Хранение и предварительная обработка больших наборов данных с помощью TensorFlow

к.т.н., доцент кафедры «Информационные системы» УлГТУ

Гуськов Глеб Юрьевич

ИНСТИТУТ РЕГИОНАЛЬНОГО РАЗВИТИЯ

НОЦ ИНЖЕНЕРИЯ БУДУЩЕГО

Ростех

РОСКОСМОС

РЖД

План занятия

1. Что такое big data?
2. Примеры информационных систем построенных на Big Data
3. Архитектура информационных систем построенных на Big Data
4. Алгоритмы обработки Big Data

Большие данные вокруг нас

1. Что такое Большие данные (Big Data)?

Большие данные вокруг нас

- Когда данные информационной системы становятся большими?
- Чем отличается обработка больших данных?
- Почему Big Data выделяют в отдельный пункт при проработке проекта?
- Как Big Data влияет на архитектуру информационной системы?
- Для Big Data требуются специфические алгоритмы обработки данных?

ВАЖНО!

Big Data (Большие данные) – это данные характеризуемые несколькими особенностей:

- Объём (Volume);
- Поступление и обработка новых данных (Velocity);
- Разнородность данных (Variety).

А лучше даже сформулировать большие данные именно как произведение всех перечисленных особенностей:

Большие данные = Объём × Обработка данных × Разнородность данных

Big data : объём данных (Volume)

Объём данных в абсолютных значениях сильно зависит от времени и применяемых технологий хранения.

Гибкий магнитный диск или «дискета»

Объём : 1.44мб
Период: 1970 - 1990

Компакт диск (CD)

Объём : 700мб-50гб
Период: 1980 - 2021

Жесткий диск (HDD->SSD(Твердотельный))

Объём : 700мб-50гб
Период: 1980 - 2021

Сервер/ЦОД

Объём : 100+пб.
Но так ли важен объём?
Период: 1995 - 2021

Обратите внимание!
Big data : объём данных (Volume)

Но так ли важен объём?

С ростом объёма и доступности носителей хранения цифровой информации, важность объёма одного накопителя падает.

- Данные можно хранить в облаках и загружать необходимые в нужный момент;
- Данные делятся по актуальности резонно хранить в быстром доступе самые необходимые;
- Многие данные на самом деле не востребованы и избыточны;
- Данные можно масштабировать/защищать и хранить централизовано;

С переходом от носителя к ЦОД для пользователя изменился способ взаимодействия с данными:

- Удаленный доступ к любому интересующему документу
- Отсутствие риском «потерять» данные
- Передача документов между пользователями/совместная работа с документами
- Масштабируемость доступного объёма информации

Big data : Поступление и обработка новых данных (Velocity)

Переход от носителей к центрам обработки данных позволяет рассматривать данные не как набор документов, а скорее как совокупность потоков данных.

Часто под Velocity понимают обработку данных. Тем не менее процесс поступления и обработки новых данных более комплексный.
Данные могут поступать:

- с разной скоростью;
- нелинейно (например раз в сутки поступает большой документ с данными с датчиков только в случае существенных изменений на метеостанции или производственной линии);
- обрабатываться с разной скоростью (данные могут обрабатываться не линейно по времени, что может влиять на состояние системы).

Big data : Разнородность данных (Variety)

Разнородность данных (Variety) подразумевает формат и комплексность данных. Данные могут быть представлены в формате текста, переписки, таблиц, аудио-документов, записанной речи, видео файлов и т.д.





Большие данные: Ценность и достоверность (Value and Veracity)

Как правило, под **достоверностью** принято понимать правдивость набора данных. Во многих случаях достоверность наборов данных можно проследить с точностью до до источника. Данные часто собираются из разных источников, что затрудняет отслеживание их достоверности. С другой стороны большее число источников несколько диверсифицирует проблему достоверности данных.

Значение данных – это некоторая оценочная величина, которую можно выразить в виде потенциальной социальная или экономическая ценности, которую можно извлечь проанализировав данные.

Примером экономической ценности может служить выборка клиентов среди пользователей, которые с высокой вероятностью приобретут товар или услугу. Связи могут быть более сложными.

Примером социальной ценности может быть расстановка приоритетов по реализации благ для жителей города. Данные приоритеты могут быть формированы не с помощью социальных опросов, а при помощи анализа данных.

? Что такое Big Data? Контрольные вопросы:

1. Когда данные информационной системы становятся большими?

2. Чем отличается обработка больших данных?

3. Почему Big Data выделяют в отдельный пункт при проработке проекта?

4. Как Big Data влияет на архитектуру информационной системы?

5. Для Big Data требуются специфические алгоритмы обработки данных?

6. Что понимается под “3V” Объём (Volume),Поступление и обработка новых данных (Velocity), Разнородность данных (Variety). И как именно данные особенности образуют Большие данные?

2. Примеры систем основанных на больших данных

Какие системы основаны на Big Data?

- Можно ли считать любую автоматизированную информационную систему, системой работающей с большими данными?
- Какие знания можно извлечь из систем генерирующих большие данные?
- Большие данные в продвижении, веб-аналитике и в управлении организацией.

