

Как решать и побеждать в соревнованиях по анализу данных?

Евгений Путин
Университет ИТМО
putin.evgeny@gmail.com

25 мая 2011
Санкт-Петербург

ROUND ONE

FIGHT

Платформы по соревнованиям в анализе данных (DM)



АлгоМост



CodaLab



INNOCentive



CHALEARN



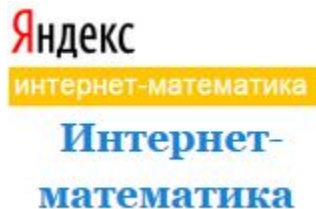
KDDCup



kaggle



DRIVENDATA



Преимущества Kaggle

- ❖ Наиболее раскрученная платформа
- ❖ Возможность запускать in-class соревнования
- ❖ Крутые соревнования от топовых IT компаний
- ❖ Красивый, простой сайт
- ❖ Большое количество участников
- ❖ Богатый форум
- ❖ Датасеты
- ❖ Тutorials
- ❖ Скрипты
- ❖ Поиск работы в Data Science



kaggle

Крутой профайл на kaggle.com отличная строчка в резюме для каждого data scientist-а и не только 😊

Как выглядят соревнования по DM?

otto group

Completed • \$10,000 • 3,514 teams

Otto Group Product Classification Challenge

Tue 17 Mar 2015 – Mon 18 May 2015 (19 months ago)

Dashboard

- Home
- Data
- Make a submission

Information

- Description
- Evaluation
- Rules
- Prizes
- Timeline

Forum

Kernels

- New Script
- New Notebook

Leaderboard

- Public
- Private

Private Leaderboard

1. Gilberto Titericz & Stanislav Semenov
2. _(ツ)_/
3. i dont know
4. Dmitry & Davut
5. Mikhail Trofimov
6. Optimistically Convergent
7. x2x4x8

Competition Details » [Get the Data](#) » [Make a submission](#)

Classify products into the correct category

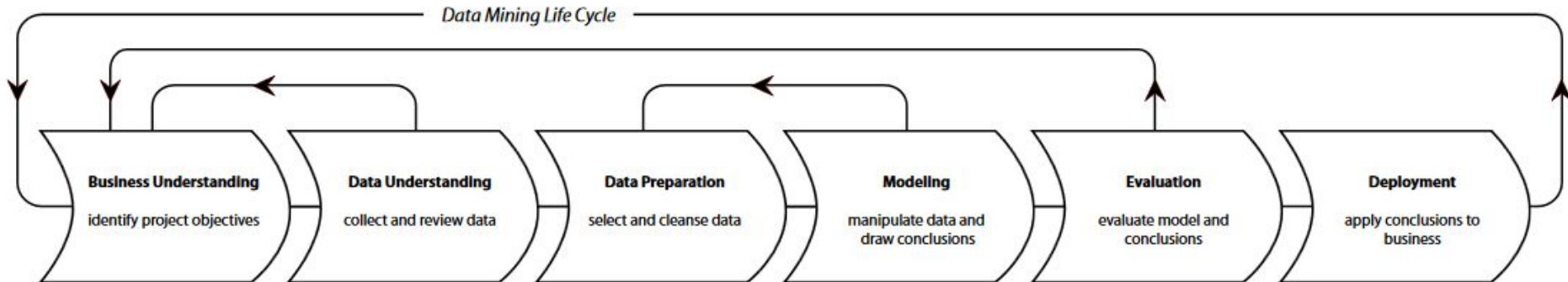
[Get started on this competition through Kaggle Scripts](#)

The Otto Group is one of the world's biggest e-commerce companies, with subsidiaries in more than 20 countries, including Crate & Barrel (USA), Otto.de (Germany) and 3 Suisses (France). We are selling millions of products worldwide every day, with several thousand products being added to our product line.

A consistent analysis of the performance of our products is crucial. However, due to our diverse global infrastructure, many identical products get classified differently. Therefore, the quality of our product analysis depends heavily on the ability to accurately cluster similar products. The better the classification, the more insights we can generate about our product range.



Цикл решения задач DM



Determine Business Objectives

*Background
Business Objectives
Business Success Criteria
(Log and Report Process)*

Assess Situation

*Inventory of Resources,
Requirements, Assumptions,
and Constraints
Risks and Contingencies
Terminology
Costs and Benefits
(Log and Report Process)*

Determine Data Mining Goals

*Data Mining Goals
Data Mining Success Criteria
(Log and Report Process)*

Produce Project Plan

*Project Plan
Initial Assessment of Tools and
Techniques
(Log and Report Process)*

Collect Initial Data

*Initial Data Collection Report
(Log and Report Process)*

Describe Data

*Data Description Report
(Log and Report Process)*

Explore Data

*Data Exploration Report
(Log and Report Process)*

Verify Data Quality

*Data Quality Report
(Log and Report Process)*

Data Set

*Data Set Description
(Log and Report Process)*

Select Data

*Rationale for Inclusion/
Exclusion
(Log and Report Process)*

Clean Data

*Data Cleaning Report
(Log and Report Process)*

Construct Data

*Derived Attributes
Generated Records
(Log and Report Process)*

Integrate Data

*Merged Data
(Log and Report Process)*

Format Data

*Reformatted Data
(Log and Report Process)*

Select Modeling Technique

*Modeling Technique
Modeling Assumptions
(Log and Report Process)*

Generate Test Design

*Test Design
(Log and Report Process)*

Build Model Parameter Settings

*Models
Model Description
(Log and Report Process)*

Assess Model

*Model Assessment
Revised Parameter
(Log and Report Process)*

Evaluate Results

*Align Assessment of Data
Mining Results with
Business Success Criteria
(Log and Report Process)*

Approved Models

*Review Process
Review of Process
(Log and Report Process)*

Determine Next Steps

*List of Possible Actions
Decision
(Log and Report Process)*

Plan Deployment

*Deployment Plan
(Log and Report Process)*

Plan Monitoring and Maintenance

*Monitoring and
Maintenance Plan
(Log and Report Process)*

Produce Final Report

*Final Report
Final Presentation
(Log and Report Process)*

Review Project

*Experience
Documentation
(Log and Report Process)*

Понимание задачи

- ❖ Определение и формулирование бизнес задачи
- ❖ Оценка рисков, затрат, общего профита
- ❖ Постановка DM целей
- ❖ Определение критериев успешности
- ❖ Выработка плана решения задач
- ❖ Четкое понимание того что надо предсказать



Понимание данных: Первый шаг

- ❖ Сбор данных
- ❖ Какие данные есть: сколько примеров, сколько признаков, какие признаки по природе
- ❖ Достаточно ли данных для решения задачи, есть ли необходимость собирать дополнительные данные



Понимание данных: Второй шаг

- ❖ Описательные статистики

$$\mu_x = \frac{\sum_{i=1}^n x_i}{n} \quad s = \sqrt{\frac{\sum_{i=1}^n (x_i - \mu_x)^2}{n-1}}$$

