:

$$out_{h2} = 0.596884378$$



# Умножаем матрицы «в уме»

- 2 входных вектора, размером 3 => матрица B(3,2)
- Ширина слоя сети = 2
- Веса сети => матрица A(2,3)
- Получаем активации слоя для каждого вх. вектора:

(2, 2).

веса входы

1 1

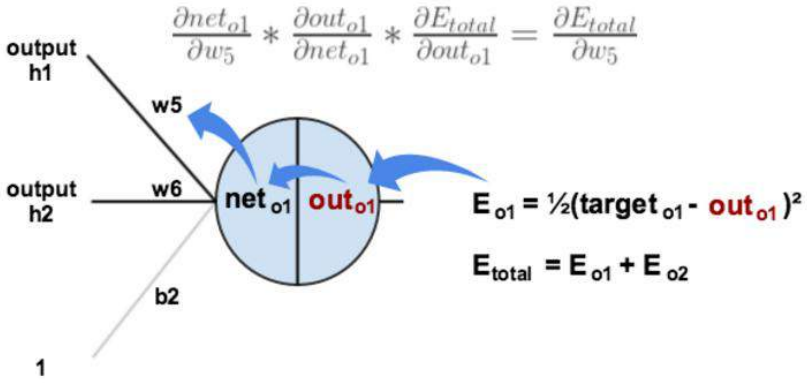
1ш 2ш

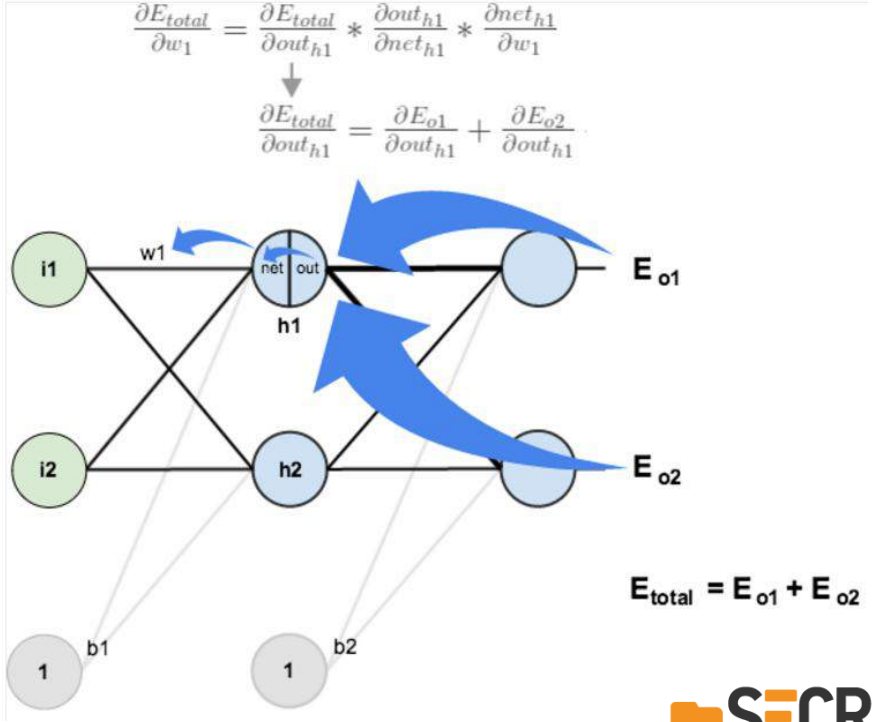
$$\mathbf{AB} = \begin{pmatrix} a & b & c \\ x & y & z \end{pmatrix} \begin{pmatrix} \alpha & \rho \\ \beta & \sigma \\ \gamma & \tau \end{pmatrix} = \begin{pmatrix} a\alpha + b\beta + c\gamma & a\rho + b\sigma + c\tau \\ x\alpha + y\beta + z\gamma & x\rho + y\sigma + z\tau \end{pmatrix},$$

1 2

# Обратное распространение ошибки

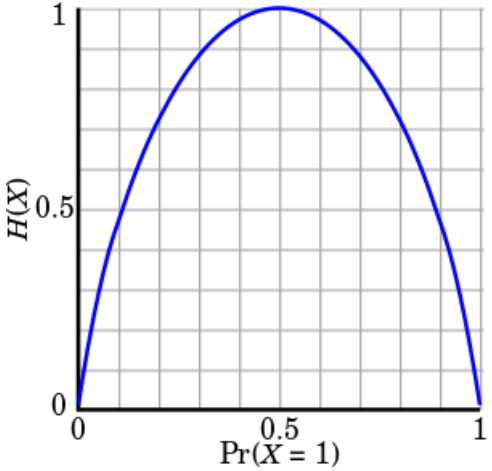
- Chain rule, здравствуй дифференциальное исчисление! Чмоки чмоки.
- На самом деле тут все просто!



# Cost - функции

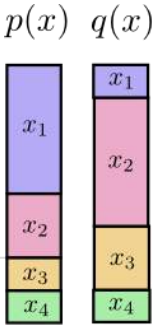
- mean squared error
- entropy, cross-entropy (binary/multiclass), здравствуй теория информации и тервер!




## Cross Entropy

- Entropy: $H(X) = -\sum_x p(x) \log p(x)$
- Cross Entropy: $H_c(X) = -\sum_x p(x) \log q(x)$
- Cross entropy is a distance measure between p(x) and q(x): p(x) is the true probability; q(x) is our estimate of p(x).

$$H_c(X) \geq H(X)$$




## Cross-Entropy: $H_p(q)$

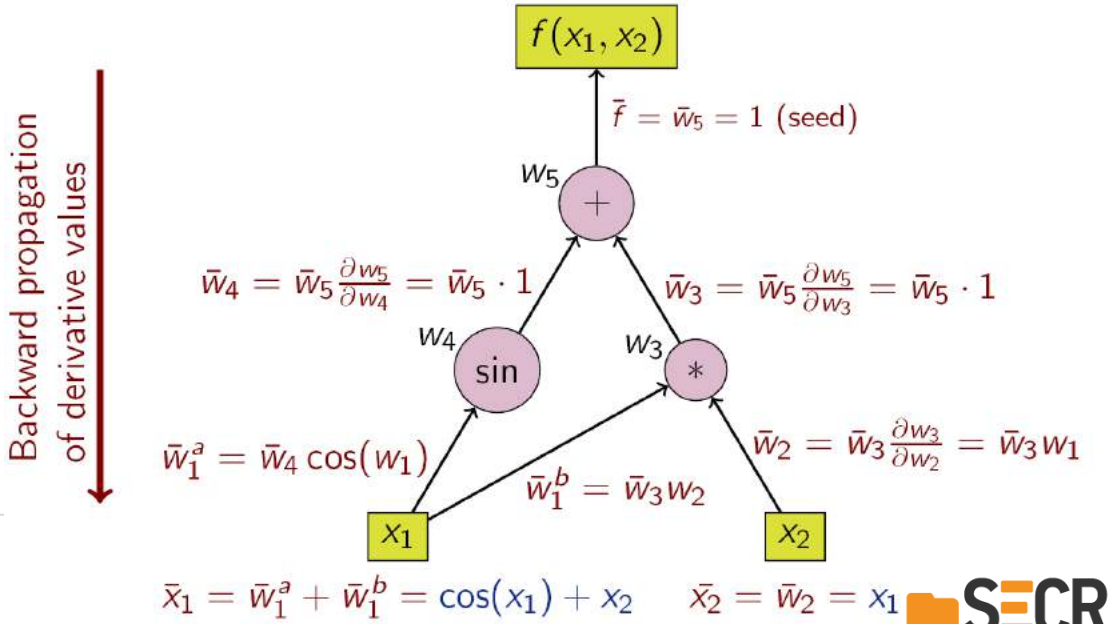
Average Length
of message from q(x)
using code for p(x).

# Cost – функции, Keras

- mean_squared_error / mse
- mean_absolute_error / mae
- mean_absolute_percentage_error / mape
- mean_squared_logarithmic_error / msle
- squared_hinge
- hinge
- binary_crossentropy: Also known as logloss.
- categorical_crossentropy: Also known as multiclass logloss. **Note**: using this objective requires that your labels are binary arrays of shape `(nb_samples, nb_classes)`.
- sparse_categorical_crossentropy: As above but accepts sparse labels. **Note**: this objective still requires that your labels have the same number of dimensions as your outputs; you may need to add a length-1 dimension to the shape of your labels, e.g with `np.expand_dims(y, -1)`.
- kullback_leibler_divergence / kld: Information gain from a predicted probability distribution Q to a true probability distribution P. Gives a measure of difference between both distributions.
- poisson: Mean of `(predictions - targets * log(predictions))`
- cosine_proximity: The opposite (negative) of the mean cosine proximity between predictions and targets.

# Автоматическое/ручное дифференцирование

- Torch7 – ручное, afaik
- Theano – автоматическое
- Tensorflow – автоматическое
- Deeplearning4j – ручное
- Keras (Theano/Tensorflow)



# Методы градиентного спуска (SGD)

- Stochastic gradient descent $\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i)}; y^{(i)})$.
- Mini-batch gradient descent $\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i:i+n)}; y^{(i:i+n)})$.
- Momentum:

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta).$$

$$\theta = \theta - v_t.$$

- Nesterov accelerated gradient
- Adagrad
- Adadelta
- RMSprop
- **Adam**

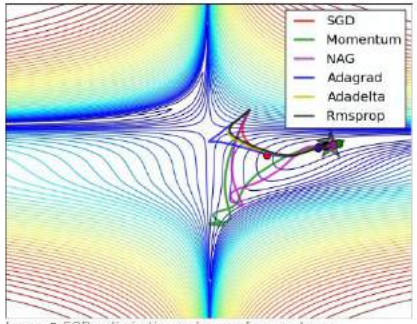


Image 5: SGD optimization on loss surface contours

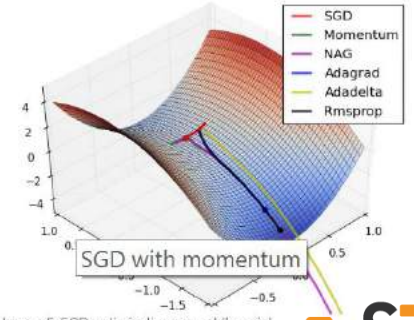


Image 6: SGD optimization on saddle point

# Тензоры. Проще SQL.

• В терминологии нейронок – это многомерные массивы элементов одного типа.

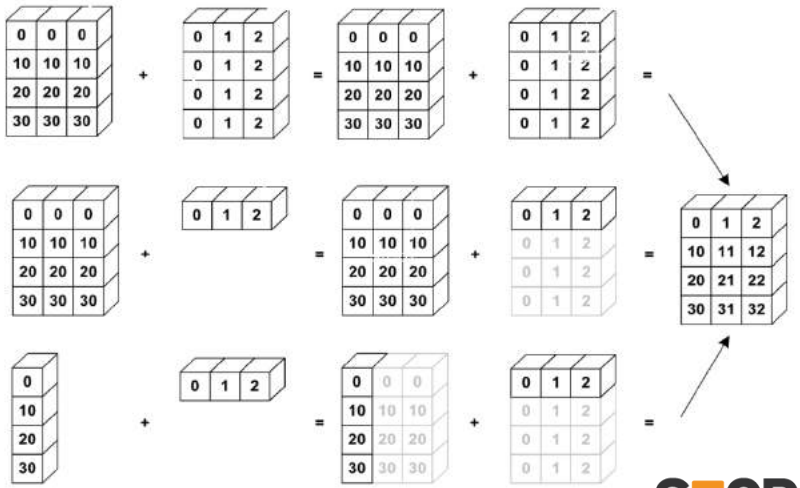
• Требуется их складывать, умножать, делить и выполнять статистические операции: Basic Linear Algebra Subprograms (BLAS)

• numpy (python)

• nd4j (java)

• Tensor (torch/lua)