Автоматизированные информационные системы, которые основаны на больших данных

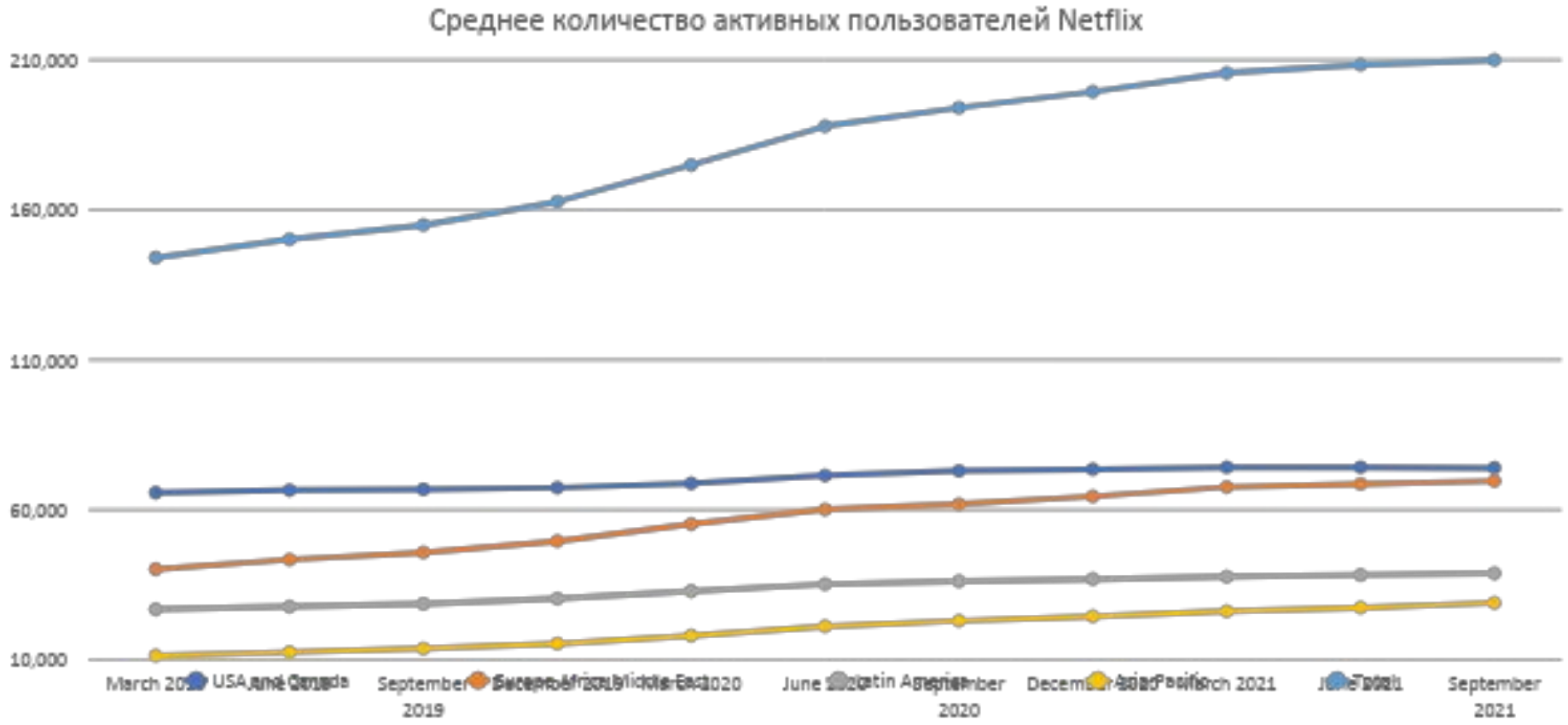
Информационная система	Технологии больших данных?
Система отслеживания банковских транзакций по банку	да
Данные о клиентах/покупках/товарах крупной торговой сети	да
Данные секвенирования по пациентам	да
Сервис цифровой дистрибуции контента (Netflix, Litres)	да
История действий пользователей провайдера	да
Система отслеживания данных GPS автомобилей (Система ПЛАТОН)	да
Данные с датчиков метеорологической службы по РФ	да
Интернет-магазин нишевых товаров (товары для художников, локальный магазин выпечки)	нет
Данные из 1С по региональному предприятию	нет
Мобильное приложение “Зимний комфорт” Toyota motor	нет
Мобильное приложение “Facebook”	да

Big data в цифровом контенте: Netflix

Компания Netflix использует большие данные для прогнозирования потребительского спроса при помощи создания предиктивные моделей для новых продуктов и услуг, классифицируя ключевые атрибуты предыдущих или существующих продуктов и моделируя взаимосвязь между этими атрибутами и коммерческим успехом предложений, кроме того, используются данные и статистика, получаемые из фокус-групп, социальных сетей, а также по результатам рыночных тестов и пробных продаж, после чего выпускают новые продукты.