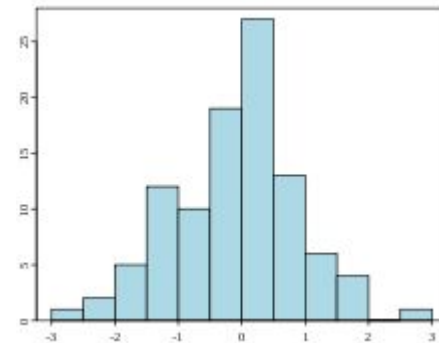
- ❖ Коэффициент корреляции

$$\frac{\sum_{i=1}^m (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^m (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^m (y_i - \bar{y})^2}} \quad \text{cov}(x, y)$$

- ❖ Проверка статистических гипотез (нормальность, проверку на распределение)

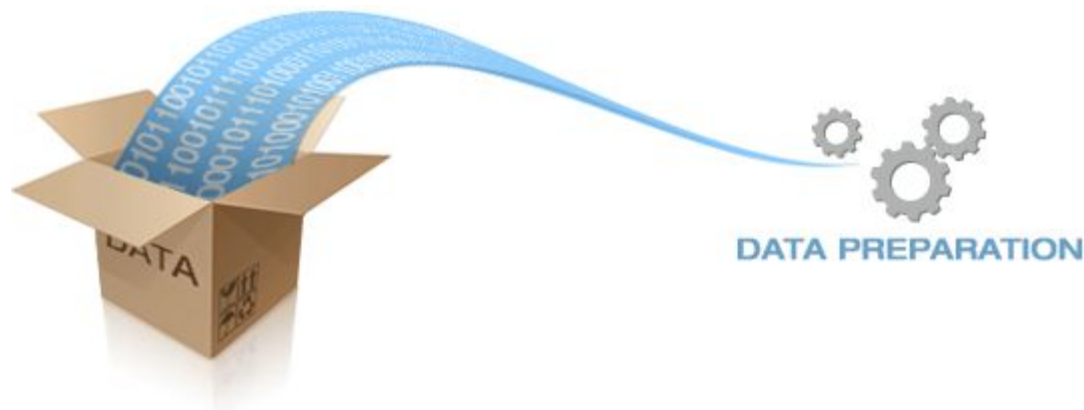
- ❖ Исследование распределений: гистограммы признаков, таргетов

$$F = \frac{D_1}{D_2} = \frac{\sigma_1^2}{\sigma_2^2}$$



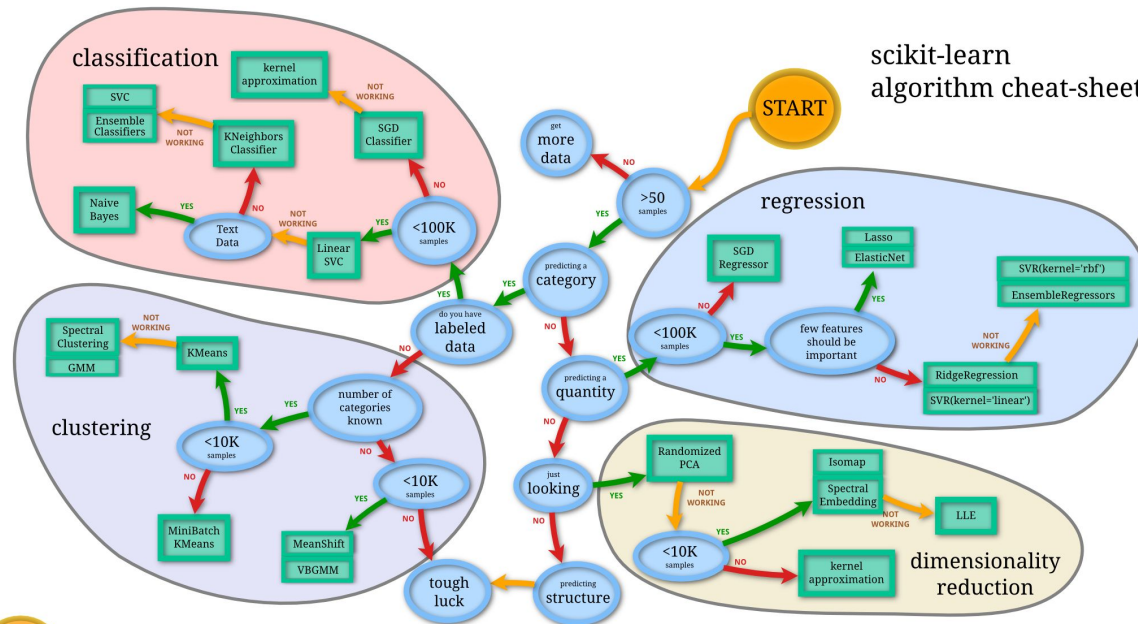
Подготовка данных

- ❖ Выбор и интеграция данных
- ❖ Форматирование данных
- ❖ Предобработка данных: заполнение пропусков, определение выбросов, нормализация данных, т.д.
- ❖ Выбор/экстракция признаков, сокращение размерности
- ❖ Инженерия признаков
- ❖ Разбиение на тренировочное, тестовое множества



Построение моделей

- ❖ Выбор подходящих моделей, соответствующих проверяемым гипотезам
- ❖ Определение дизайна тестирования
- ❖ Обучение моделей с настройкой гиперпараметров
- ❖ Контроль overfitting/underfitting



Оценка качества моделей

- ❖ Анализ эффективности моделей на тестовом множестве:
статистические гипотезы, корреляции
- ❖ Вычисление метрик оценки качества моделей
- ❖ Проверка overfitting/underfitting
- ❖ Постпроцессинг
- ❖ Достигнут ли желаемый результат?