CUDA, GPU



# Тензоры

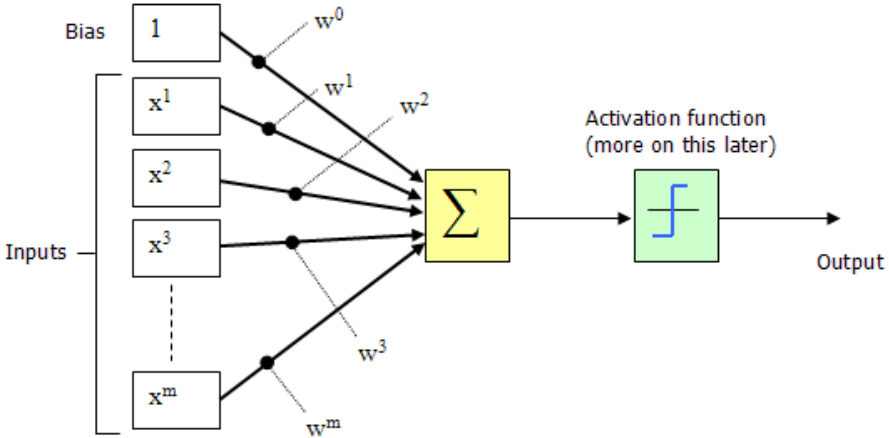
- nd4j, примерно тоже самое

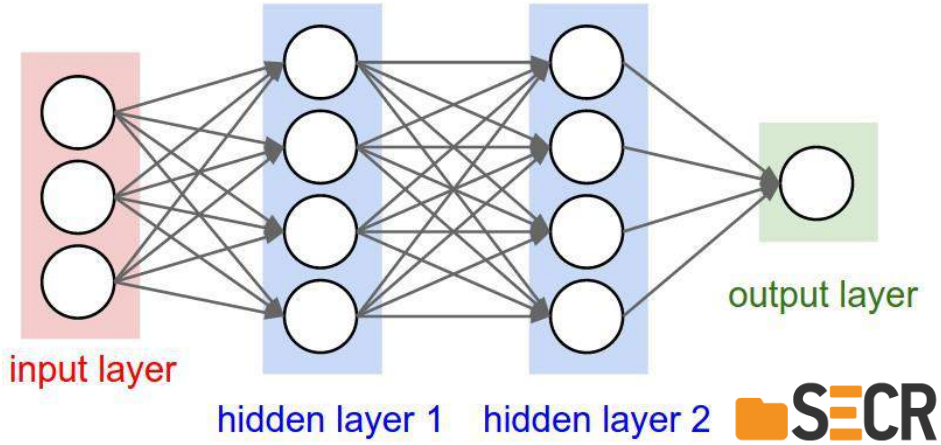
```
case Add:
    INDArray sum = inputs[0].dup();
    for( int i=1; i<inputs.length; i++){
        sum.addi(inputs[i]);
    }
    return sum;
case Subtract:
    if(inputs.length != 2) throw new IllegalA
    return inputs[0].sub(inputs[1]);

case Product:

    //Alex Serbul improvement
    INDArray mul = inputs[0].dup();
    for( int i=1; i<inputs.length; i++){
        mul.muli(inputs[i]);
    }
    return mul;

default:
    throw new UnsupportedOperationException("
```

```
INDArray input = Nd4j.zeros(3, INPUT_WIDTH, SEQ_LENGTH);
INDArray labels = Nd4j.zeros(3, OUTPUT_WIDTH, SEQ_LENGTH);

INDArray inputMasks = Nd4j.zeros(3, SEQ_LENGTH);
INDArray labelMasks = Nd4j.zeros(3, SEQ_LENGTH);

//
input.putScalar(new int[] { 0, 0, 0 }, 1);
labels.putScalar(new int[] { 0, 0, 0 }, 1);
inputMasks.putScalar( new int[] { 0, 0 }, 1 );
labelMasks.putScalar( new int[] { 0, 0 }, 1 );

//
input.putScalar(new int[] { 1, 0, 0 }, 1);
input.putScalar(new int[] { 1, 0, 1 }, 1);

labels.putScalar(new int[] { 1, 0, 1 }, 1);
labels.putScalar(new int[] { 1, 1, 2 }, 1);
```

# Простой классификатор

- Зачем нужна нелинейность?
- Зачем нужны слои?



# Врач никому не нужен? ☺ Ныряем в прикладные кейсы

# Полезные (готовые) инструменты

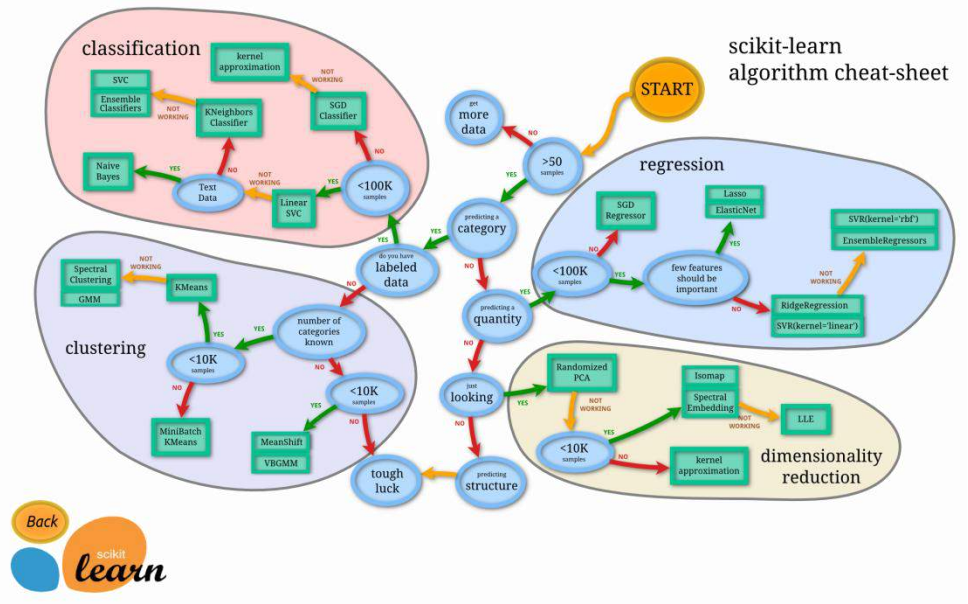
- Rapidminer
- SAS
- SPSS
- …




Готовые блоки, серверные редакции (hadoop), графики

# Полезные библиотеки (бесплатные)

- Spark MLlib (scala/java/python) – много данных
- scikit-learn.org (python) – мало данных
- R



# Рабочее место аналитика

# Аналитик

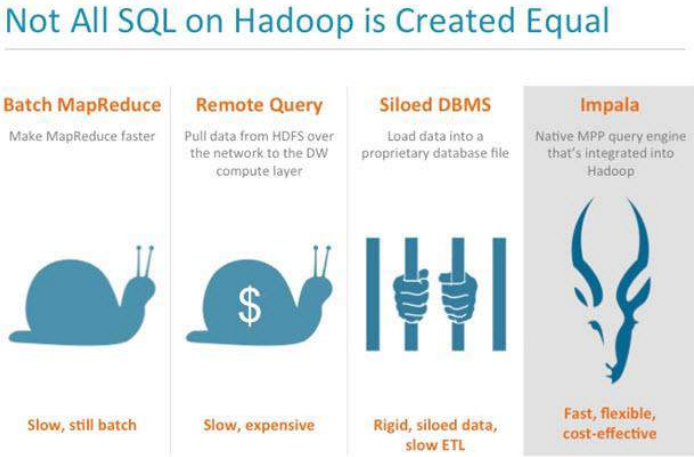
- Организовать сбор данных
- Минимум программирования
- Работа в инструментах (Rapidminer, R, SAS, SPSS)
- Bigdata – как SQL

# Война систем хранения

- SQL на MapReduce: Hive, Pig, Spark SQL
- SQL на MPP (massive parallel processing):

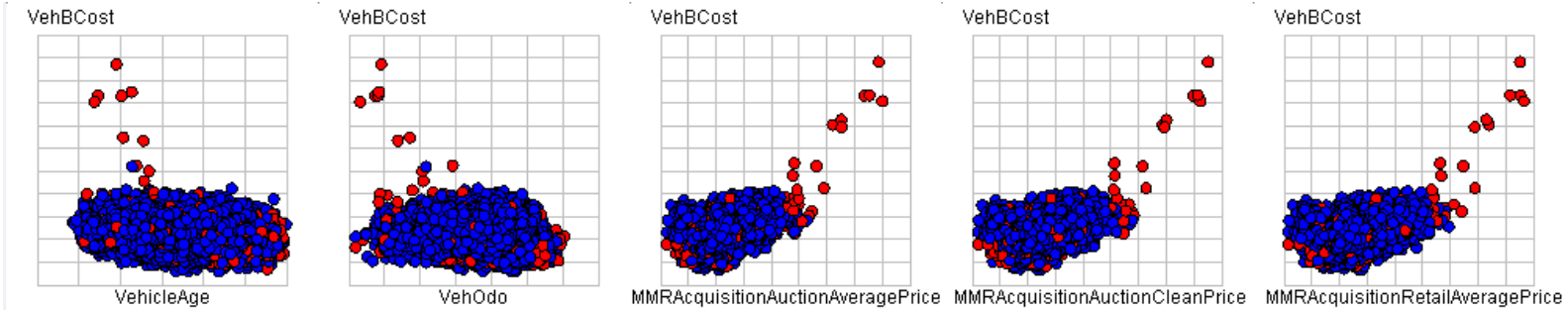
Impala, Presto, Amazon RedShift, Vertica

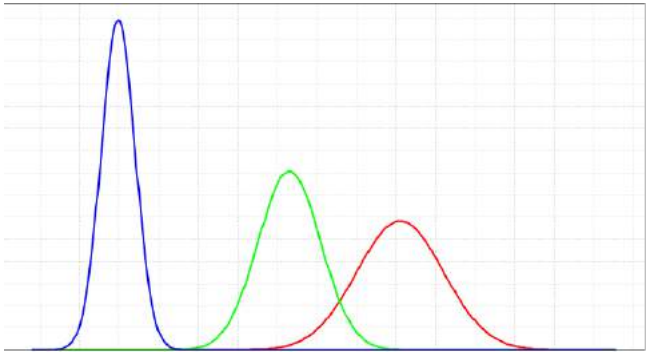
- NoSQL: Cassandra, Hbase, Amazon DynamoDB
- Классика: MySQL, MS SQL, Oracle, …

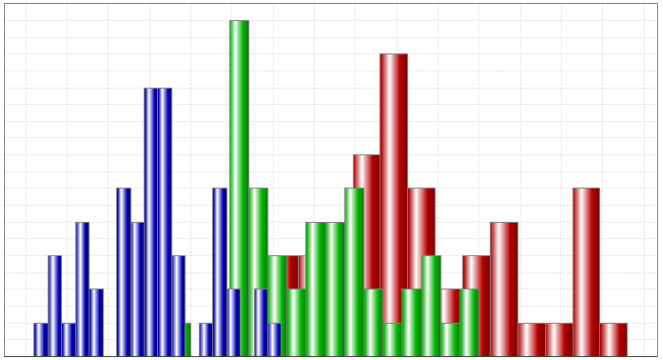


# Визуализация

# Визуализация!



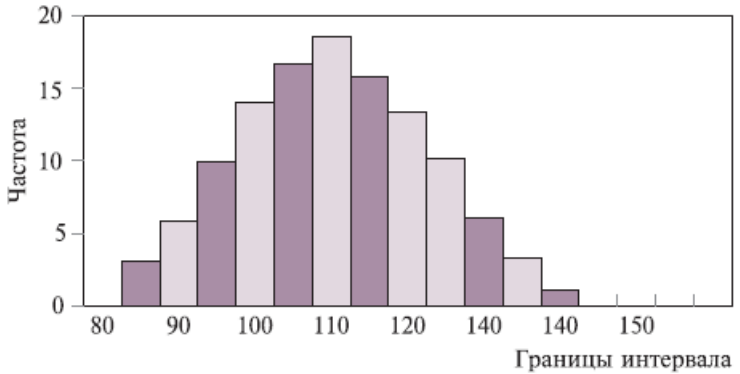


# Визуализация!

- Кто мои клиенты (возраст, средний чек, интересы)?
- Тренды, графы
- Корреляция значений
- 2-3, иногда больше измерений
- «Дешевле/проще» кластеризации