- **Формирование персонифицированных рекомендаций для пользователя на основе его предпочтений и принадлежности к пользовательским группам**
- **Формирование линейки внутренних продуктов в соответствии с предпочтениями пользователей**
- **Поиск скрытых зависимостей в данных между атрибутами продуктов, пользователей, категорий и т.д.**

Big data в цифровом контенте: Netflix

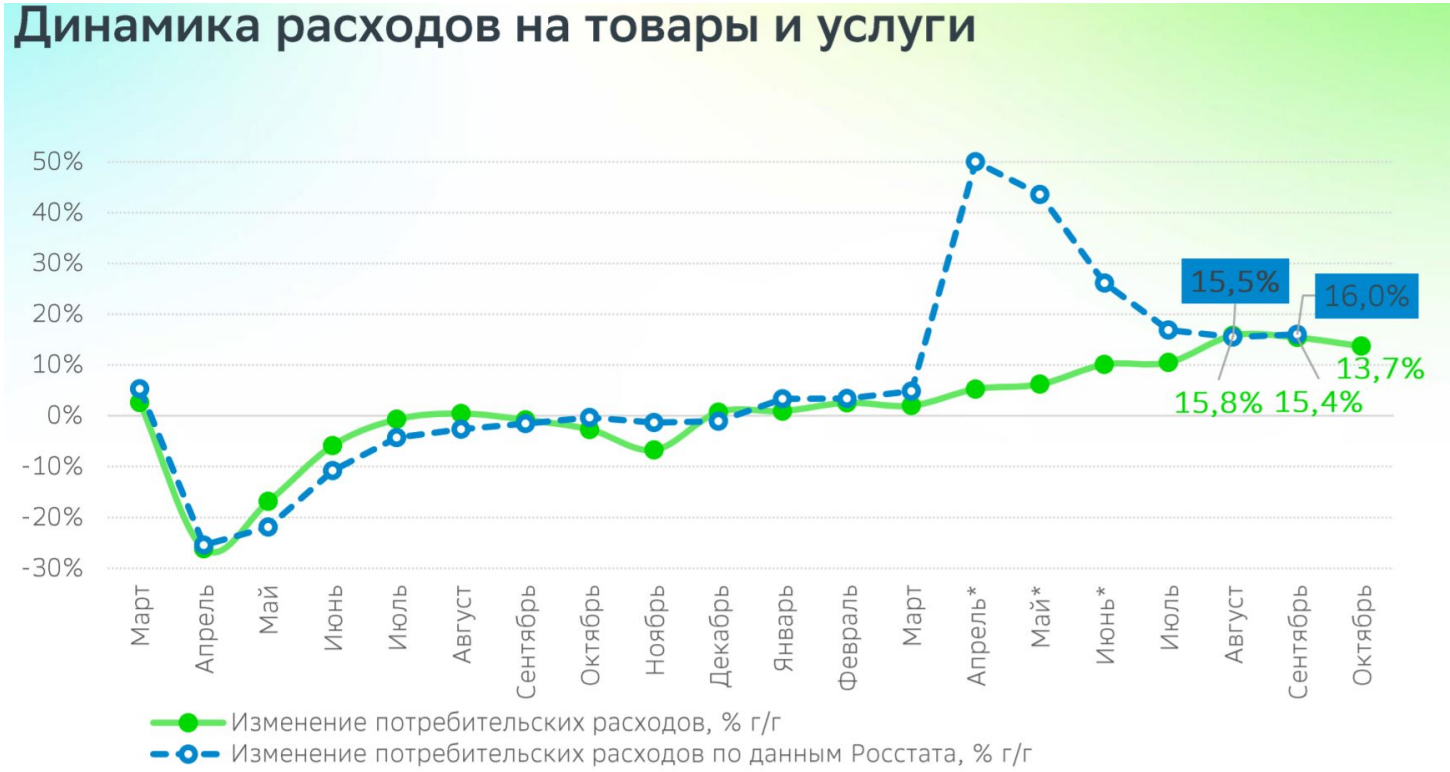


Big data в банковском секторе: Сбербанк

"Количество платежей по банковским картам уже превышает 1 миллиард операций в месяц и продолжает расти. Также мы отмечаем, что количество транзакций, совершаемых в интернете, растет существенно быстрее, чем в классическом offline-эквайринге", — сообщили в Сбербанке.
16.11.2018

- **Прогнозирование трат пользователей в следующем месяце**
- **Формирование ставок по кредитам и вкладам**
- **Создание автоматизированных моделей машинного обучения для автоматизации решения задач, например выдача кредитов клиентам по их аттрибутам**