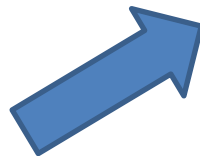
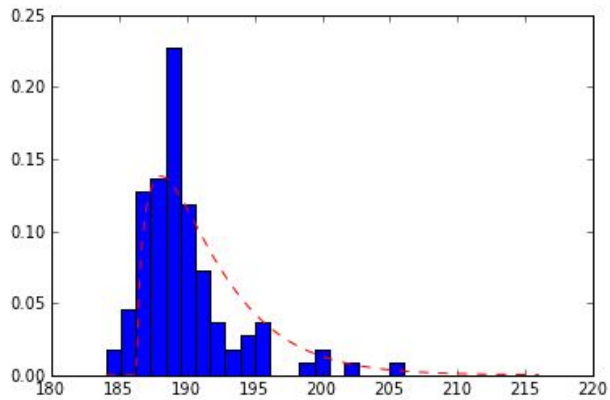
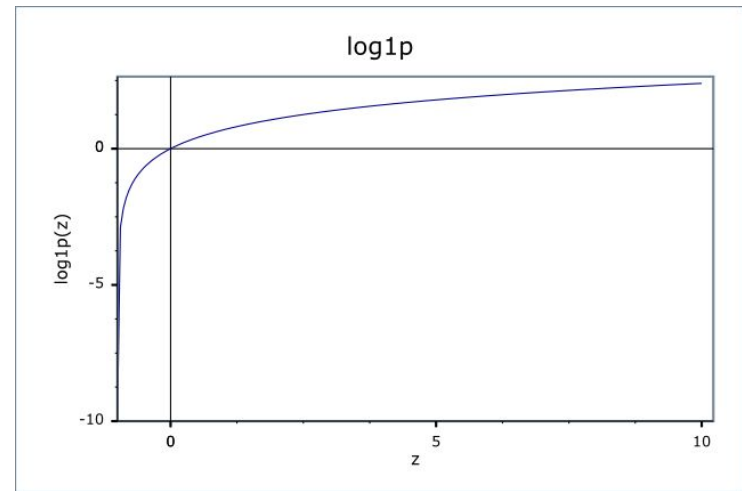
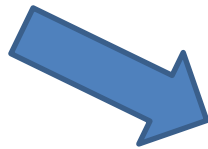
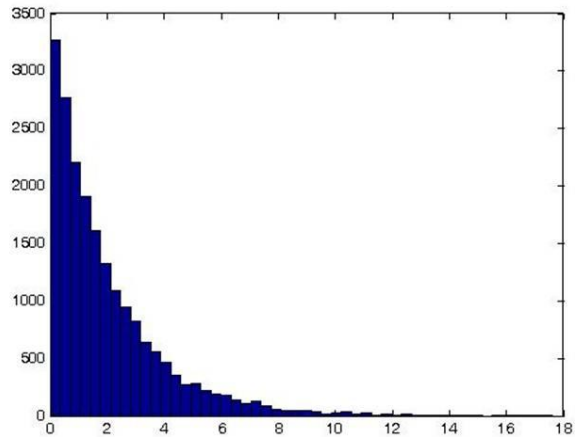
Развертывание системы

- ❖ Финальный отчет по проекту
- ❖ Выполнены ли все поставленные DM цели?
- ❖ Удовлетворяют ли результаты критерия успешности?



ROUND
TWO

Исследование распределений: Линеаризация



Предобработка данных

- ❖ Преобразование категориальных переменных: OneHotEncoder, LabelEncoder

```
>>> from sklearn.preprocessing import OneHotEncoder
>>> enc = OneHotEncoder()
>>> enc.fit([[0, 0, 3], [1, 1, 0], [0, 2, 1], [1, 0, 2]])
OneHotEncoder(categorical_features='all', dtype=<... 'numpy.float64'>,
              handle_unknown='error', n_values='auto', sparse=True)
>>> enc.n_values_
array([2, 3, 4])
>>> enc.feature_indices_
array([0, 2, 5, 9])
>>> enc.transform([[0, 1, 1]]).toarray()
array([[ 1.,  0.,  0.,  1.,  0.,  0.,  0.,  1.,  0.,  0.]])
```

```
>>> le = preprocessing.LabelEncoder()
>>> le.fit(["paris", "paris", "tokyo", "amsterdam"])
LabelEncoder()
>>> list(le.classes_)
['amsterdam', 'paris', 'tokyo']
>>> le.transform(["tokyo", "tokyo", "paris"])
array([2, 2, 1]...)
>>> list(le.inverse_transform([2, 2, 1]))
['tokyo', 'tokyo', 'paris']
```

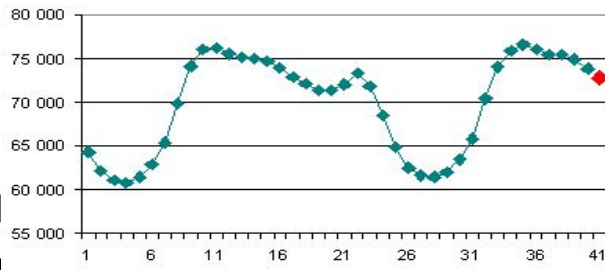
- ❖ Преобразование
- ❖ Преобразование

```
>>> from sklearn.feature_extraction import FeatureHasher
>>> h = FeatureHasher(n_features=10)
>>> D = [{'dog': 1, 'cat': 2, 'elephant': 4}, {'dog': 2, 'run': 5}]
>>> f = h.transform(D)
>>> f.toarray()
array([[ 0.,  0., -4., -1.,  0.,  0.,  0.,  0.,  0.,  2.],
       [ 0.,  0.,  0., -2., -5.,  0.,  0.,  0.,  0.,  0.]])
```

time, т.д.

Заполнение пропусков

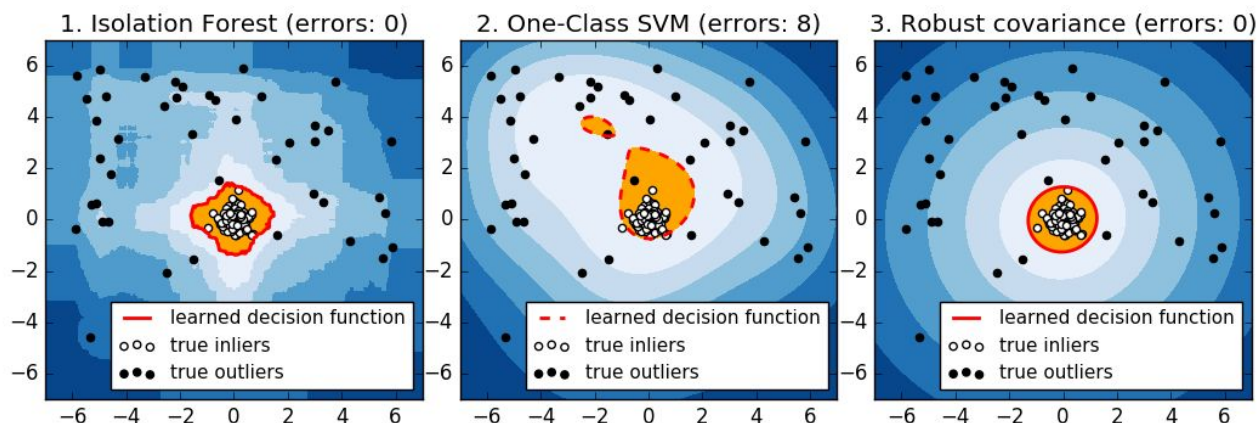
- ❖ Заполнение нулями
- ❖ Заполнение следующими, предыдущими значениями (`pandas.fillna`)
- ❖ Заполнение средними, модами, медианами (`sklearn.preprocessing.Imputer`)
- ❖ Заполнение с использованием (авто)регрессии/сезонных моделей



- ❖ Заполнение с использованием (авто)регрессии/сезонных моделей с помощью специальных адаптированных алгоритмов, например, ARIMA, T.D.

