Baseline model

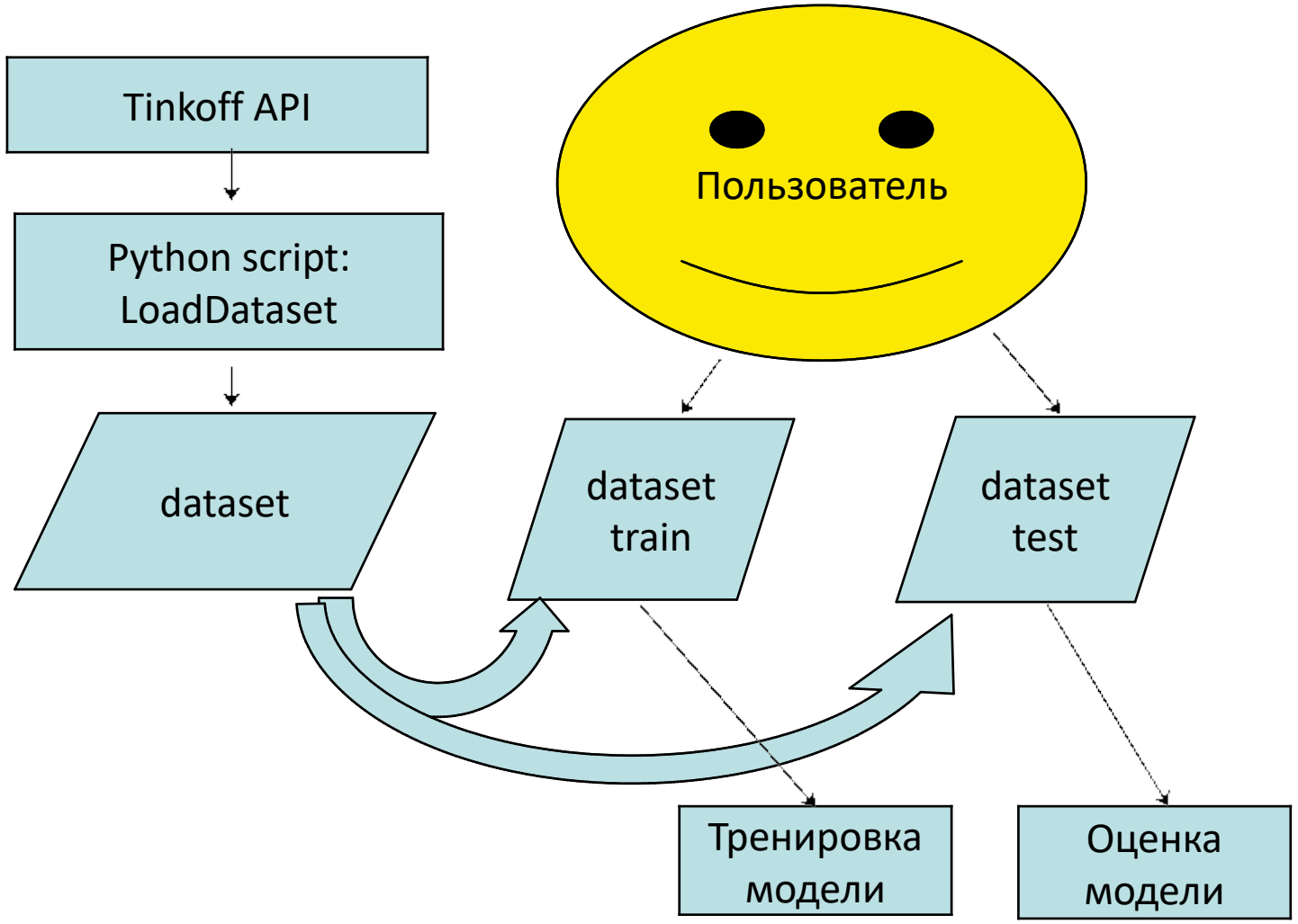
Действующая MVP реализация на R

Использованы технологии: Python, R (RStudo), Plumber, Docker.

1. Пользователь скачивает данные с описаниями акций (в данной реализации с помощью скрипта Python через Tinkoff API).
2. Пользователь разделяет данные на train/test, проставляет метки и обучает модель.
3. Обученная модель собирается в микросервис на базе R Plumber.
(Почему R? В качестве ответа:
- Многие специалисты владеют R. Навык R даст возможность лучше коммуницировать в команде.
- Прототипирование на R быстрее, чем на Python.)