Big data в банковском секторе: Сбербанк

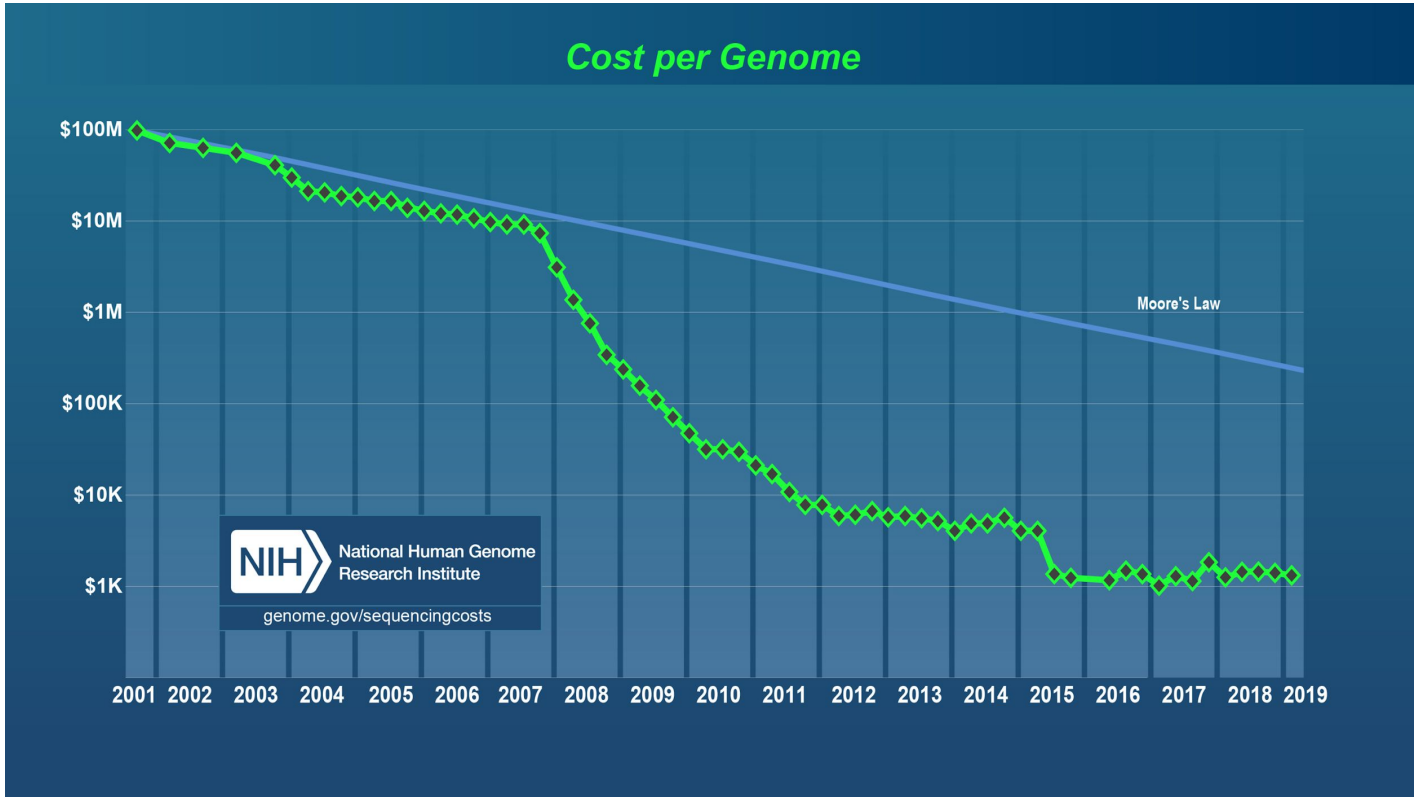


Источник: рассчитано по данным Сбера, Росстата.

В сентябре 2021 г. завершена синхронизация методологии оценок СберИндекса с Росстатом.

А на сколько просто собрать подобного рода статистику по данным с ~ 12 миллиардов размеченных транзакций?

Данные о геноме человека



Стоимость секвенирования человеческого генома по логарифмической шкале. Резкий скачок графика стоимости вниз соответствует появлению третьего поколения методов секвенирования.

Геном одного человека может быть сохранен в FASTA формате может быть сохранен в виде файла объёмом около 3,5 Гб. 3,5Гб – на первый взгляд совсем не много.

Данные о геноме человека

Исследования генома совершило революцию в медицинских услугах. Обработка данных генома становится крайне необходима в рамках фундаментальных исследований о природе заболеваний.

Данные о геноме используются в сочетании с дополнительной информацией:

- Данные о родственниках и наследственности пациентов;
- Общие анализы (например анализ крови);
- Изображения с аппаратов узи;
- Видео-записи;
- Фотографии травм/образований на коже;
- Изображения тканей созданные при помощи современных микроскопов;
- Изменения/мутации в геноме;
- Данные о геномах других живых существ, не только человека из которых формируется филогенетическое древо.

Если мы будем учитывать, что данные должны храниться по всем пациентам, имеющим отношение к поликлинике, то мы имеем дело с разнородными данными огромного размера, скорость генерации которых постоянно растёт.

Более подробно можно прочитать в статье – «Основные задачи больших данных в генетической информации». [6]

Другие применения больших данных

Для каких ИС большие данные имеют смысл?

Технологии больших данных дают максимальную эффективность в случае, если имеется множество источников данных, данные иногда меняют формат и тип, данные распределены и должны быть доступны для широкого круга пользователей. Это такие системы как : банки, онлайн библиотеки, операторы связи, социальные сети.

Имеет ли смысл использовать технологии больших данных, если информационная система достаточно локализована и проста?

В целом, если объёмы и сложность данных не так велика, то можно решить большую часть анализа данных стандартными средствами библиотек в которых реализованы пакеты статистики и моделей прогнозирования.

Только в случае, если потенциально ожидается рост объёмов данных и пользователей.

Основные задачи, решаемые при помощи больших данных

Технологические

- Распределенное хранение данных (данных которые не могут храниться на одном узле физически)
- Обеспечение горизонтальной масштабируемости решения
- Отказоустойчивость и сохранение целостности данных
- Распределение вычислительной нагрузки по узлам на уровне технологии

Пользовательские (Бизнес-ориентированные)

- Возможность обрабатывать огромные объёмы данных за конечное время в многопоточном режиме при помощи MapReduce
  - Рассчитывать характеристики для огромной выборки
  - Упорядочивать выборку
- Проводить корреляционный анализ скрытых зависимостей
- Проведение и выделение кластеров среди объектов (пользователей/товаров/ услуг и т. д.)