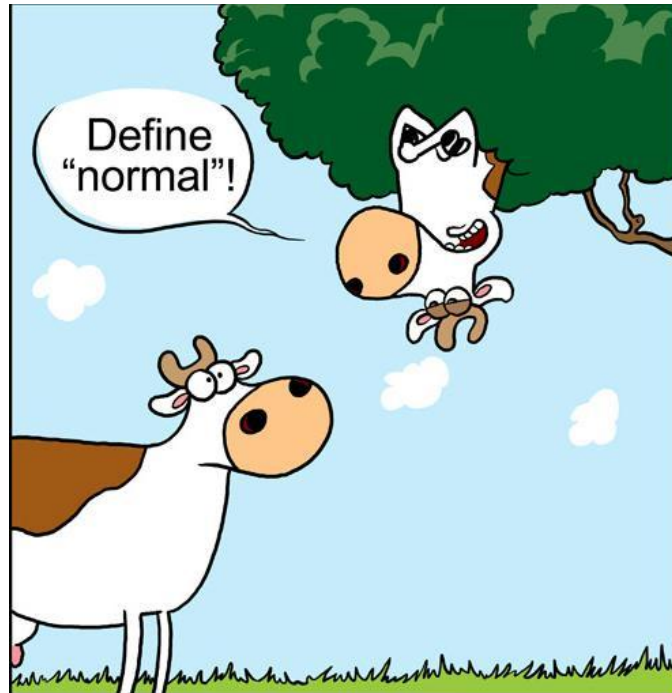
Определение выбросов

- ❖ Определение через распределения: по квантилям, перцентилям, по другим правилам пальца
- ❖ Определение через визуализацию, используя алгоритмы кластеризации (k-means, affinity propagation, ..), сокращения размерности (PCA, t-SNE, ..)
- ❖ Определение через анализ построенных моделей, постпроцессинг
- ❖ Определение через специальные адаптированные [алгоритмы](#):
[OneClassSVM](#), [EllipticEnvelope](#), [IsolationForest](#)



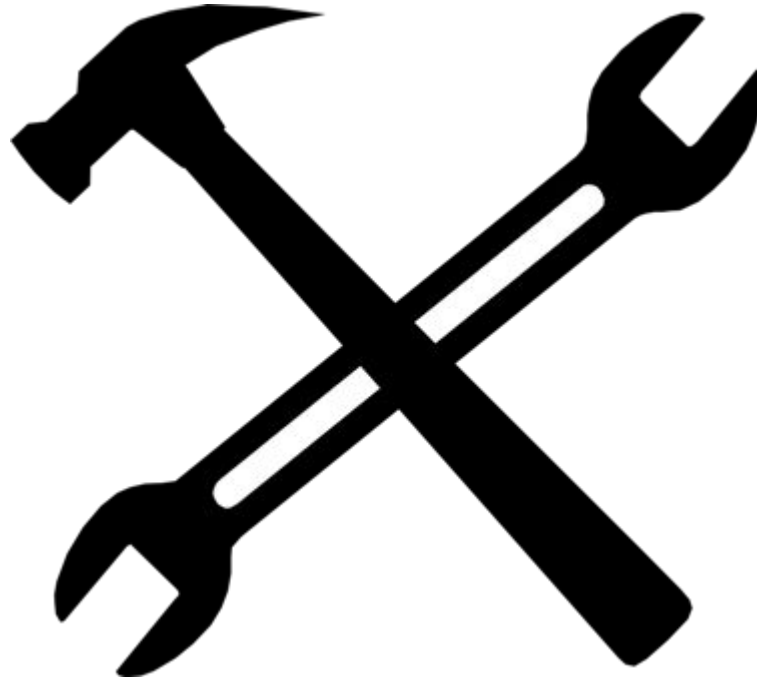
Нормализация данных

- ❖ Стандартная нормализация
- ❖ Нормализация в 0-1 или в -1, 1 (для нейронных сетей)
- ❖ Стемминг, лемматизация, TF-IDF и другие методы для текста
- ❖ [Нормализация](#) для звука
- ❖ Вычитание среднего по всем пикселям, нормализация цветов для картинок



Выбор признаков

- ❖ Выбор через model-free методы: [scikit-feature](#)
- ❖ Статистики (`sklearn.feature_selection.SelectKBest`)
- ❖ Корреляции Пирсона, Спирмена
- ❖ Выбор через model-based методы:
 - ❖ RandomForest (`rf.feature_importances_`, PFI)
 - ❖ Lasso, ElasticNet (`lr.coefs_`)
 - ❖ NN (DFS, HVS, PFI)
 - ❖ RFE (SVM, kNN, т.д.)