Обучение модели

(диаграмма потока данных)

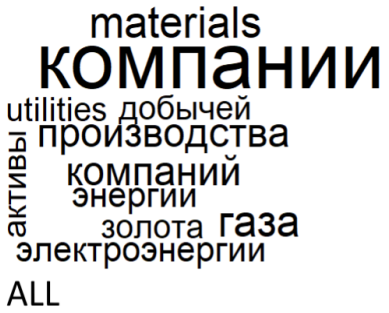


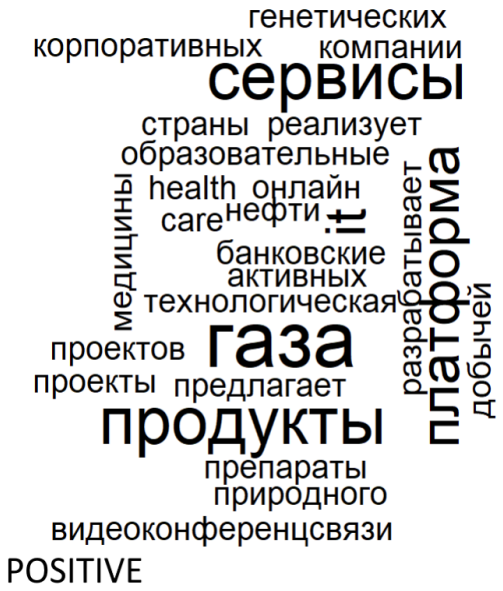
Архитектура модели

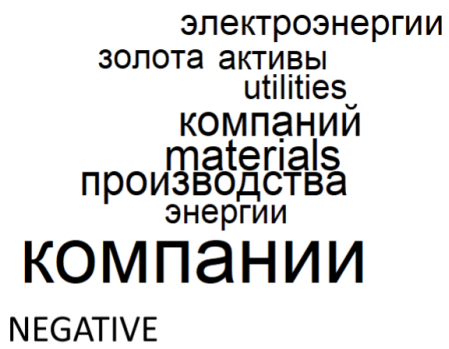
- Текст описаний предобрабатывается: stop words, пунктуация, цифры, в нижний регистр, word stem.
- К тренировочному набору может добавляться дополнительный искусственный набор записей со словами, которые редки, но выглядят важными для классификации с точки зрения пользователя (с word stem), например

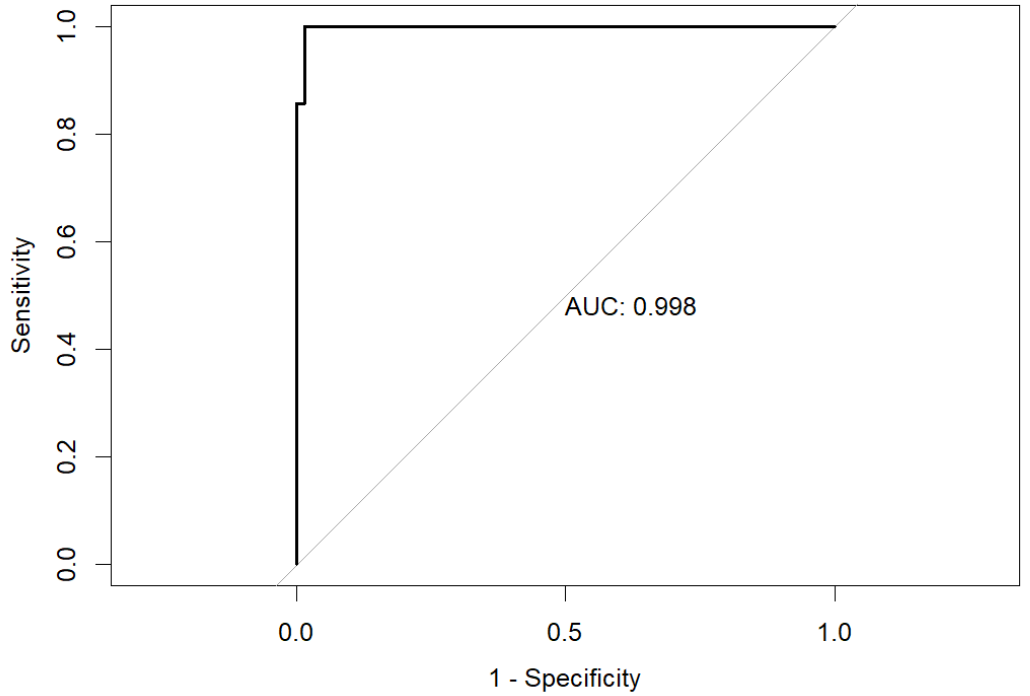
  label text
  1 gpt
  0 materials

- Строится document-term matrix по алгоритму Bag Of Words.
- Для анализа получаем Word Cloud.
- Для document-term matrix значения нормализуем единицей (значения частоты ограничиваем значением 1).