? Примеры систем основанных на больших данных Контрольные вопросы:

1.Приведите основные отличительные особенности систем в которых предпочтительно использовать технологии больших данных?

2. Приведите 5 своих примеров систем в которых используется технология больших данных.

3. Приведите 5 своих примеров систем в которых НЕ используется технология больших данных и в её использовании нет смысла.

4. Подумайте для каких систем с которыми вам довелось поработать технология разработки больших данных актуальна.

3. Архитектура информационных систем обрабатывающих большие данные

Архитектура информационных систем построенных на Big Data

- Что такое распределенная информационная система?
- Чем отличаются распределенные информационные системы от информационных систем построенных на классической схеме клиент-сервер?
- Что такое Apache Hadoop и HDFS?

Данные в информационных системах

К сожалению часто данные в информационных системах имеют низкий уровень структурированности/упорядоченности/обработки. Данные зачастую хранятся на одном сервере, а отказоустойчивость обеспечивается наличием бекапов (сохранением предыдущих состояний данных информационной системы).

Производить анализ данных в таком состоянии можно, но для использования современных аналитических инструментов стоит изменить подход к хранению данных.
Если речь идёт о поточной обработке больших данных с поддержкой масштабирования и отказоустойчивости решения стоит использовать специальное программное обеспечение.

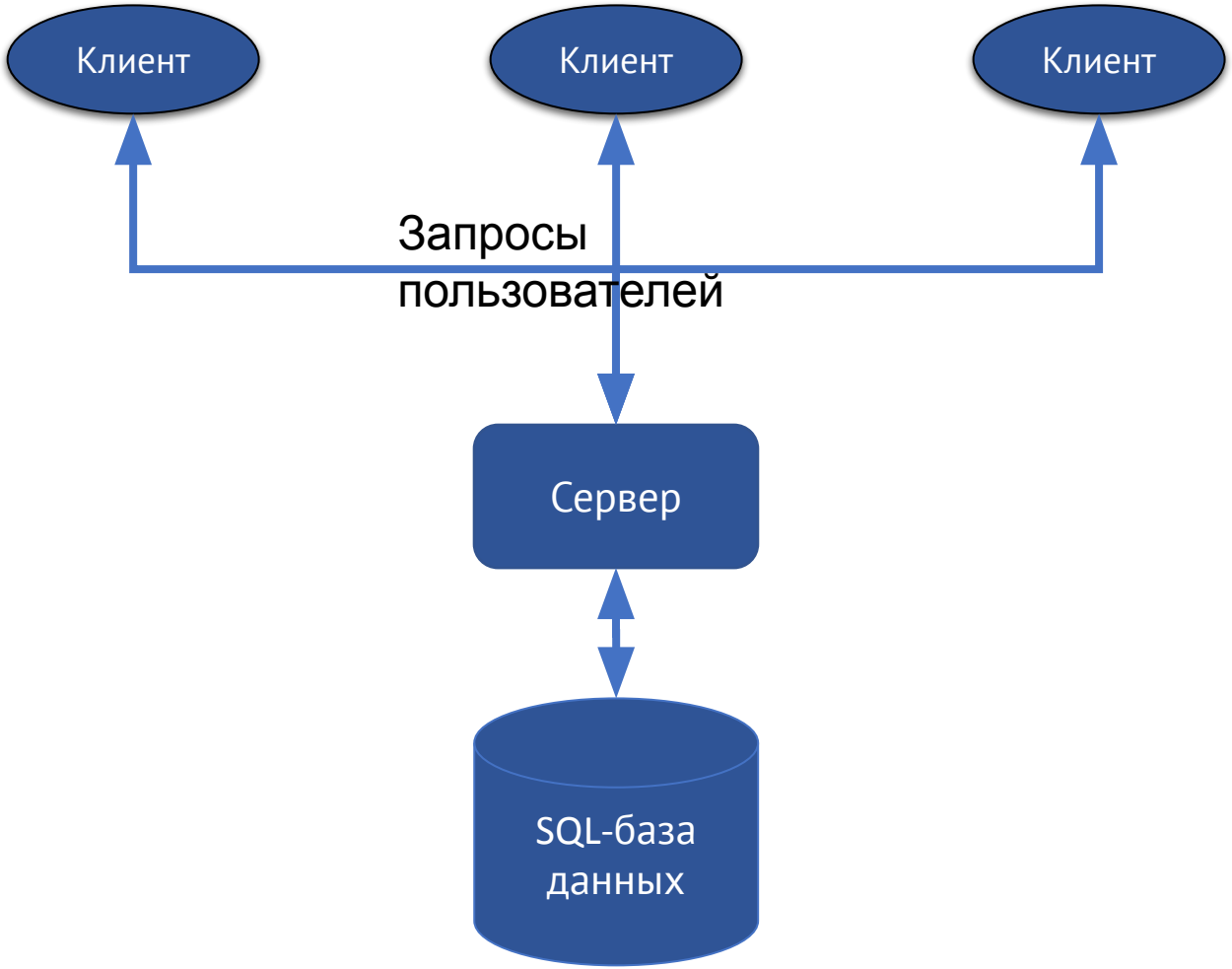
Apache Hadoop

Apache Hadoop – это развивающаяся платформа с открытым исходным кодом (open-source software) обеспечивающий надёжность, масштабируемость и распределённость обработки больших наборов данных на кластерах компьютеров.
Данная платформа позволяет масштабировать решение от отдельного сервера до тысяч машин, каждая из которых предлагает локальную обработку и хранение данных.