Permutation Feature Importance

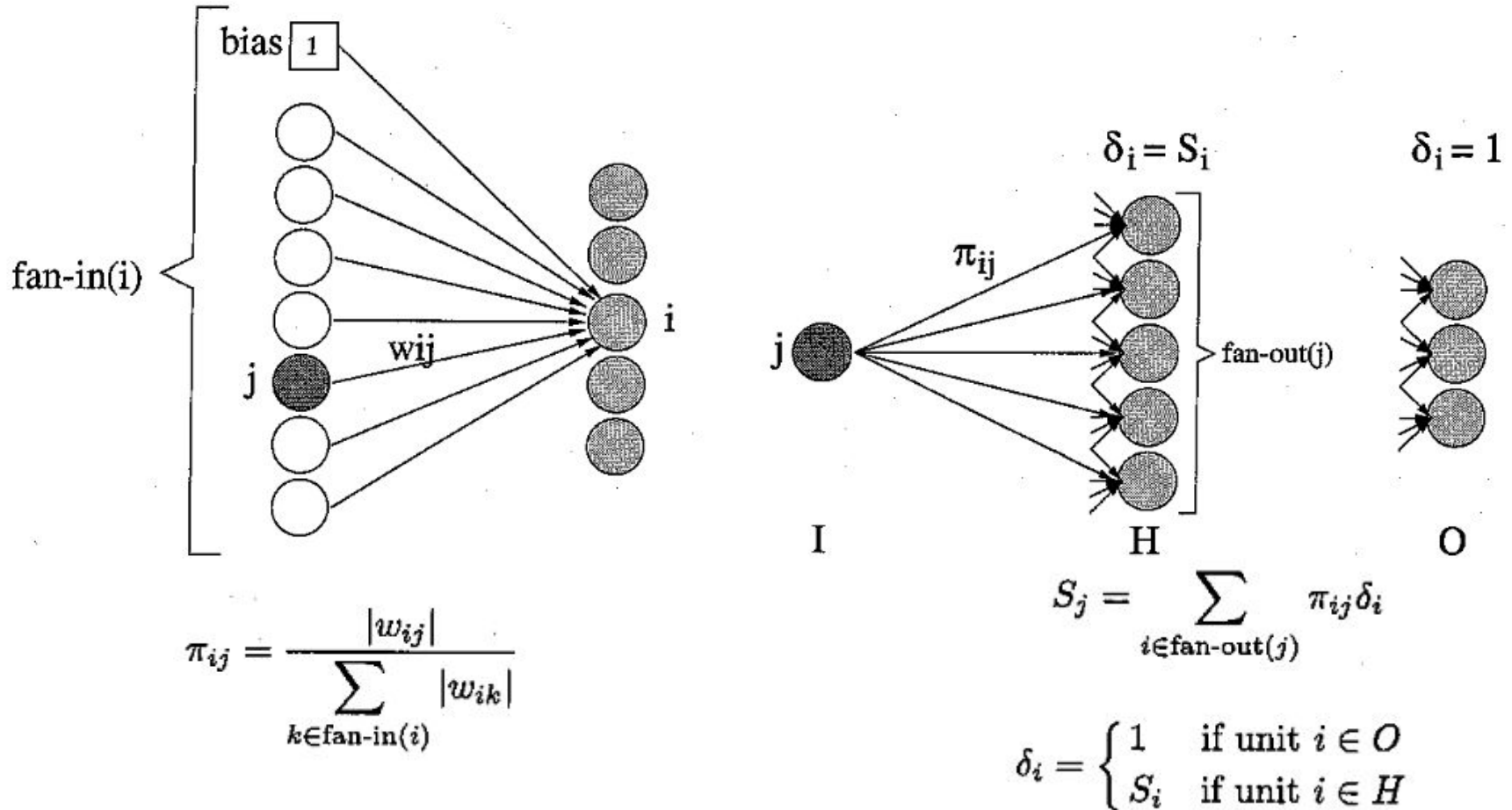
```
1. def shuffle(array):
2.     return np.array(sorted(array, key=lambda *args: np.random.random()))
3.
4. def pfi(model, data, target, metric, bootstrap=10):
5.     performance = metric(target, model.predict(data))
6.     feature_importances = []
7.     for feature in range(num_features):
8.         feature_score = []
9.         clone = np.copy(data)
10.        for step in range(0, bootstrap):
11.            clone[:,feature] = shuffle(clone[:,feature])
12.            pred = model.predict(clone)
13.            feature_score.append(metric(target, pred))
14.            feature_importances.append(performance-np.mean(np.array(r2_feature_score)))
15.
16.    return feature_importances
```

Deep Feature Selection

$$\begin{aligned} \min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = & l(\boldsymbol{\theta}) + \lambda_1 \left(\frac{1 - \lambda_2}{2} \|\mathbf{w}\|_2^2 + \lambda_2 \|\mathbf{w}\|_1 \right) \\ & + \alpha_1 \left(\frac{1 - \alpha_2}{2} \sum_{k=1}^{K+1} \|\mathbf{W}^{(k)}\|_F^2 + \alpha_2 \sum_{k=1}^{K+1} \|\mathbf{W}^{(k)}\|_1 \right) \end{aligned}$$

Li, Yifeng, Chih-Yu Chen, and Wyeth W. Wasserman. "Deep Feature Selection: Theory and Application to Identify Enhancers and Promoters." *Journal of Computational Biology* 23.5 (2016): 322-336.

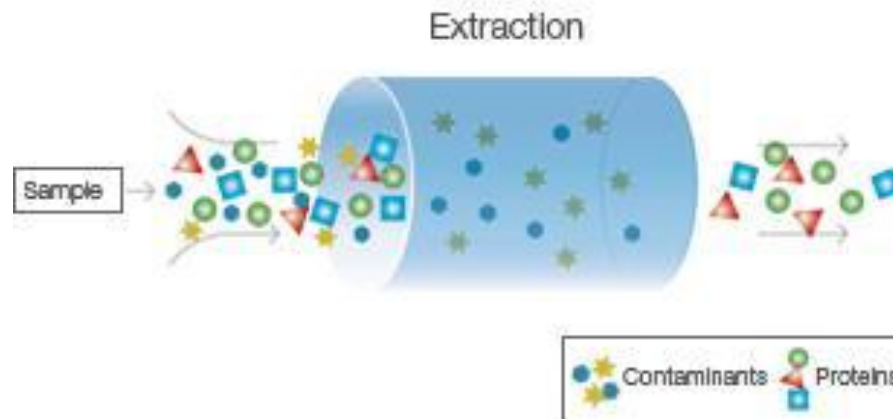
Heuristic Variable Selection



Yacoub, Meziane, and Y. Bennani. "HVS: A heuristic for variable selection in multilayer artificial neural network classifier." Intelligent Engineering Systems Through Artificial Neural Networks, St. Louis, Missouri. Vol. 7. 1997.

Экстракция признаков

- ❖ Экстракция через визуальный анализ (handcrafted признаки)
- ❖ Экстракция через model-based методы (NN, RandomForest, т.д.)
- ❖ Экстракция через автоэнкодеры (AE) (Stacked AE, Denoising AE, т.д.)
- ❖ Экстракция через методы сокращения размерности (PCA, kernel PCA, t-SNE)
- ❖ Экстракция через методы кластеризации (kNN, AffinityPropagation, DBSCAN, т.д.)



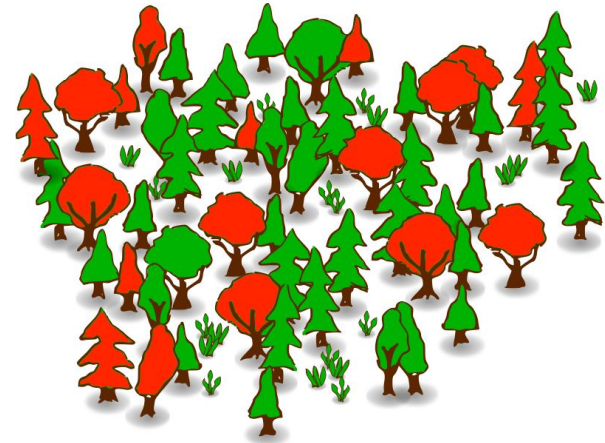
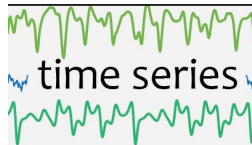
Инженерия признаков

- ❖ Простейшие handcrafted признаки: среднее, дисперсия и т.п. по примеру
- ❖ Исследование взаимодействия признаков между собой и признаков с таргетами
- ❖ Handcrafted признаки, основанные на знании области (формулы, т.д.)
- ❖ Введение зависимостей $x_1 * x_2$, x_1^2 , $\sin(x_1) * x_2$, $\log_{1p}(x_1)$, т.д.
- ❖ Handcrafted признаки, основанные на анализе временных рядов (сезонность, т.д.)
- ❖ Handcrafted признаки, основанные на правилах (правила переходов, т.д.)
- ❖ Handcrafted признаки, основанные на пространственной зависимости примеров (ближайшие соседи, т.д.)