# Визуализация!

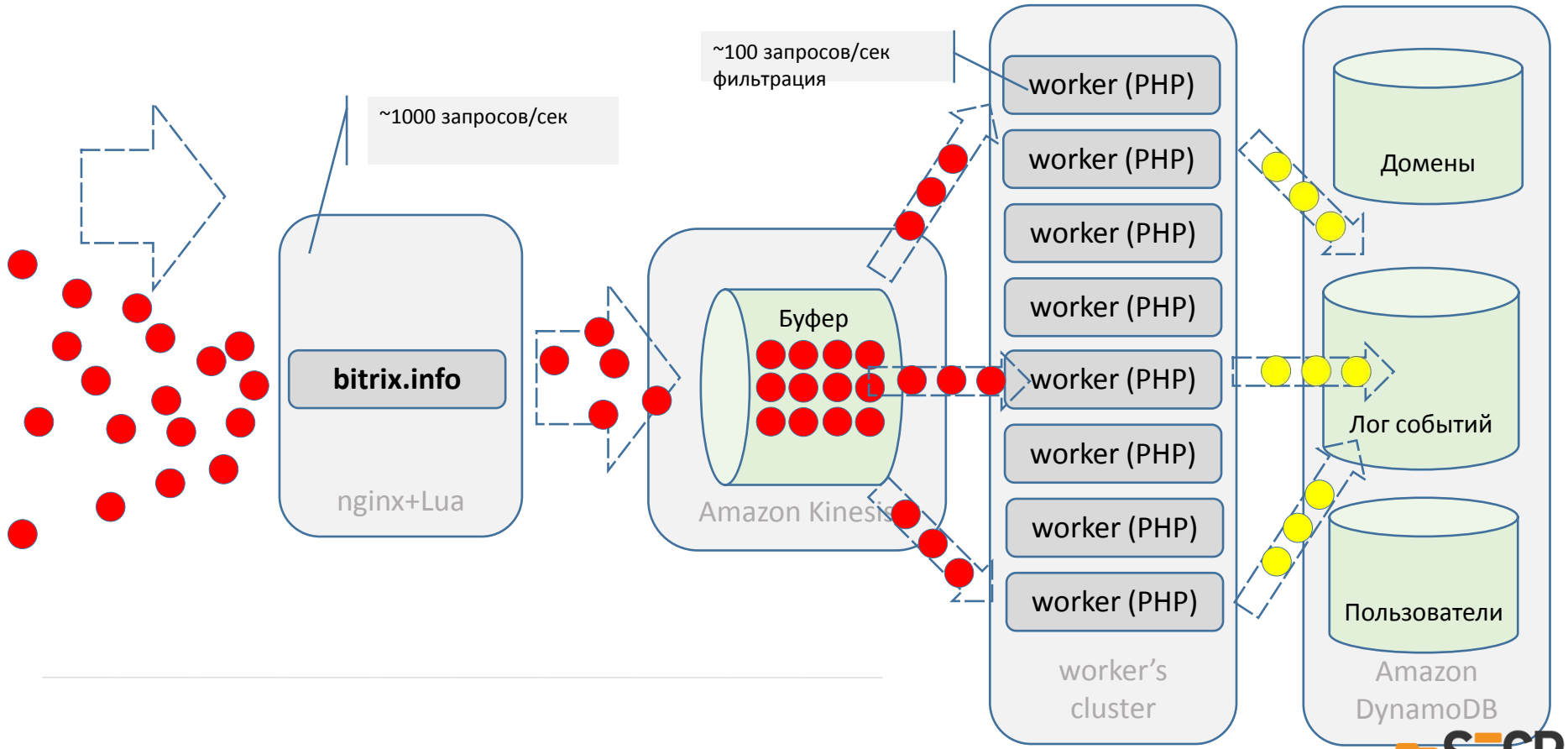
- Гистограмма:
- - Время пребывания клиента в разделе сайта
- - Число платных подписок в зависимости от числа пользователей услуги



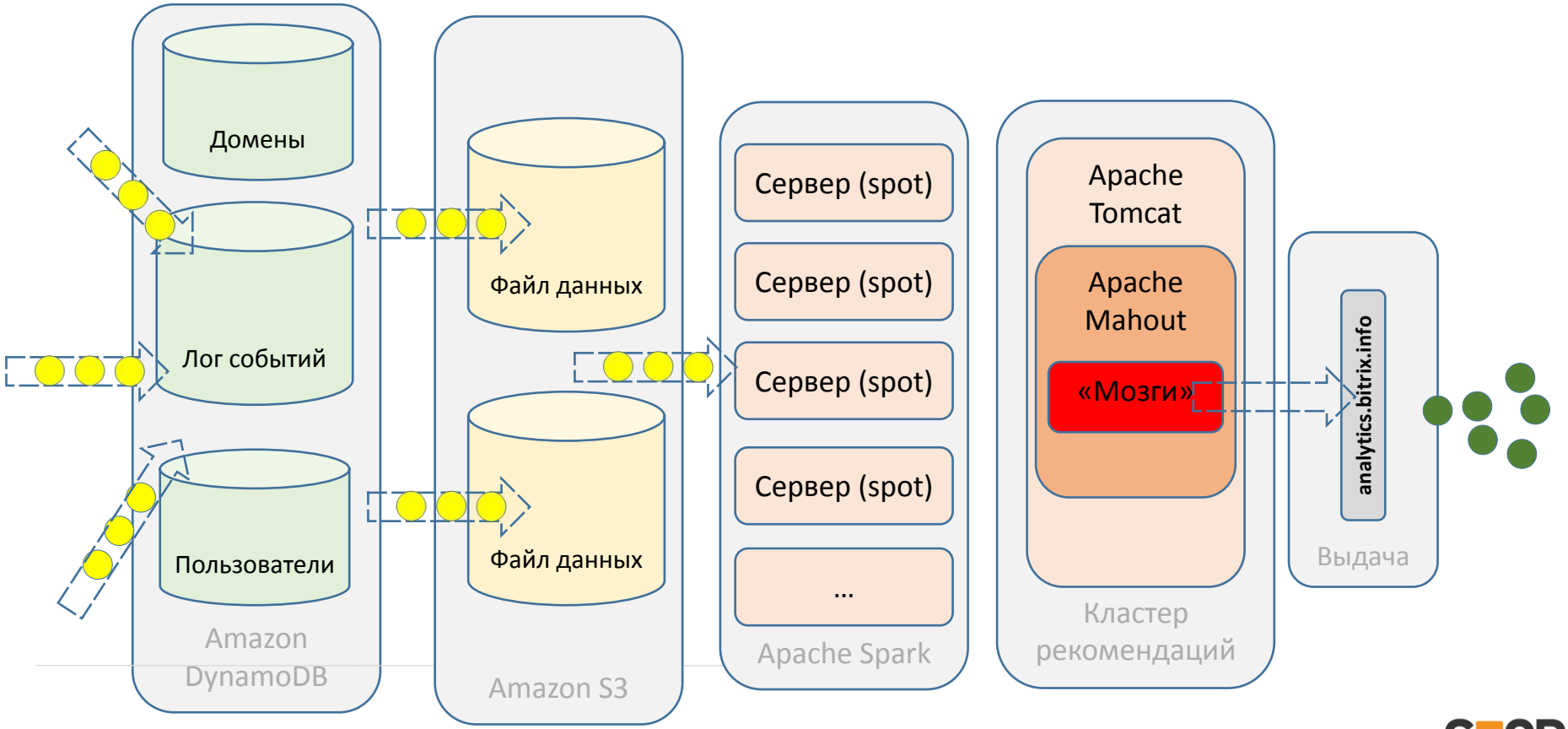
# Сервис «Скорость сайта» клиента на Битрикс

1. Собираем хиты из Amazon Kinesis в Redis
2. Хит содержит метрики js navigation timing.
3. Храним последние 1250 хитов (redis list)
4. Удаляем 20% самых долгих
5. Рассчитываем медиану времени отображения страницы в кластере
6. Отдаем на карту, jsonp, RemoteObjectManager

# BigData – «под капотом». Регистрация событий.



# BigData – «под капотом». Обработка, анализ, выдача.



# Сервис «Скорость сайта»

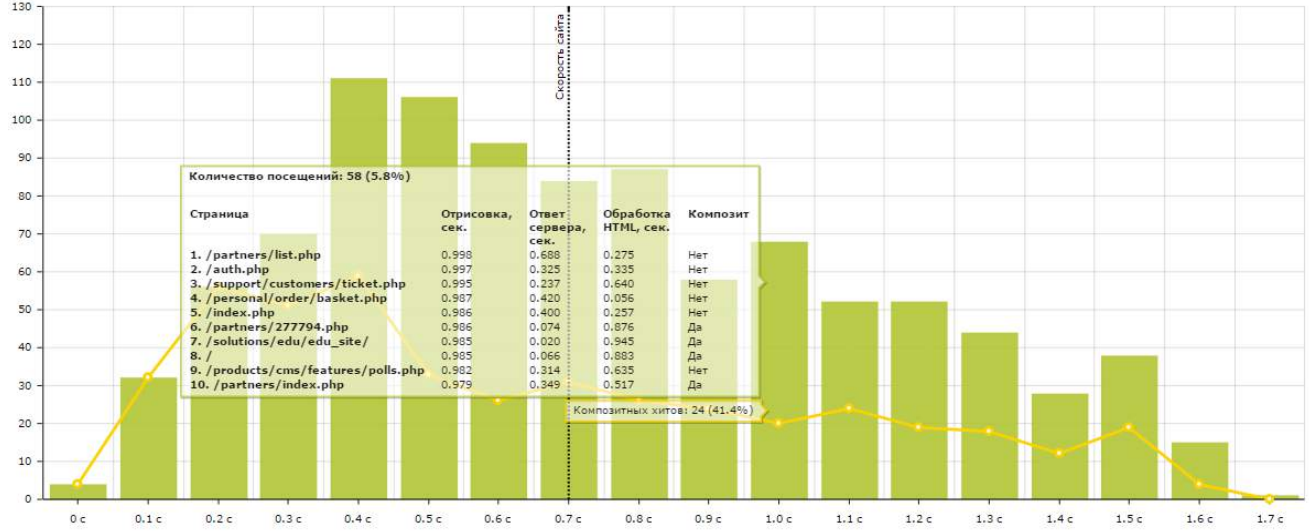
Скорость сайта: **Быстро (0.73 сек.)**


Обработано хитов: **1000** | За период: **21 Июня 13:24 - 21 Июня 13:48** | Композитных хитов: **458 (45.8%)**

**Скорость сайта** — комплексный показатель комфортности работы с сайтом для посетителей.
Учитывает качество разработки сайта, качество хостинга и доступность сайта по сети. Рассчитывается для 1000 последних посетителей вашего сайта.
Скорость сайта фактически показывает, как быстро отобразился ваш сайт для большинства из этих 1000 посетителей.

Монитор производительности: **15.57** Композитный сайт: **включено** Ускорение сайта (CDN): **включено**

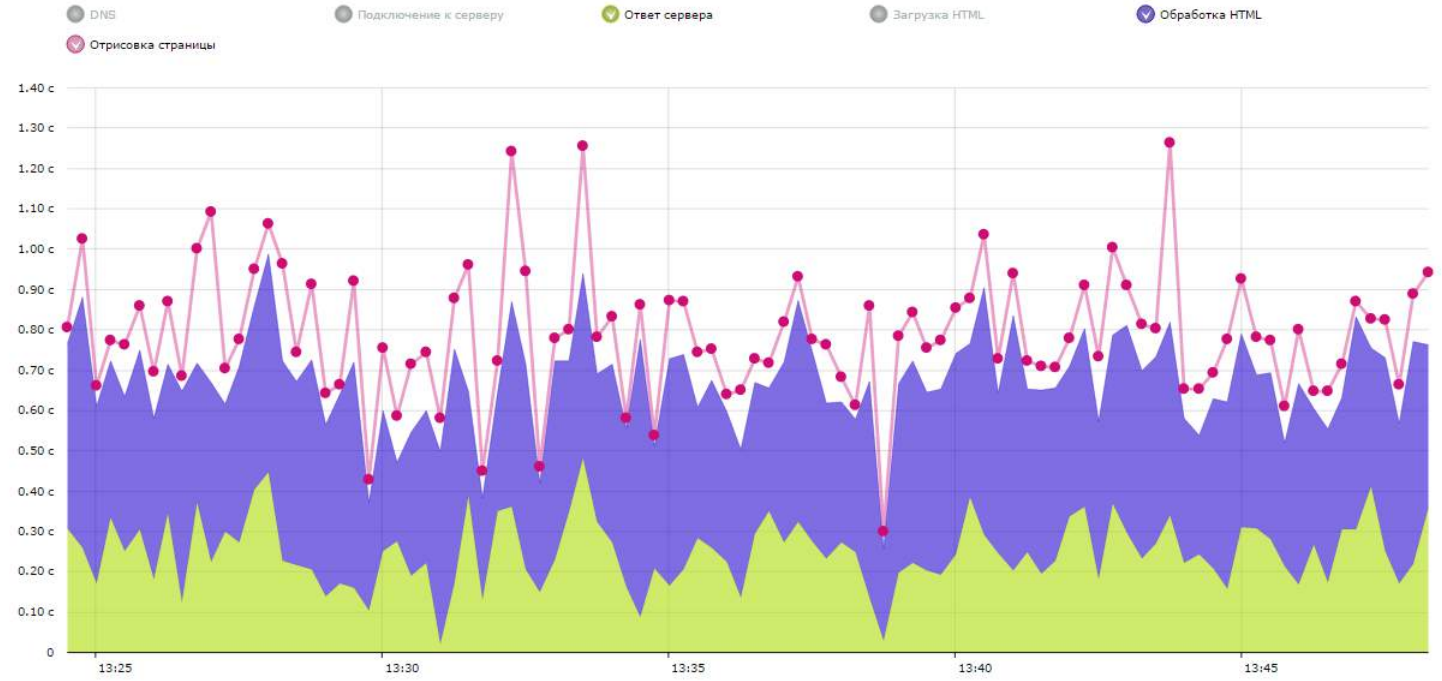
## Распределение скорости сайта по времени