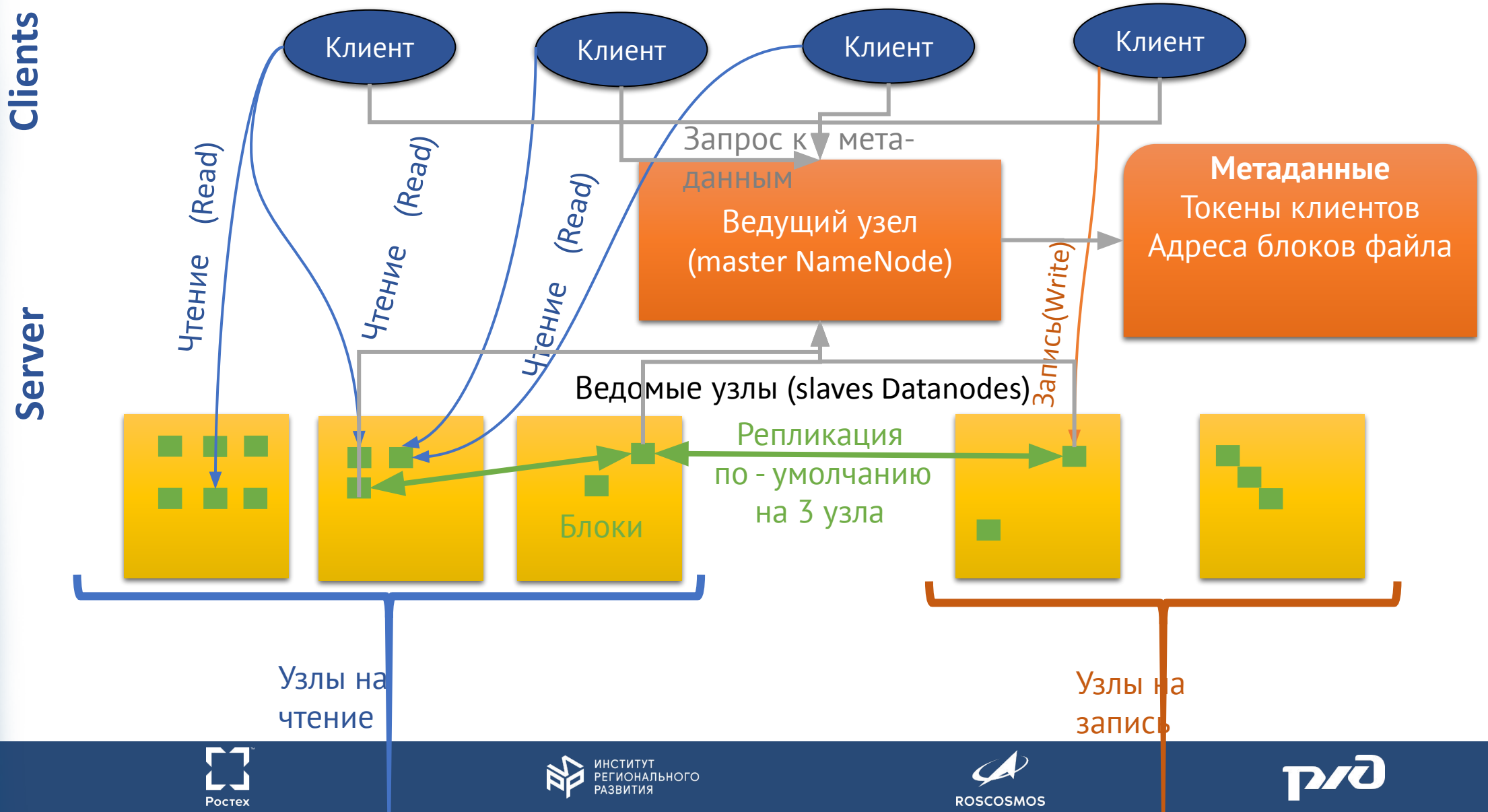
Apache Hadoop предоставляет распределенную файловую систему (Hadoop Distributed File System HDFS) имеющую следующие преимущества:

- предназначена для работы на стандартном оборудовании (не предполагает специализированных аппаратных средств);
- обладает высокой отказоустойчивостью ;
- обеспечивает высокопроизводительный доступ к данным приложения и подходит для приложений с большими наборами данных;
- позволяет масштабировать количество узлов.

Классическая Client-Server архитектура



Архитектура HDFS



Архитектура HDFS

Описание архитектуры HDFS

Apache Hadoop основана на архитектуре ведущий-ведомый (master-slave). Каждый кластер HDFS содержит один NameNode (ведущий узел/master), который осуществляет управление документами на DataNode(ведомых узлах/slave).

Программное обеспечение разработано на языке Java и может быть исполнено в любой операционной системе, поддерживающей виртуальную машину Java.