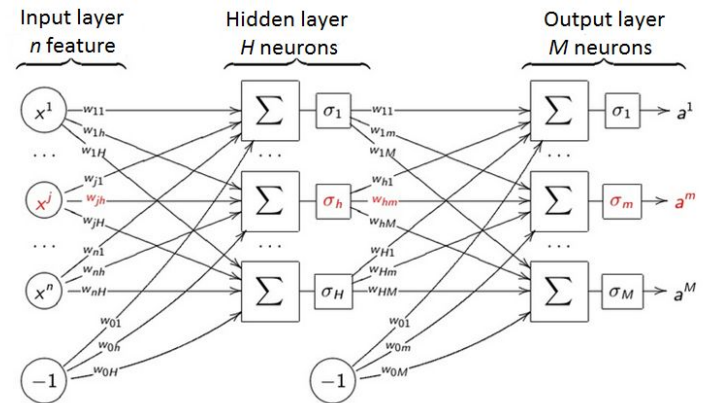
Построение моделей



Simple data

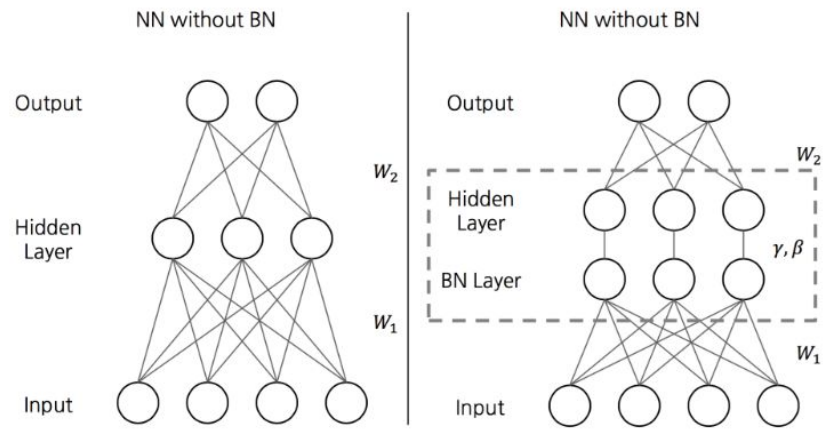
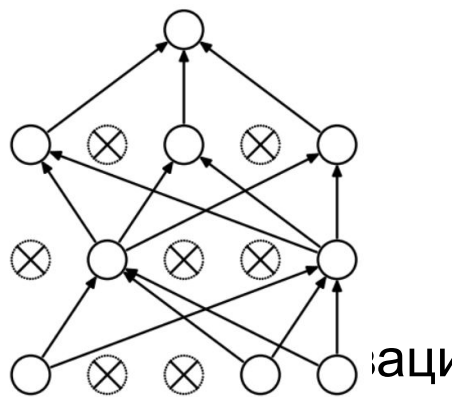
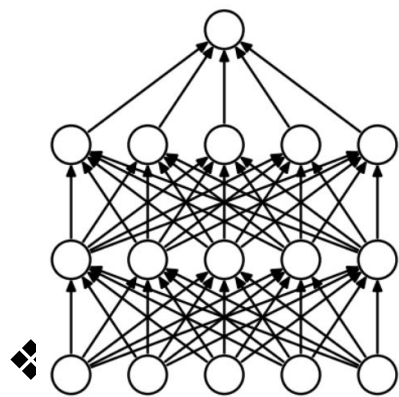


Complex data

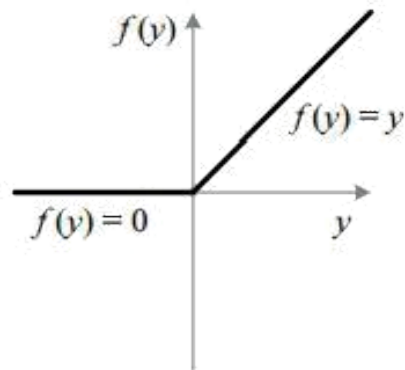


Обучение нейронных сетей

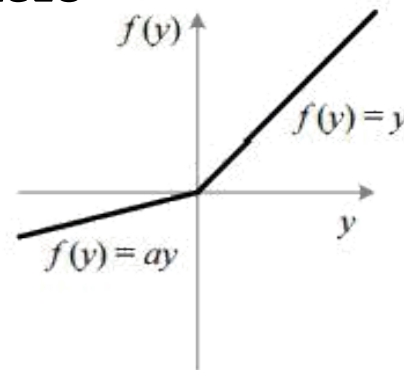
- ❖ Использовать методы регуляризации для сетей: Dropout, BatchNormalization, weight decay