# Сервис «Скорость сайта»

1С·БИТРИКС

## Последние посещения сайта

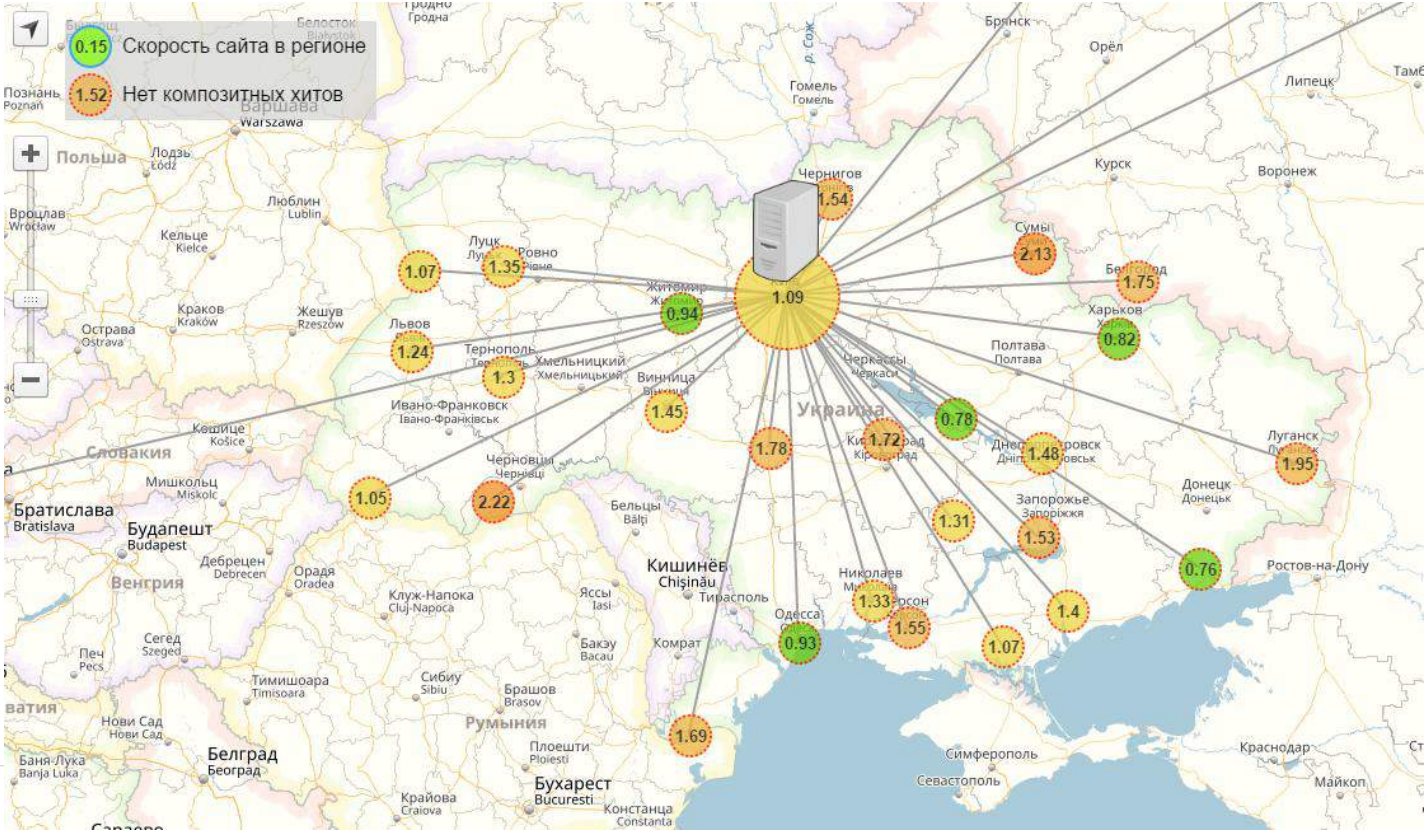



**Отрисовка страницы** — время от начала перехода на страницу до появления её на экране. Именно по этому показатели считается **Скорость сайта**.
**DNS** — время выполнения запроса DNS для страницы.
**Подключение к серверу** — сколько времени компьютер пользователя устанавливает соединение с сервером.
**Ответ сервера** — время обработки сервером запроса пользователя (включая время реакции сети для местоположения пользователя).
**Загрузка HTML** — время загрузки HTML страницы без ресурсов (картинки, CSS, Javascript).
**Обработка HTML** — время, в течение которого браузер обрабатывал содержимое страницы (синтаксический анализ HTML, CSS, обработка элементов JavaScript и отображение страницы) после загрузки её с сервера и до начала отрисовки.

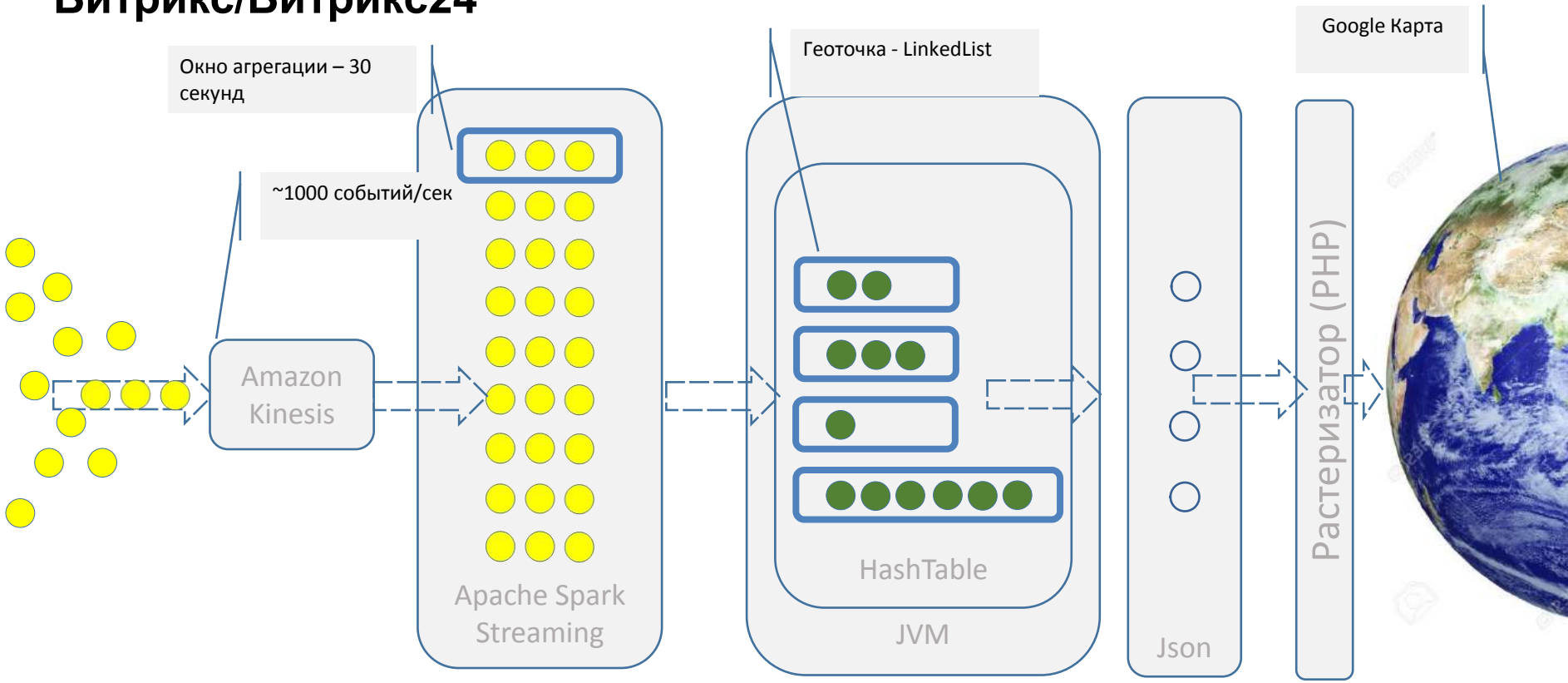
SECR

# Сервис «Скорость сайта»

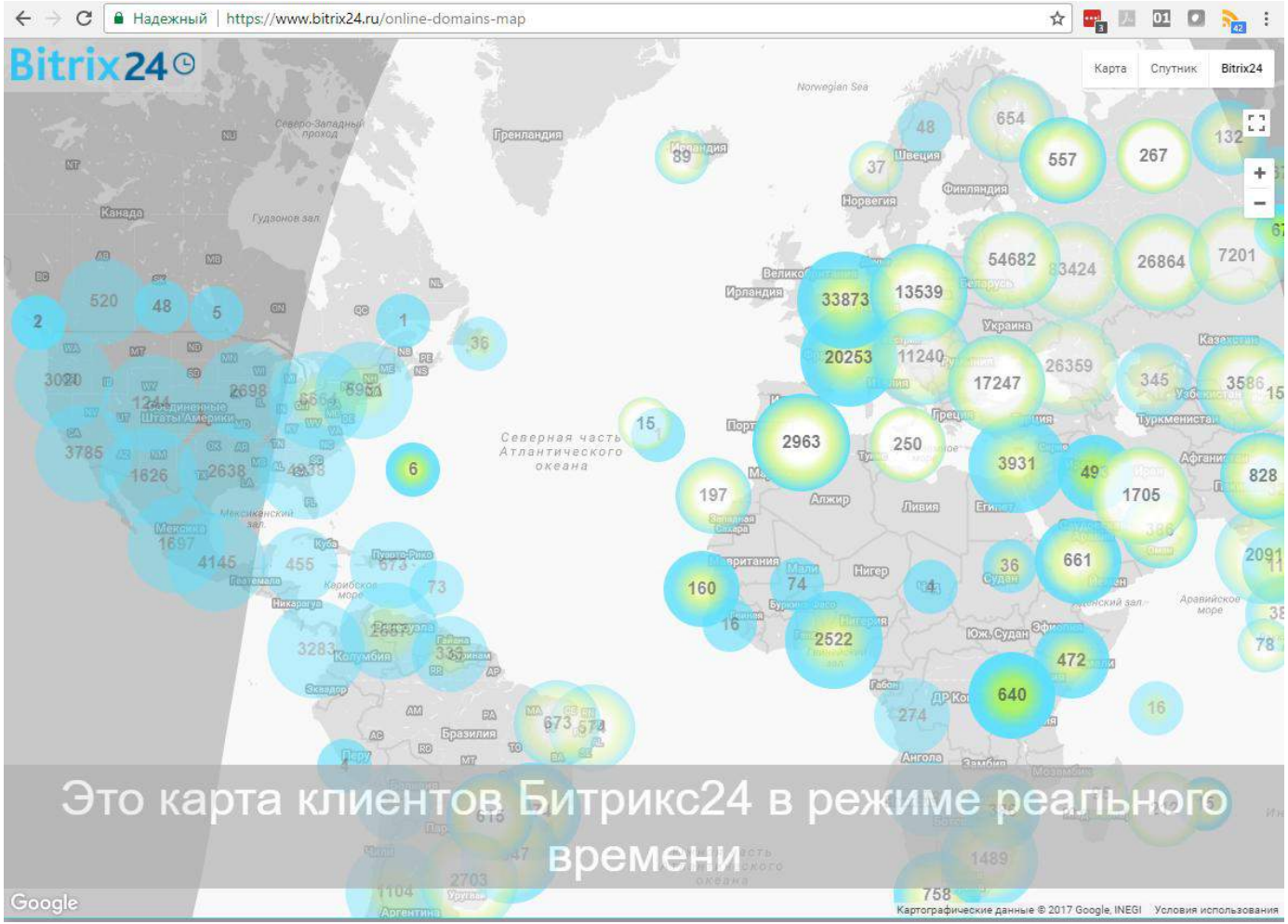
# Сервис «Скорость сайта»



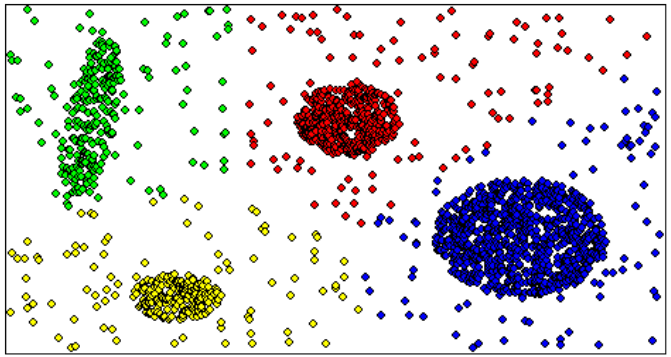
# Архитектура карты активности клиентов Битрикс/Битрикс24