Документы в HDFS разбиваются на несколько блоков (bloks) равного размера. На DataNode хранятся именно блоки, а не документы целиком. Это позволяет работать с документами произвольного размера и прогнозировать время обработки одного блока.

Все операции с документами, которые производят клиенты инициализирует NameNode. NameNode проверяет права клиента, генерирует токен доступа на чтение или запись, синхронизирует адреса для блоков конкретного документа.

Блоки документов передаются с DataNode клиентам напрямую, не попадая на NameNode. Это важно, т.к. количество NameNode не является узким местом(бутылочным горлышком «bottleneck») системы.

Архитектура HDFS

Отказоустойчивость архитектуры HDFS

В случае падения одного из DataNode, блок документа загружается из другого DataNode в которых храниться данный блок.
Блоки реплицируются между DataNode без участия NameNode или вносящего изменения клиента. Блок реплицируется на 3 узла DataNode по умолчанию, значение может быть изменено в настройках.
Падение NameNode так же возможно и в реально работающих системах узлы NameNode так же дублируются, кроме того в случае отказа одного из NameNode будет поднят другой.
В случае одновременного отказа некоторого количества NameNode есть некоторый риск потерять часть не сохраненных данных. Но вероятность подобного события довольно мала.

Масштабируемость архитектуры HDFS

Масштабируемость архитектуры HDFS представлена возможностью редактировать число DataNode и NameNode динамически: в случае отказа или требований к наращиванию объёма данных.
Кроме того число клиентов так же может изменяться. Под клиентом понимается не конкретный пользователь, а программное обеспечение позволяющее совершать клиентские запросы.

Apache Hadoop решения и аналоги

Apache Spark - это единый аналитический движок (программный продукт) обработки больших (относительно) данных. Apache spark предоставляет высокоуровневые API-интерфейсы для языков Java, Scala, Python и R, также поддерживает богатый набор инструментов более высокого уровня, включая Spark SQL, MLlib, GraphX.
Набор инструментов Apache Spark схож с Apache Hadoop, но он не поддерживает распределенность. Apache Spark использует только один узел, на котором он развернут. В качестве некоторого бонуса он позволяет работать с данными в оперативной памяти.

Разные поставщики. Так как сам по себе Hadoop распространяется по open-source лицензии есть довольно много реализаций данного программного обеспечения от разных компаний: Cloudera, MapR, HortonWorks, и т.д. Все они предоставляют поддержку в виде исправления ошибок и добавления новых возможностей.

Platform as a service Сейчас достаточно сильно распространён подход предоставления не только программного обеспечения, но и облака в целом(платформы). На данном рынке представлено много решений от Amazon, Microsoft, IBM, Google, Mail.ru, Yandex.
Для работы в рамках законодательства РФ о хранении персональных данных (ФЗ № 242-ФЗ от 21 июля 2014 г.), интернет трафика и шифровании (ФЗ № 374-ФЗ от 6 июля 2016 г. и № 375-ФЗ от 6 июля 2016 г.) подходят только Mail.ru и Yandex, т.к. из ЦОДы находятся на территории РФ.

? Архитектура информационных систем построенных на Big Data
Контрольные вопросы:

1. Что такое принцип ведущий-ведомый(master-slave)?

2.Что такое HDFS?

3. Что используется в качестве единицы информации в узлах HDFS?

4. Какой из элементов HDFS создаёт токены для чтения и записи?

5. Как задействована репликация в процессе передачи информации между узлами?

6. На каких операционных системах есть возможность запустить HDFS и почему?

4. Алгоритмы обработки больших данных

А как использовать преимущества больших данных?

Эволюция средств хранения, обработки и использования данных безусловно улучшает качество пользовательских услуг и меняет подход специалистов к данным.
Но так же подобный объём данных может быть использован для извлечения информации и последующего формирования знаний.

Способов обработать данные и повысить уровень их упорядоченности довольно много. Данный процесс плохо поддаётся автоматизации в широком смысле этого слова.
Так или иначе для обработки произвольных данных необходимо провести первичный анализ и найти достаточно актуальное описание или эксперта в предметной области.

Программное обеспечение для обработки больших данных

Если учесть, что большие данные хранятся в распределенных системах, то классические программные средства первичной обработки применить затруднительно.
В качестве основы для обработки данных в паре с хранением в HDFS предлагается использовать MapReduce.

MapReduce – довольно специфическая технология определяющая ход распределенных вычислений. *Так же MapReduce можно определить как очень высокоуровневый абстрактный интерфейс, который должна поддерживать требуемая пользовательская операция над массивом данных.*