ReLU

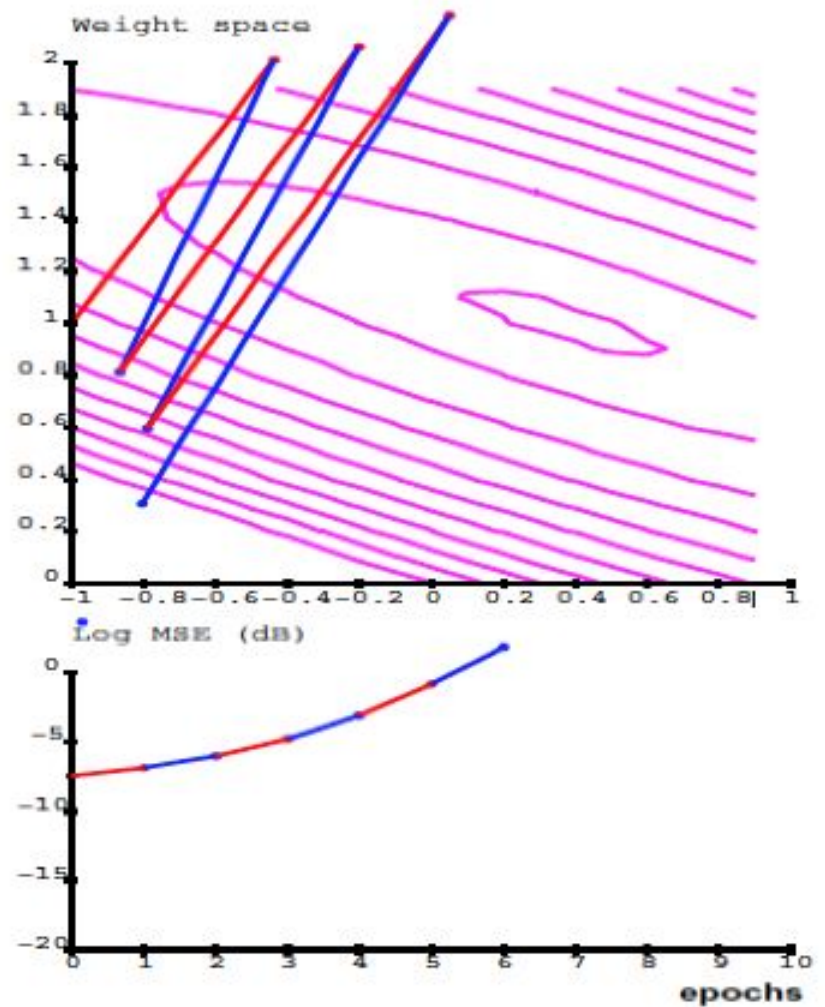
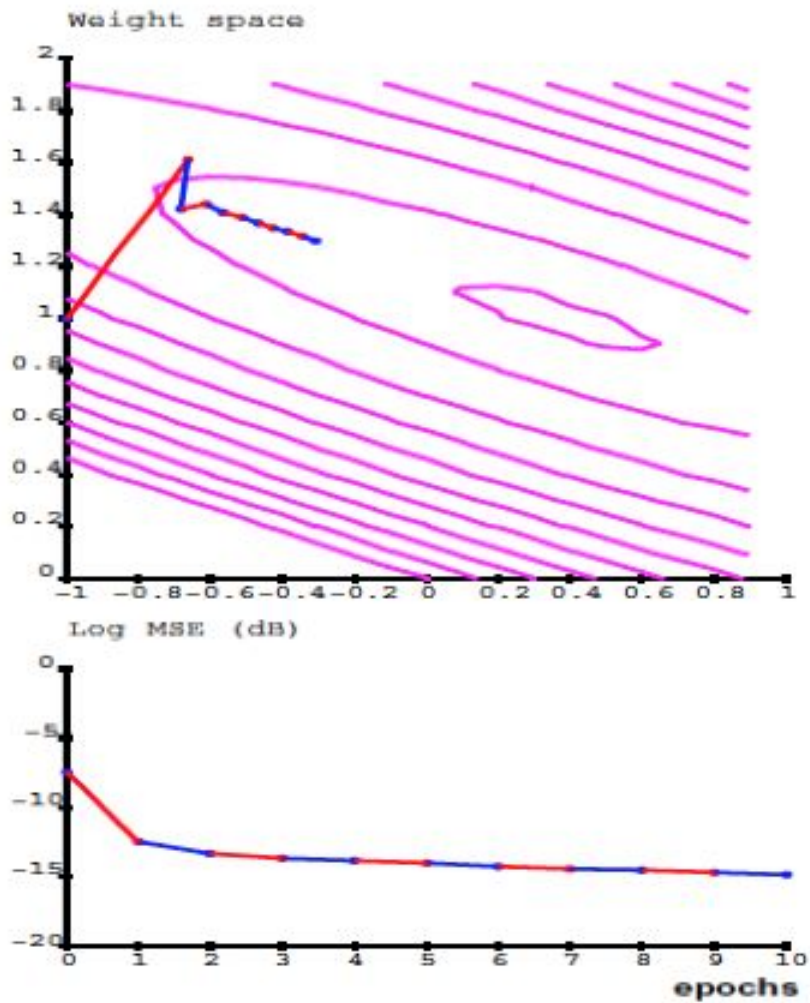