# Архитектура карты активности клиентов Битрикс24

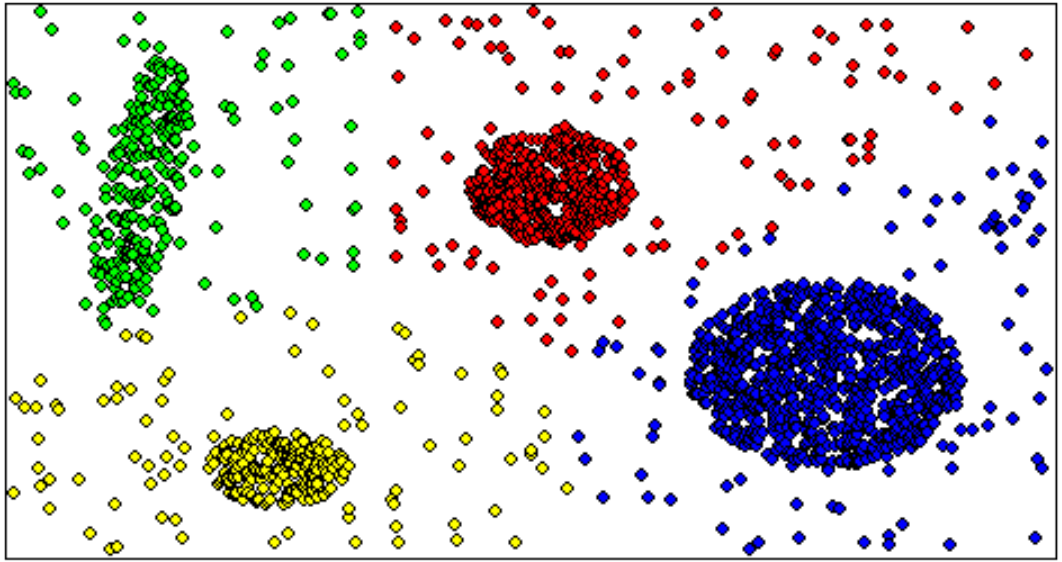


# Кластерный анализ



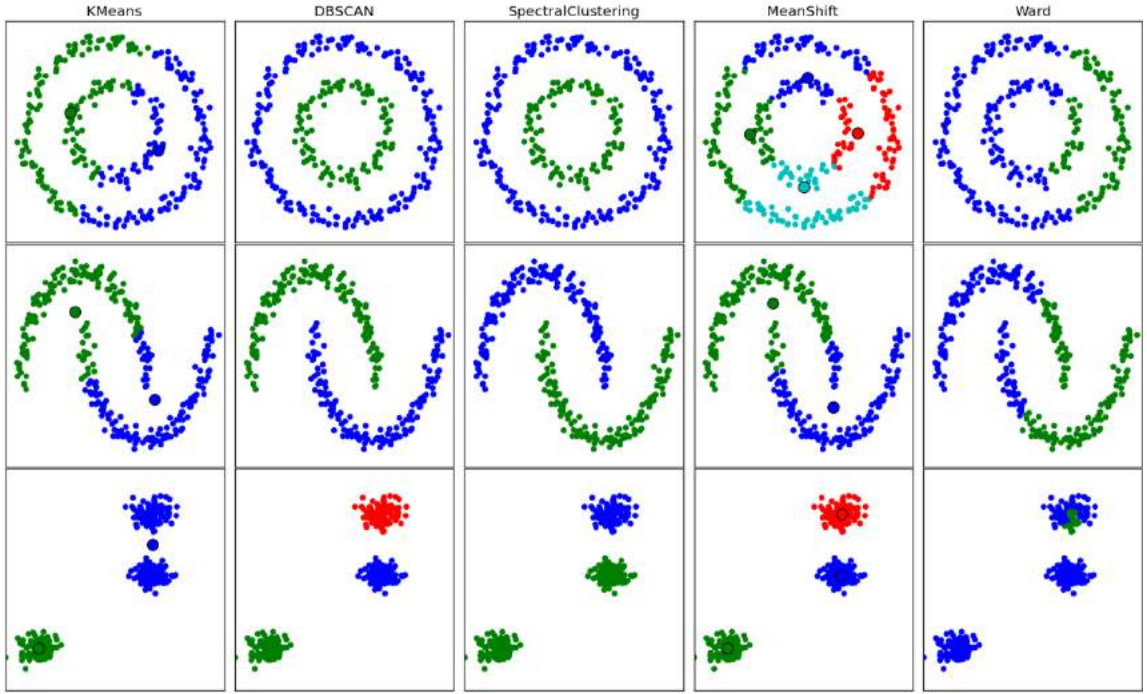
# Кластерный анализ

- Когда измерений много
- Если «повезет»
- Четкая/нечеткая
- Иерархическая
- Графы
- Данных много/мало
- Интерпретация



# Кластерный анализ

- Иерархическая
- K-means
- C-means
- Spectral
- Density-based (DBSCAN)
- Вероятностные
- Для «больших данных»



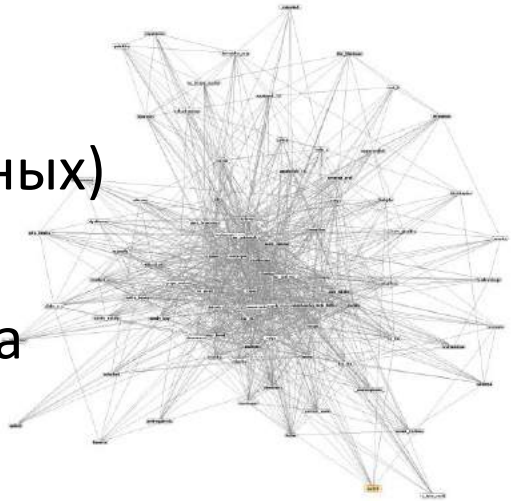
# Кластерный анализ – бизнес-кейсы

- Сегментация клиентов, типов использования сервиса, …
- Кластеризация «общего» товарного каталога
- Кластеризация графа связей сайтов (пересечение аудитории)

- Маркетинг работает с целевыми группами, информация разбита на «смысловые облака».

# Кластерный анализ – оценки на программирование

- Данные должны быть уже собраны
- Анализ в Rapidminer (0.1-2 часа)
- Анализ в Spark Mllib (1-2 дня, много данных)
- Анализ в scikit-learn – аналогично (мало данных)



- На выходе: список кластерных групп, иногда визуализация.
- Метрики качества кластеризации.

# Кластерный анализ – риски

- Много данных – медленно!
- Тексты, каталоги товаров …
- Как интерпретировать?

- Рецепты:
- Spark MLlib, векторизация текста, LSH (locality sensetive hashing), word2vec

# Кластерный анализ в Битрикс

- События использования инструментов на Битрикс24: задачи, вики, видеозвонки, календарь, чаты, поиск…
- Агрегация метрик по пользователям: Amazon Kinesis -> Amazon DynamoDB -> s3 -> Apache Spark
- Apache Spark Mllib, стандартный k-means – не взлетает, O(n^3)
- «Иерархический k-means с усреднением», Scala, Spark
- На выходе 3-4 группы = типа использования продукта

# Коллаборативная фильтрация – «сжатие» Товаров

- «Единый» каталог
- Склеить дубликаты
- Передать «смысл» между Товарами
- Улучшить качество персональных рекомендаций
- Семантическое сжатие размерности, аналог матричной факторизации
- Скорость
- Ранжирование результатов

# Minhash

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}.$$

- Min-wise independent permutations locality sensitive hashing scheme
- Снижаем размерность
- Совместима с LSH (следующий слайд)

Pr[ hmin(A) = hmin(B) ] = J(A,B)

- Размер сигнатуры: 50-500

simhash

| $Row$ | $S_1$ | $S_2$ | $S_3$ | $S_4$ | $x+1 \mod 5$ | $3x+1 \mod 5$ |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 2 | 4 |
| 2 | 0 | 1 | 0 | 1 | 3 | 2 |
| 3 | 1 | 0 | 1 | 1 | 4 | 0 |
| 4 | 0 | 0 | 1 | 0 | 0 | 3 |

# Text shingling

- Shingle – «черепица»
- Устойчивость к вариантам, опечаткам

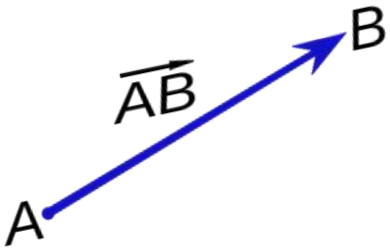
«Штаны красные махровые в полоску»

{«штан», «таны», «аны », «ны к», «ы кра», «крас», …}

«Красные полосатые штаны»

# Векторизация описания Товара

- Текст: «Штаны красные махровые в полоску»
- Вектор «bag of words»: [0,0,0,1,0,…0,1,0] – ~ 10000 -1000000 элементов (kernel hack)
- Minhash-сигнатура после shingling:
- [1243,823,-324,12312,…] – 100-500 элементов, совместима с LSH



# Locality-Sensitive Hashing (LSH)

- Вероятностный метод снижения размерности
- Использовали для minhashed-векторов
- Banding:

b – корзины, r – элементов в корзине.

P{ “Векторы совпадут хотя-бы в одной корзине” }:

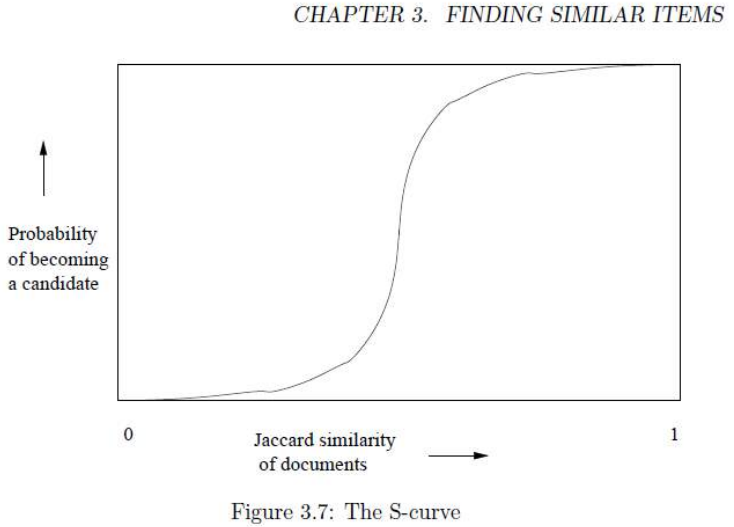
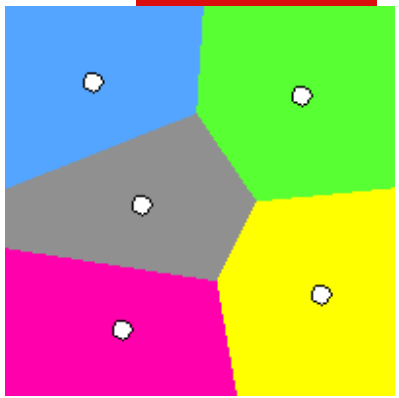


Figure 3.7: The S-curve

An approximation to the threshold is $(1/b)^{1/r}$

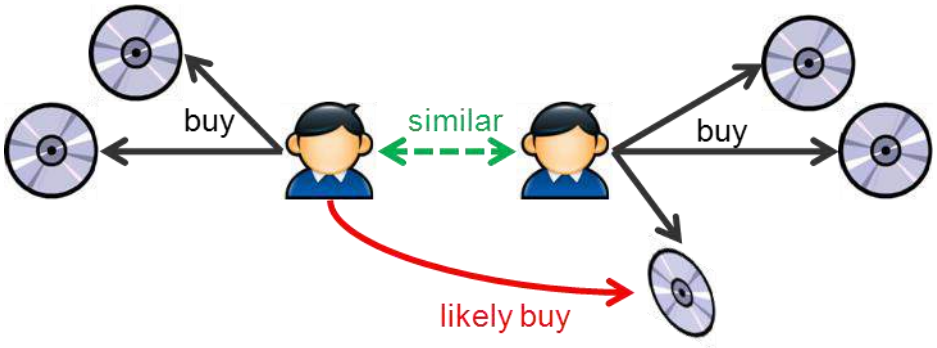
# Кластеризация каталога



- Apache Spark
- 2-3 часа, 8 spot-серверов
- 10-20 млн. Товаров => 1 млн. кластеров
- Адекватные по смыслу кластера
- Персональные рекомендации - стали в разы «лучше»
- DynamoDB – хранение кластроидов

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| application_1437483006918_0346 | hadoop | MahoutModelsBuilderClustered | SPARK | default | Sun, 06 Sep 2015 12:00:12 GMT | Sun, 06 Sep 2015 14:39:44 GMT | FINISHED | SUCCEEDED | History |