- Используем модель Random Forest (в базовой реализации использован размер леса: 200 деревьев).

Использование модели

- Модель реализована на R (проект в RStudio), для REST JSON API использован модуль Plumber, реализация собирается в докер контейнер. Пример локального адреса страницы Swagger: http://127.0.0.1:5762/__docs__/#/
- Для сборки докер-образа использован менеджер пакетов pak — это дало высокую скорость сборки и хорошую совместимость докер-образа.

```
FROM rocker/r-ver:4.3.3

# install os dependencies
RUN apt-get update -qq
RUN apt-get install -y --no-install-recommends \
  git-core \
  libssl-dev \
  libcurl4-gnutls-dev \
  curl \
  libsodium-dev \
  libxml2-dev \
  && rm -rf /var/lib/apt/lists/*

# install pak alternatives to install.packages
RUN Rscript -e "install.packages('pak', repos = sprintf('https://r-lib.github.io/p/pak/stable'))"

# install latest plumber from github main branch
RUN Rscript -e "pak::pkg_install('rstudio/plumber@main')"

# install other R packages
RUN Rscript -e "pak::pkg_install(c('caret', 'randomForest', 'quanteda', 'irr', 'stringr', 'dplyr', 'SnowballC', 'swagger', 'rapidoc'))"


RUN mkdir /app
COPY / /app
WORKDIR /app
EXPOSE 5762
ENTRYPOINT ["Rscript", "runPlumber.R"]
```

POST `/api/inference` inference

Parameters | Try it out

No parameters

Request body | application/json

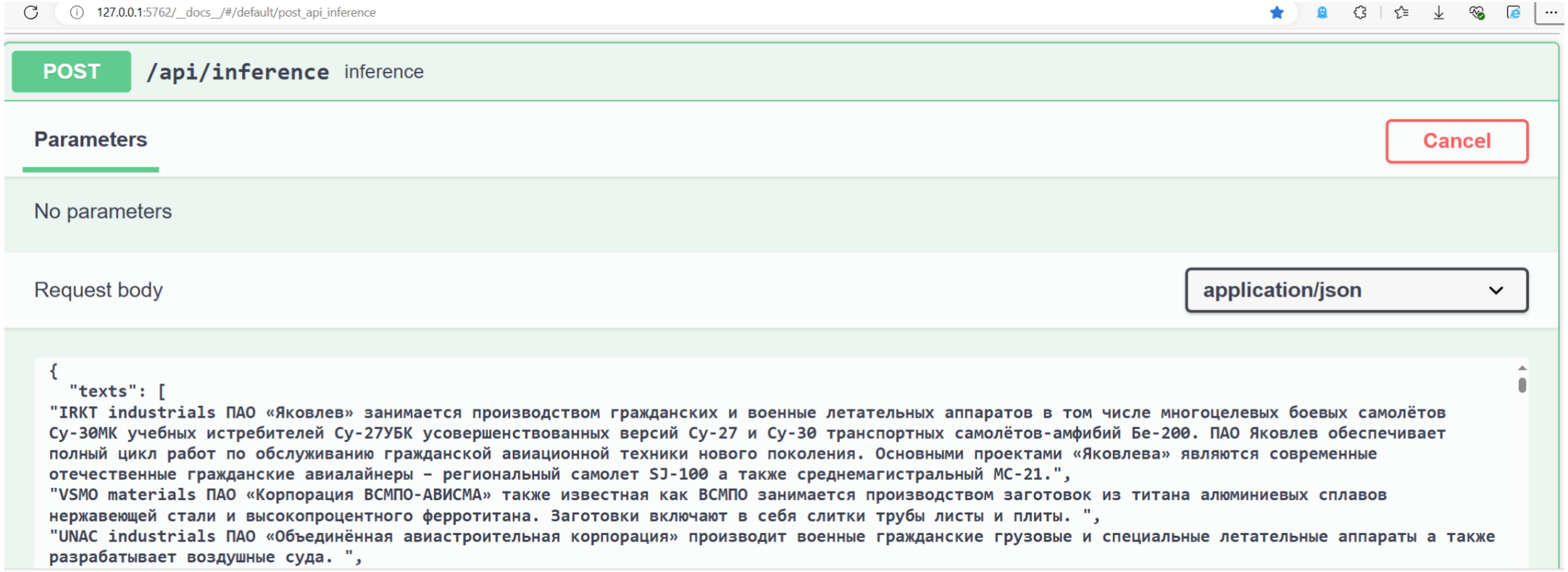
Example Value | Schema

```
{
  "texts": [
    "TICKET sector description"
  ]
}
```