PReLU



Обучение нейронных сетей

Обучение нейронных сетей





TOP SECRET

Модели победители на Kaggle соревнованиях

- ❖ Использовать GBM из xgboost, random forest, regularized greedy forest

dmlc

XGBoost

- ❖ Использовать NN и Theano (Keras, Lasagne, Caffe) или Tensorflow

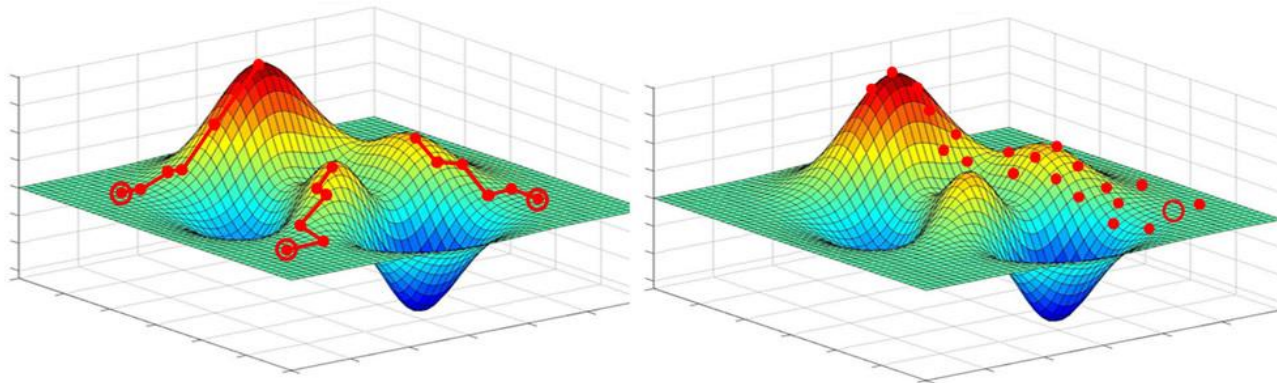
theano

Технические Tips & Tricks

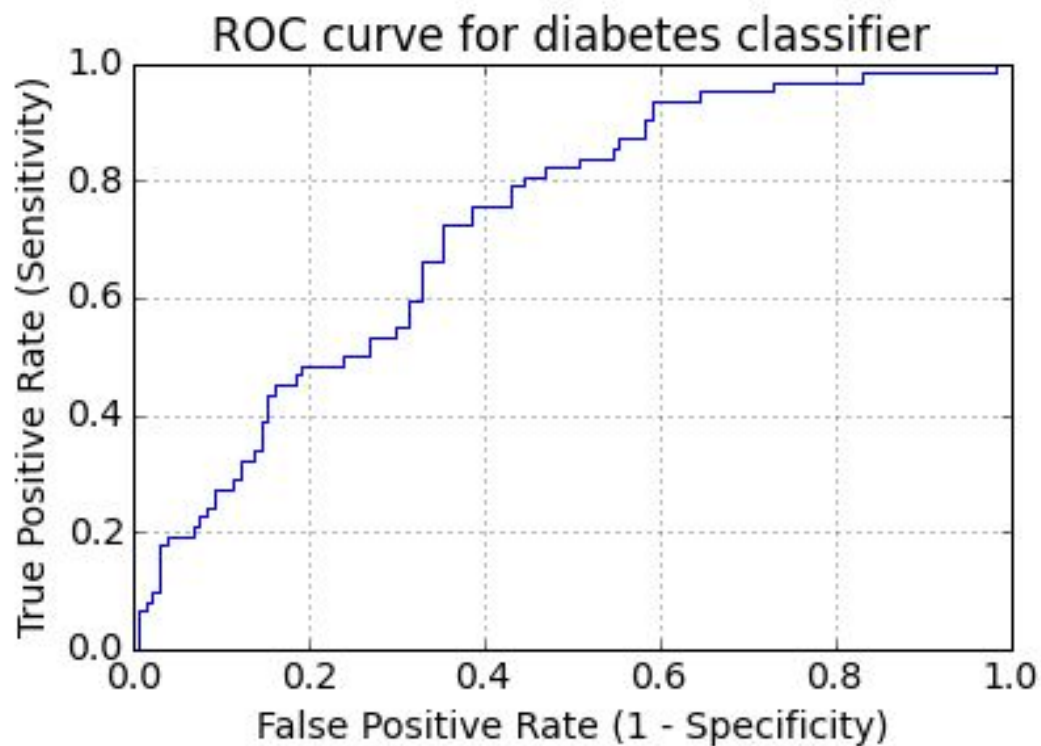
- ❖ Делать верную предобработку данных
- ❖ Правильно работать с нормализацией/выбросами/пропусками
- ❖ Проводить визуальный анализ данных, чтобы лучше понять данные, извлечь высокоуровневые признаки
- ❖ Делать сокращение размерности, выбор/экстракцию признаков model-free & model-based методами
- ❖ Строить разные модели на разных данных
- ❖ Выбрать правильные метрики
- ❖ Выбрать верную кроссвалидацию
- ❖ Правильно подбирать модель и ее гиперпараметры
- ❖ Подбирать оптимальный порог предсказаний
- ❖ Делать калибровку вероятностей моделей
- ❖ Правильно работать с несбалансированными данными
- ❖ Использовать аугментацию данных
- ❖ Строить ансамбли моделей, т.е. делать стекинг (Stacking)

Настройка гиперпараметров

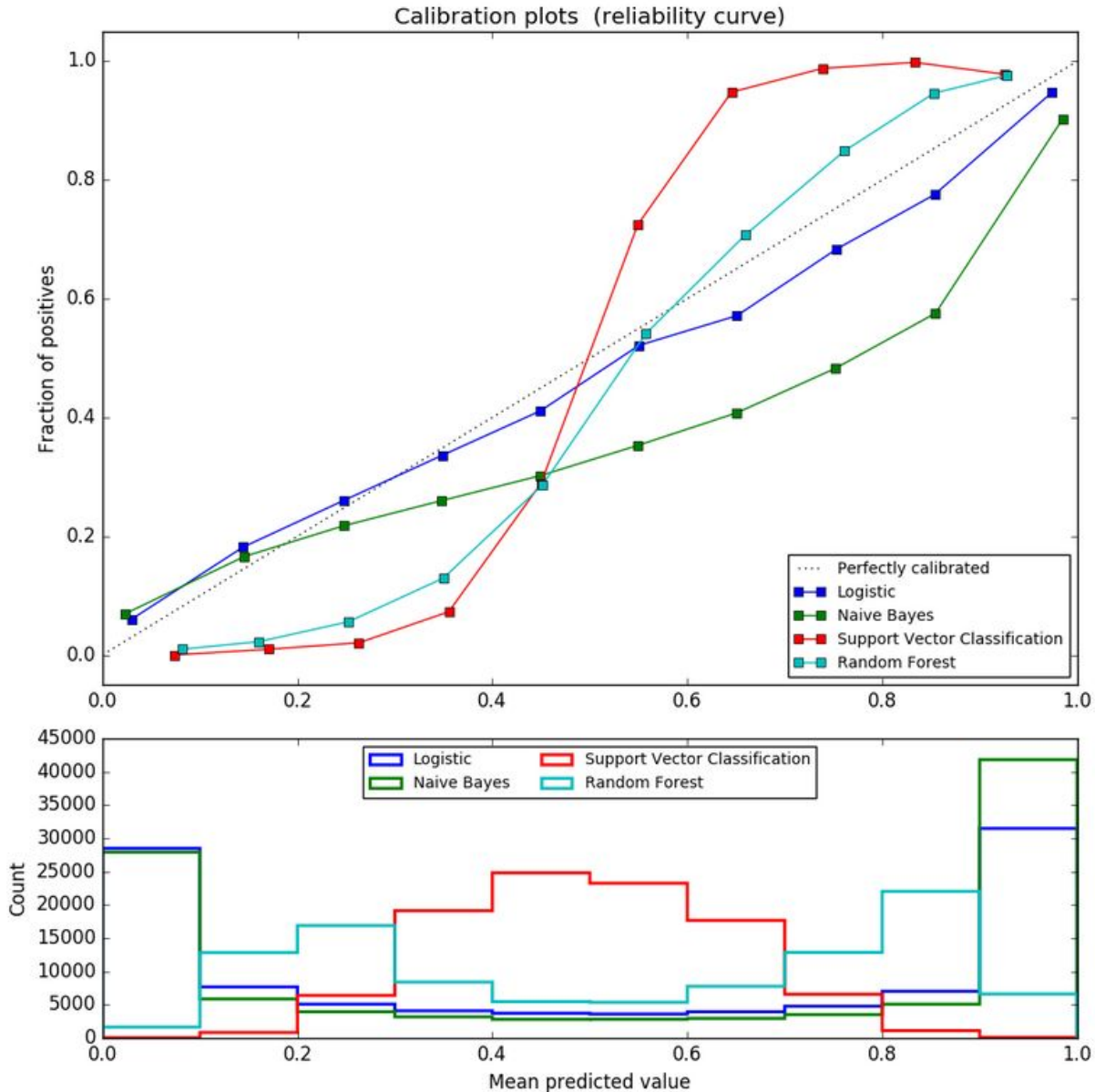
- ❖ Найти оптимальное подмножество данных на котором стоит обучаться и настраивать гиперпараметры моделей
- ❖ Использовать следующую схему GridSearch, RandomSearch:
 - ❖ На первом шаге используем попараметровый grid search для большого количества значений
 - ❖ На втором шаге для лучших значений используем обычный GridSearch, RandomSearch
- ❖ Использовать методы байесовской оптимизации:
 - ❖ [baeys_opt](#) для моделей sklearn
 - ❖ [hyperopt](#) для моделей нейронных сетей на Theano



Оптимальный порог



Калибровка вероятностей



`sklearn.calibration.CalibratedClassifierCV`

Несбалансированные данные

- ❖ Использовать методы балансировки данных: [imbalanced-learn](#)
- ❖ Undersampling (SVM, kNN, NN)
- ❖ Oversampling (SVM, kNN, NN)
- ❖ Использовать методы генерирующие данные в процессе своего обучения (NN)
- ❖ Использовать метрики для несбалансированных данных (F1-score, [Matthews correlation coefficient](#))



Спасибо за внимание!
Вопросы?

Евгений Путин
Университет ИТМО
putin.evgeny@gmail.com

25 мая 2017
Санкт-Петербург