# Персонализация

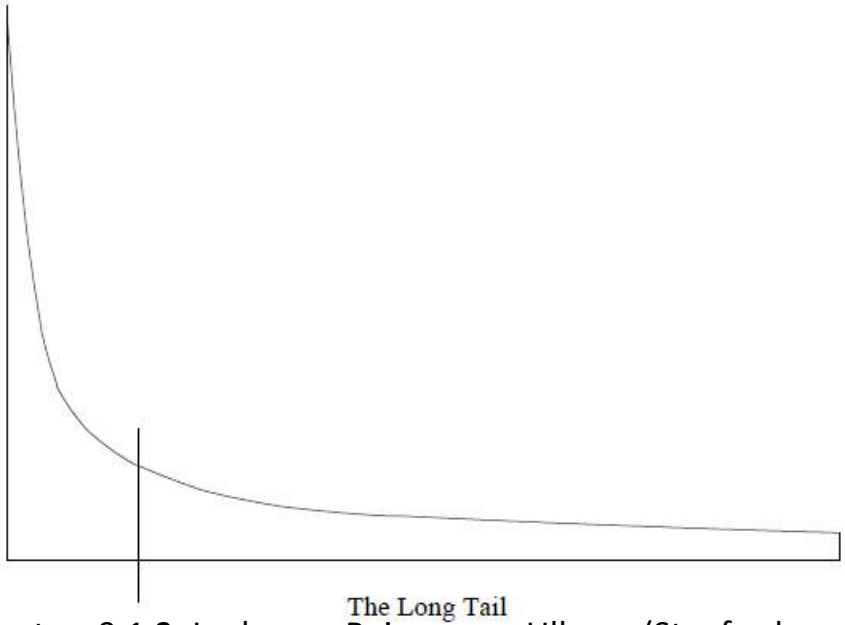


# Персонализация

- Релевантный контент – «угадываем мысли»
- Релевантный поиск
- Предлагаем то, что клиенту нужно как раз сейчас
- Увеличение лояльности, конверсии

# Объем продаж товаров

- Best-sellers
- Топ-продаж…
- С этим товаром покупают
- Персональные рекомендации




«Mining of Massive Datasets», 9.1.2: Leskovec, Rajaraman, Ullman (Stanford University)

# Коллаборативная фильтрация

- Предложи Товары/Услуги, которые есть у твоих друзей (User-User)
- Предложи к твоим Товарам другие связанные с ними Товары (Item-Item): «сухарики к пиву»

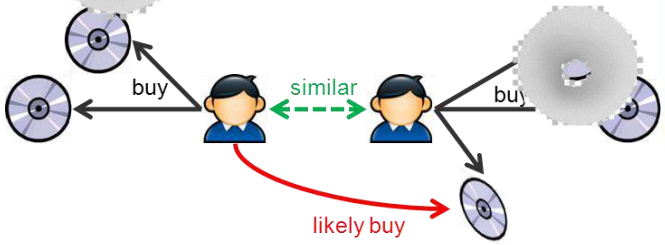
# Как работает коллаборативная фильтрация

Матрица:

- Пользователь
- Товар

Похожи[illegible]льзователи

[illegible]вары




| | $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ | $M_6$ |
|---|---|---|---|---|---|---|
| $U_1$ | ✓ | ✓ | ✓ | ✓ | | |
| $U_2$ | ✓ | | ✓ | ✓ | | |
| $U_3$ | | ✓ | | | | ✓ |

# Возможности коллаборативной фильтрации (Item-Item)

- Персональная рекомендация (рекомендуем посмотреть эти Товары)
- С этим Товаром покупают/смотрят/… (глобальная)
- Топ Товаров на сайте

# Коллаборативная фильтрация (Item-Item) – сроки, риски

- Apache Spark MLlib (als), Apache Mahout (Taste) + неделька
- Объем данных
- Объем модели, требования к «железу»

# Коллаборативная фильтрация (Item-Item) в Битрикс

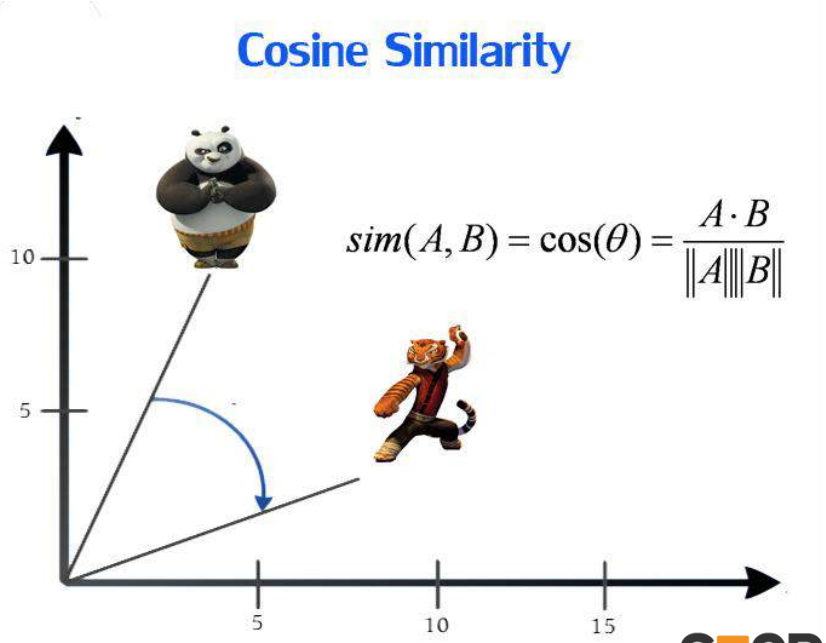
- Были модели по 10-20 миллионов Пользователей/Товаров
- Фильтрация, создание датасета для матрицы: s3->Apache Spark job.->Apache Mahout Taste
- Оставили модель 10 на 10 миллионов для «похожести» Товара на Товар
- Хак: популярный Товар на сайте через похожесть
- Проблема холодного старта

# Content-based рекомендации

- Купил пластиковые окна – теперь их предлагают на всех сайтах и смартфоне.
- Купил Toyota, ищу шины, предлагают шины к Toyota

# Content-based рекомендации – реализация, риски

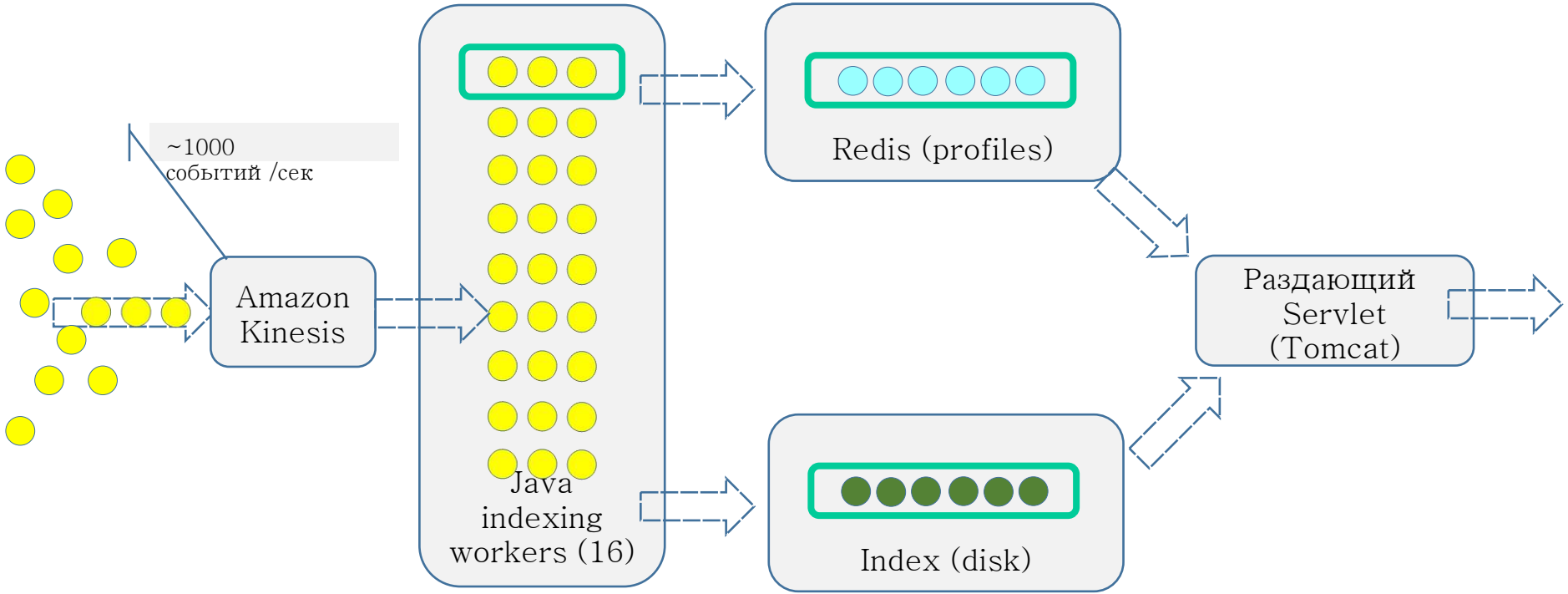
- Поисковый «движок»: Sphinx, Lucene (Solr)
- «Обвязка» для данных
- Хранение профиля Клиента
- Реализация: неделька. Риски – объем данных, языки.



# Content-based, collaborative рекомендации - разумно

- Рекомендовать постоянно «возобновляемые» Товары (молоко, носки, …)
- Рекомендовать фильм/телевизор – один раз до покупки
- Учет пола, возраста, размера, …

# Content-based рекомендации – в Битрикс



# Content-based рекомендации – в Битрикс

- Профиль Пользователя: десятки тэгов
- Стемминг Портера
- Высокочастотные и «специальные» слова
- Алгоритмы вытеснения тэгов
- Куда можно развивать… (word2vec, glove, синонимы …)

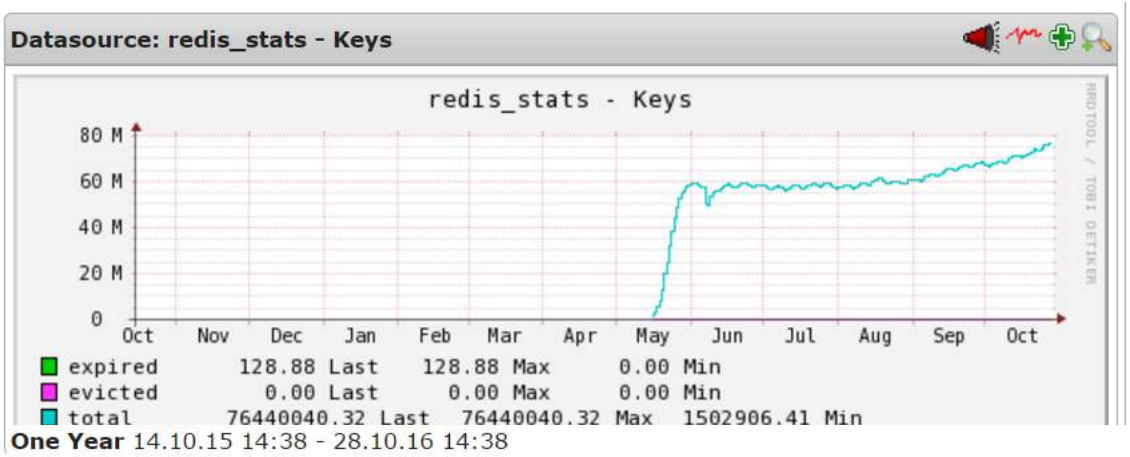
# Content-based рекомендации – в Битрикс

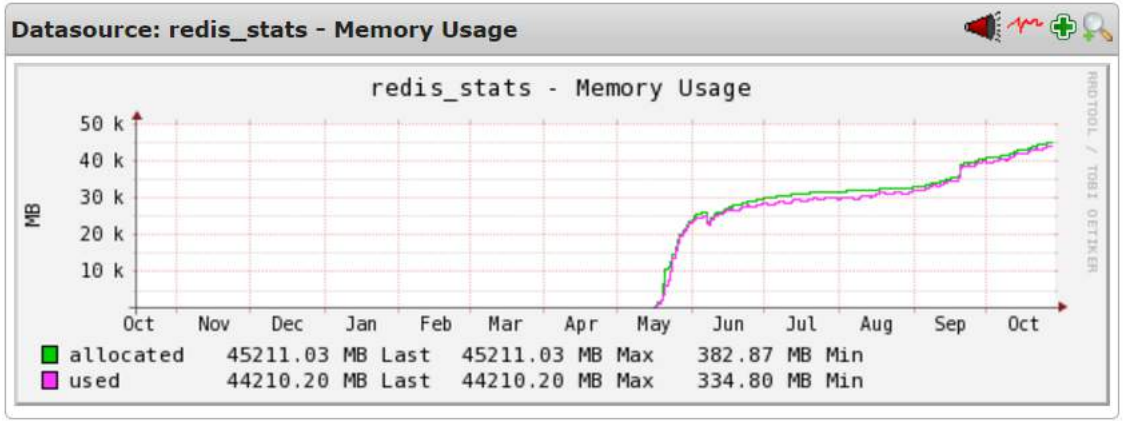
- Многопоточный индексатор, java/lucene
- Amazon Kinesis – как буффер
- Индекс в папке на диске, вытеснение
- Как реализован “онлайн”
- Раздающий Servlet