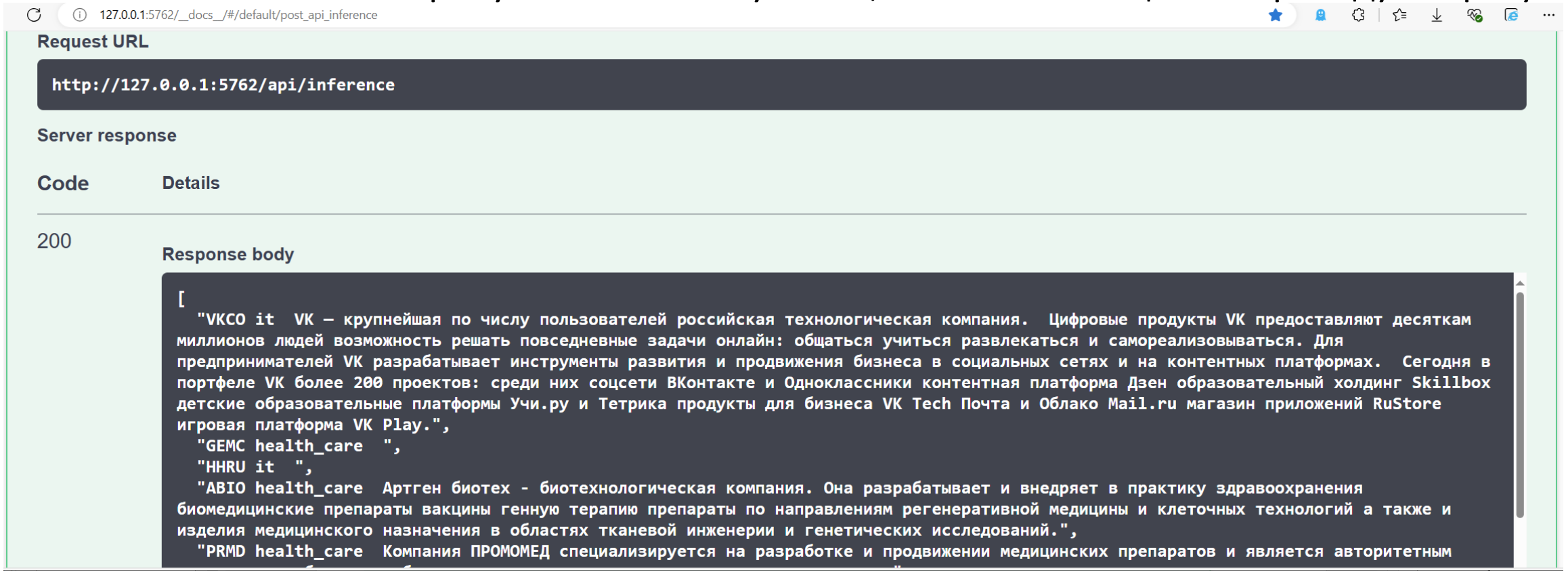
Responses

Code	Description	Links
200	OK	*No links*

На вход подается список текстов с описаниями компаний (в базовой реализации данные получается из Tinkoff API), в текст описания компании добавляются названия категорий от брокера и тикеты акций (при тренировке тикеты были удалены, как уникальные идентификаторы, но при inference они не мешают работе модели, а нужны для идентификации результатов). Предобработка текстов выполняется внутри микросервиса модели. При этом тексты приводятся к такой же document-term-matrix, как была при обучении.



На выходе возвращается отфильтрованный входной список: оставляются только компании, которым модель присвоила позитивный класс (1). По тикету пользователь идентифицирует акции. Приоритет на recall, а не precision: модель присваивает класс 1 пустым описаниям и текстам только из неизвестных слов, потому что для инвестора может быть важнее не пропустить ничего нужного, а “лишние” акции не приведут к краху.



Возможности улучшения модели

- Поменять алгоритм Bag Of Words на более актуальный TF-IDF.
- Применить для предообработки данных NER модель, которая удалит из тренировочного dataset все названия компаний (названия это идентификаторы, на них модель переобучается). Сейчас удаление названий компаний основано на редактируемом пользователем stop list.
- Добавить автоматический мониторинг дрейфа данных, чтобы вовремя заново тренировать модель при снижении качества результатов, если будут значительные изменения в данных о компаниях.
- В качестве эксперимента можно добавить features на основе графовых алгоритмов (одним компаниям могут принадлежать другие — получаем направленный граф). Такое улучшение подойдёт брокерам (у которых есть нужные данные) или для отдельных инвесторов, которые готовы сами обогащать данные.