# Content-based рекомендации – в Битрикс

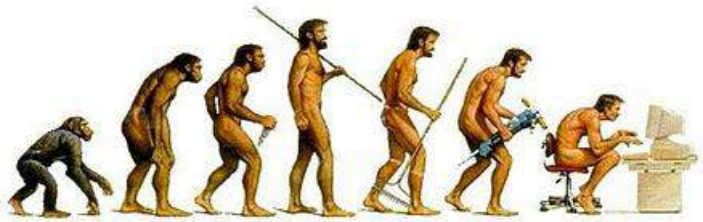
- “Потребители”: десятки тысяч интернет-магазинов
- “Поставщики”: все сайты на Битрикс, больше 100к
- Тэги Профиля: название страницы, h1
- Индекс Товаров: название, краткое описание, разделы
- Индекс: гигабайты, сотни файлов в папке

# Content-based рекомендации – в Битрикс



# Классификация

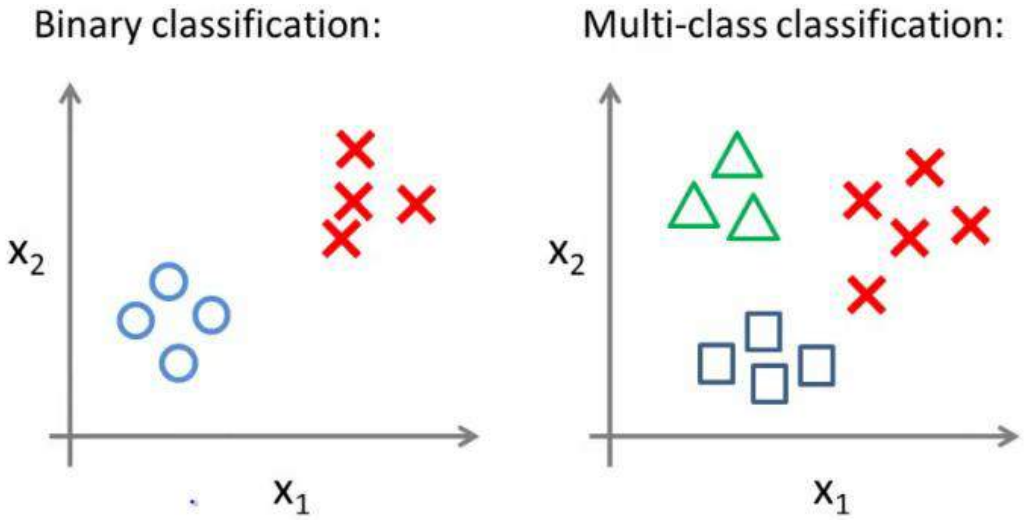


# Классификация

Разбиваем по группам,

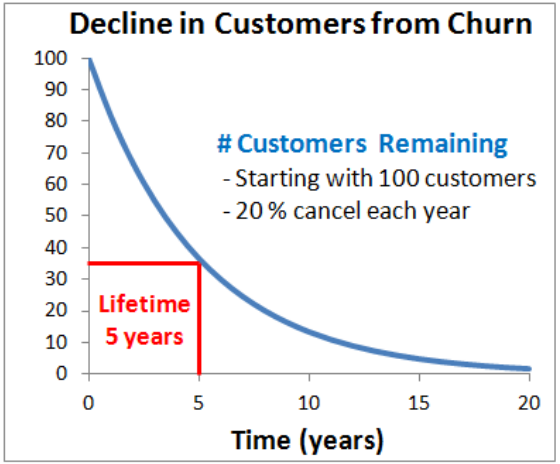
обучение

- Бинарная
- Мультиклассовая



# Классификация – бизнес-кейсы

- Удержание: найти клиентов, которые скоро уйдут (churn-rate)
- Найти клиентов, готовых стать платными
- Найти клиентов, которые готовы купить новую услугу
- Найти готовых уволиться
- Определить у клиента – пол!



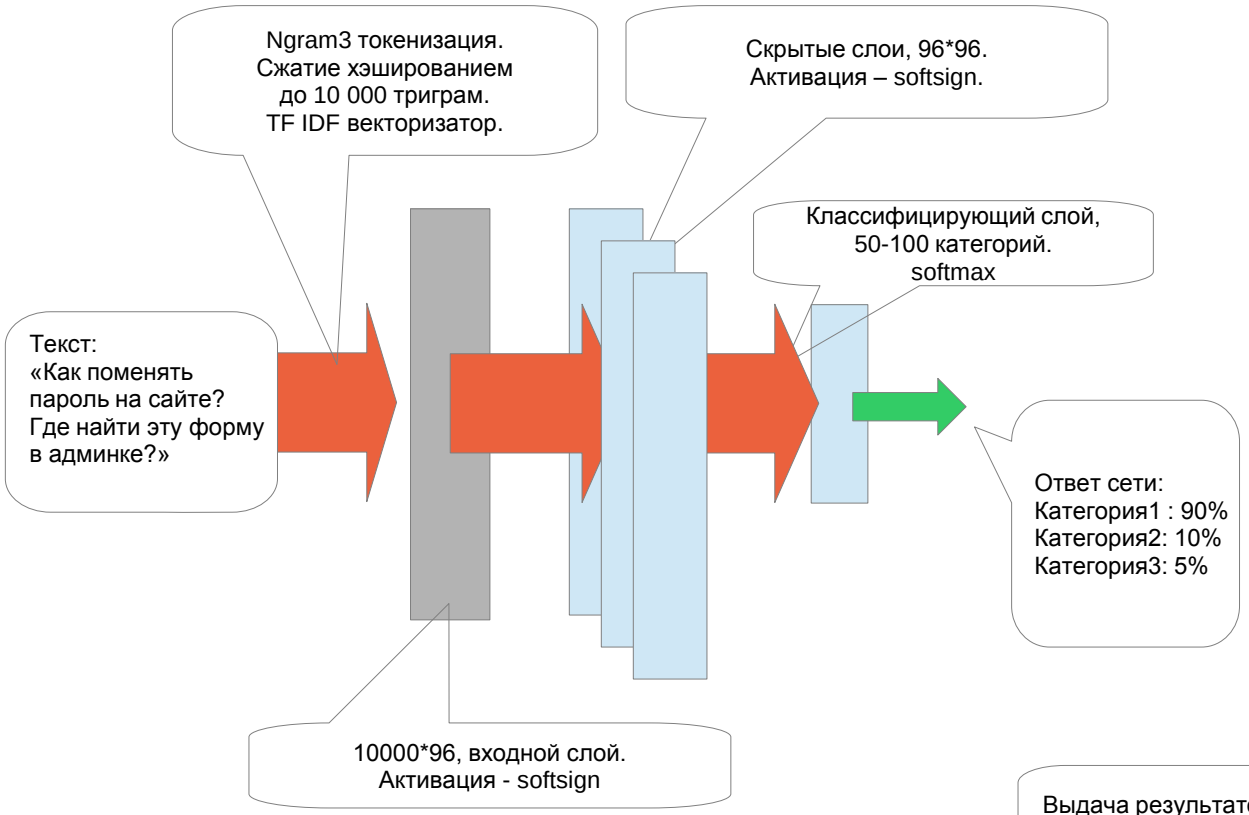
# Классификация – тонкости

- А как/чем удержать клиентов?
- Определение релевантных групп – зондирование (рассылки, опросы), база моделей
- Оценка качества моделей

# Классификация – реализация, риски

- Определение, нормализация атрибутов
- Feature engineering
- Выбор алгоритма, kernel
- Spark MLlib, scikit-learn – 2-3 дня
- Rapidminer – полчаса

# Классификатор обращений техподдержки




Глубокий классификатор, использующий ngrams3 векторизатор и сжимающее хэширование входного вектора.

Используем взвешенную cost-функцию для балансирования неравномерного числа примеров в категориях. Иногда добавляем сверточные слои. Иногда лучше работают рекуррентные слои.

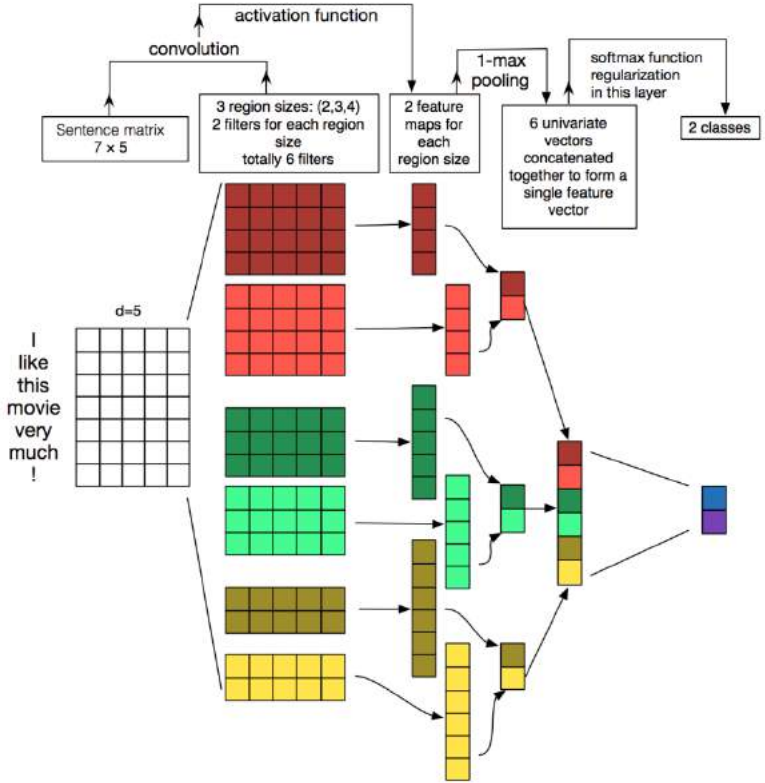
Drop out: 0.85, l2: 0.001, learning rate: 0.1, adam, batch=128. В обученной сети: 1-3 миллиона параметров.

Фреймворк: deeplearning4j
Веб-сервер: jetty.
Нейронная сеть –
набор файлов на диске (10-20 МБ).
Кеширование сетей в памяти.

Выдача результатов
через json rest API

# 1D свертка для классификации текстов

- Глубокий аналог ngrams, очень быстрое обучение на GPU
- Word/char-based 1D convolution
- Пилотная сеть для техподдержки в Битрикс. Увеличение качества на 30%.



# Классификатор обращений техподдержки Битрикс24

**# 1** 15.06.2017 22:18:01
Михаил Вопросченко [клиент]

*Управление сайтом > Перенос, резервное копирование*

**Текст ошибки или описание проблемы:**
Сайт удалили вместе со всем содержимым. Пытаюсь восстановить из облачного хранилища, требуется ключ но я не могу его найти

**# 2** 15.06.2017 22:18:01
- Категория: Резервное копирование и перенос

**# 3** 15.06.2017 22:18:01

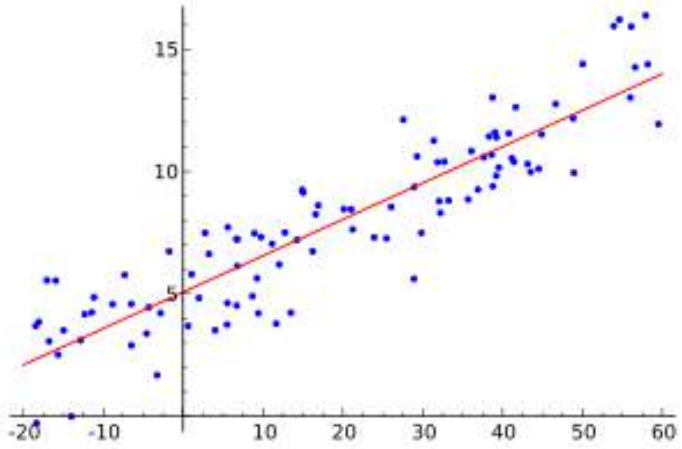
[987377] резервное копирование и перенос: 89%
[987327] главный настройка: 8%
[987296] обновления: 2%

**# 4** 15.06.2017 22:18:01
[auto]
- ответственный: Валентин Помощников [техподдержка]

# Классификация – компании-клиенты Битрикс24

- Кто из бесплатников станет платником?
- Кто из платников уйдет?
- Больше 2.5 миллиона зарегистрированных компаний
- Сбор счетчиков (десятки метрик) и агрегация в Apache Spark
- Классификация, Apache Spark MLlib, логистическая регрессия с регуляризацией
- Выгрузка моделей в админки для маркетинга, рассылки, конверсия на 2-3 процента выше
- Минус: небольшой охват с уверенным предсказанием

# Регрессия

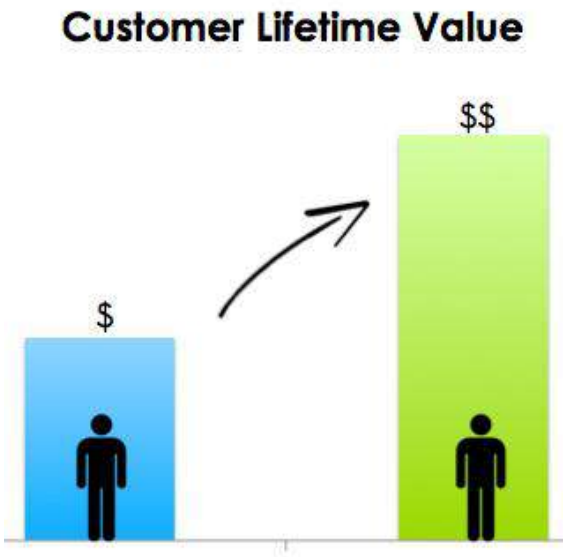


# Регрессия

- Предсказать «циферку»
- Стоимость квартиры, автомобиля на рынке
- Ценность клиента для магазина
- Зарплата на данную вакансию
- и т.д.

# Регрессия – customer lifetime value (CLV)

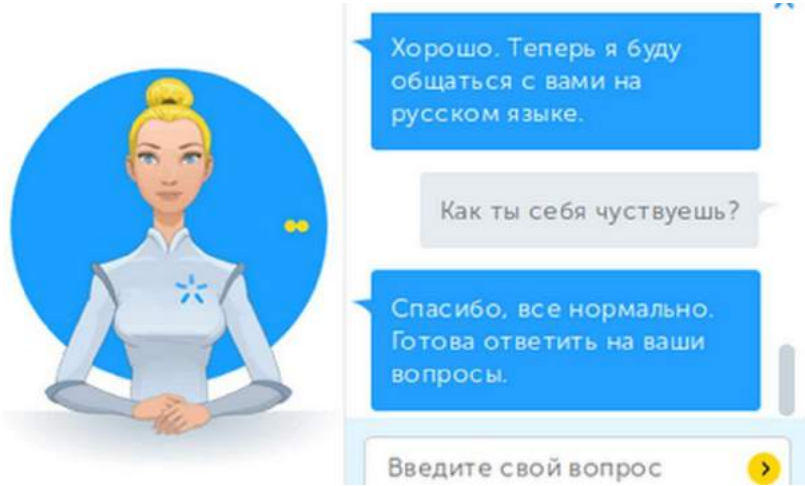
- Пришел клиент, а он потенциально прибыльный!
- Система лояльности, удержания
- Подарки, скидки, …



# Регрессия – реализация, риски

- Выявление атрибутов
- Выбор алгоритма
- Spark MLlib – не работает, scikit-learn – 1-2 дня
- «Регрессия» в Битрикс24 – CLV сделали через классификацию по дискретным ступеням

# Чатботы


Хорошо. Теперь я буду общаться с вами на русском языке.
Как ты себя чуствуешь?
Спасибо, все нормально. Готова ответить на ваши вопросы.
Введите свой вопрос

# Наши эксперименты

Наши эксперименты:

**Л.Н. Толстой, "Война и мир"**

Число слоев сети: 3
Число нейронов в каждом слое: 400
Коэффициент встряхивания "мозгов" (dropout): чуть больше единицы
Память сети: 150 символов назад
Число параметров, которые мы учим - несколько миллионов

> - Ежели бы мы мог видеть меня в государство, а ему нужно забыть.

> - Andre, je ne partont, j'ai la vaintu qui qu'on russe profende, [Пускай граф и капитан Полковник сказал, что надо делать Кутузовского французского ученица, - говорил он и всегда говорил:

> - Вы не шуби, про положение, - сказал Тихон. - Маменька, господа, - повторил бы насильно вошедший за сеном, - я тебе говог'ю, - сказал Наполеон, так же как и он, взяв свою голову через город, чтобы одно неловко. Старик вздрогнул рукой с штабными голосами, которые стала после полного генерала, указывал он белый длинной улыбкой, стоя в гостиной от одного зуба. Пьер был не при для него теплому. Он был между разговором и невольно присел сепеть; но сосредоточил войско Пьера. Прежде стоит показать что-то. Поворотливый мальчик черного и княжен и говорил об этом, неясное, уверенное и старались изволить все, не чувствуя скорее свое места. Жюли проводила от гибни и довольный, не пытьма, развшед, смеялся; но их было силы и разноречивые своими или мыслью, которые были сделаны из победа, а в доме стояло бравный последние пленных произведенных крестов и деревни. Разговор и старательно сосредоточил то, что он скажет, он не отвечал ей больное с силами и упреков, не переставая помешался в одной минут Петя, ожидая всего того, что сказал при ней и сказал это, прежде их нет; но на этом горы проявилось вперед, в то время как Даву отдавал те слова, сказав очень хорошо, что почти входила в Брюнн и быстро перебирались от усталого, как кто движением голос Земя, которым любил ограничен в эту минуту. В первое время оделялся с ним и велел княжна Марья решительная глаза с большой и гордыми правительством человека и видела повинойствия.

# Наши эксперименты

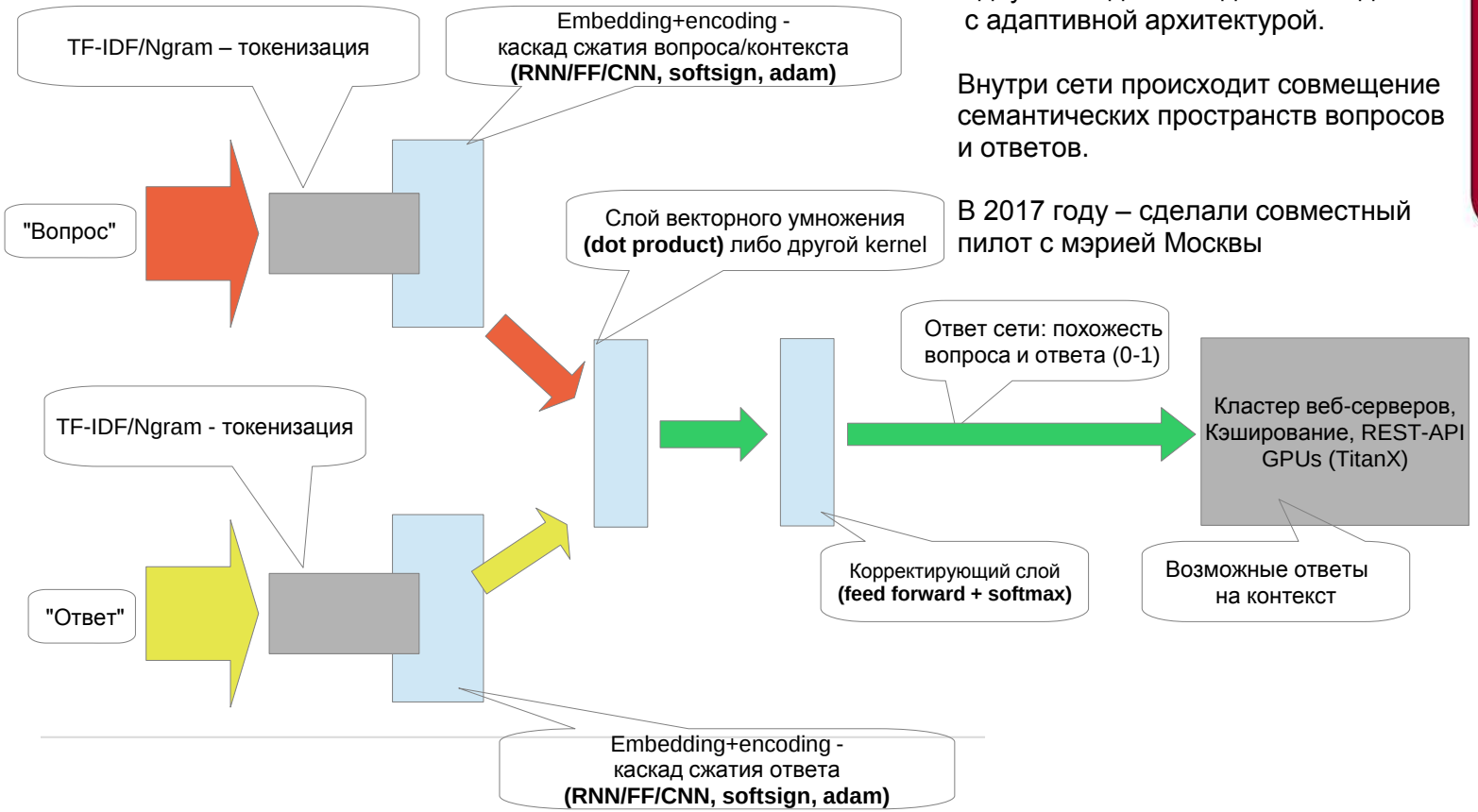
Наши эксперименты:

**Код ядра Битрикс**

3-х слойная сеть, размер слоя: 400 нейронов, несколько миллионов параметров, память: 150 символов назад, обучение - ночь

```
function GetGroupsCache($SITE_ID, $site_groups=false)
{
        $_SESSION["SEARCH"] = $arGroups;
        global $USER_FIELD_MANAGER, $USER_AGENT;
        $groups = Array();
        foreach($GLOBALS["SORT"] as $group_id)
        {
                $group_by_id = htmlspecialcharsbx($group_id);
                if ($group_by==".group")
                        $group_id = $group_id;
        }
}

//returns we log old group leave
if(is_array($GROUP_CODE_COUNT))
        $CACHE_MANAGER->Set("USER_CAL_EDIT", $zr);

//Check ID in Rating
if($CURRENT
|| $group_by==0 && check_bitrix_sessid())
{
        COption::SetOptionString('socialnetwork', 'email_provider', 'Y');
        InitBVar($_REQUEST['admin_save_relation']);

        if($group['START_COMMAND_TEXT']=='N')
        {
                $CURRENT_USER_RIGHT = $APPLICATION->GetGroupRight("statistic");
                $discount_value = intval($GLOBALS["USER"]->GetID());
                $runtime->Update($groupId, array("delete filter" => true));
```

# «Нейробот»




Глубокая нейронная сеть
с двумя входами и одним выходом
с адаптивной архитектурой.

Внутри сети происходит совмещение семантических пространств вопросов и ответов.

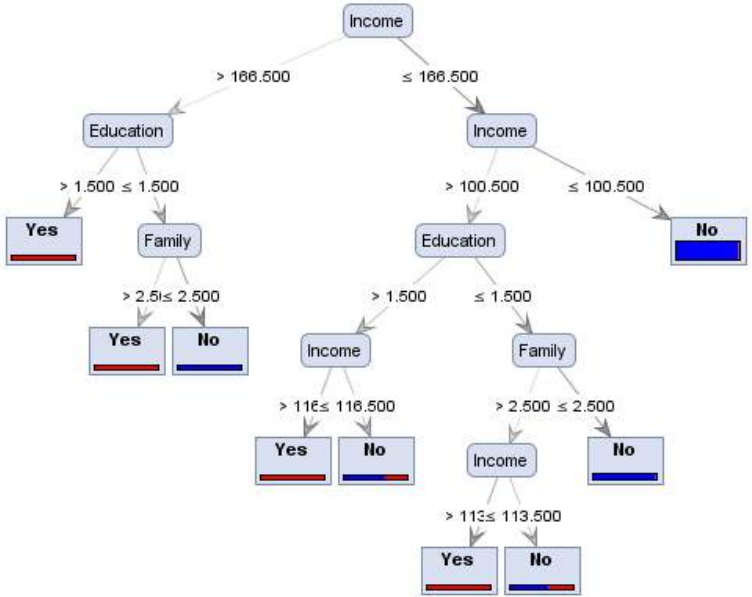
В 2017 году – сделали совместный пилот с мэрией Москвы

$$A \cdot B = |A|\,|B| \cos\theta$$

# Анализ причин

# А что влияет на конверсию в …?

- Собираем данные (хиты, логи, анкетирование)
- Строим дерево решений
- В Rapidminer – полчаса
- В Spark MLlib – чуть больше.
- CatBoost?
- Анализ графов перемещений пользователей
- Классификация графов?
- InfoGAN?



# Стратегии увеличения прибыли

# Стратегии

- Изучаем клиентов (кластерный анализ, зондирование)
- Привлечь нового дороже чем удержать старого?
- Высокий churn-rate и CLV – удерживаем релевантным предложением
- Меньше «тупого» спама - больше лояльность
- Персонализированный контент
- **Ранжирование лидов и просчет рисков в CRM** (Sales Force Einstein)

# Интересные тренды и техники

- Semi-supervised learning. Когда данных мало…
- One-shot learning
- Переобучение
- Neural turing machine/memory networks
- Attention

# Выводы

- Можно брать готовые модели в фреймворках и применять в различных бизнес-задачах уже сейчас
- Собирать данные не сложно – главное аккуратно ☺
- Все быстро меняется, нужно учиться
- Инженерные практики в компании – очень важны

Спасибо за внимание!
Вопросы?

**Александр Сербул**
@AlexSerbul
Alexandr Serbul
serbul@1c-bitrix.ru