Программное обеспечение для обработки больших данных

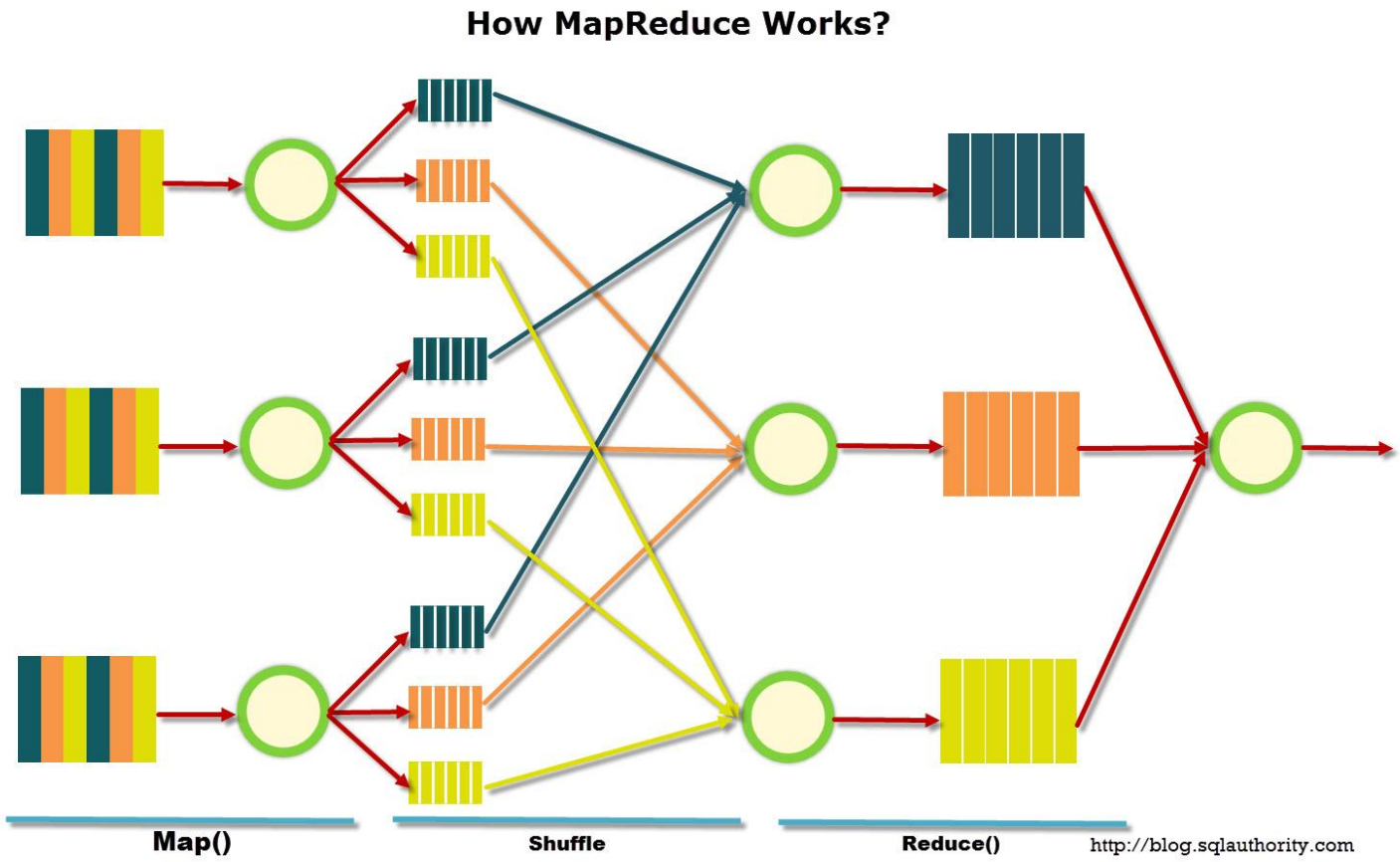
MapReduce – определяет процесс обработки данных который делится на 3 ключевые операции:

- **Map** - операция разметки данных. В рамках данной операции происходит выделение значения ключевой характеристики для каждого документа. Значение данной характеристики не должно зависеть от остальных документ . Расчёт значения характеристики будет производиться локально на узле, который хранит документ или его часть.

  Как правило результатом функции Map является одна или несколько пар формата ключ-значение.
- **Shuffle** – операция укрупненной сортировки блоков данных. Как правило полученные на стадии Map пары ключ значение разбираются на блоки с одинаковым ключом, что позволит их в дальнейшем обработать на стадии Reduce.
- **Reduce** – операция обработки полученных блоков из пар ключ-значение с одинаковым ключом.

Схема работы принципа MapReduce

MapReduce – определяет процесс обработки данных который делится на 3 ключевые операции:



Программное обеспечение для обработки больших данных

В общем виде как часто бывает операции подхода MapReduce выглядят не слишком очевидно. Давайте разберём эти задачи на характерном примере. Самым распространенным является пример с подсчетом количества вхождений каждого клова в корпус текстов (например библиотеку или википедию):

- **Map** – для каждого слова встреченного в документе добавить строку, где слово – ключ, а значение - 1.

мама мыла раму	(мама, 1) (мыла, 1) (раму, 1)
мама идет домой	(мама, 1) (идет, 1) (домой, 1)
курьер доставил посылку домой	(курьер, 1) (доставил, 1) (посылку,1) (домой, 1)

Программное обеспечение для обработки больших данных

В общем виде как часто бывает операции подхода MapReduce выглядят не слишком очевидно. Давайте разберём эти задачи на характерном примере. Самым распространенным является пример с подсчетом количества вхождений каждого слова в корпус текстов (например библиотеку или википедию):

- **Shuffle** – в случае со словами имеет смысл отсортировать их (упорядочить) по начальным буквам для того, чтобы максимальное количество пар с одним и тем же ключом оказались в одном блоке.

(мама, 1) (мыла, 1) (раму, 1)	(мама, 1) (мыла, 1) (мама, 1)
(мама, 1) (идет, 1) (домой, 1)	(раму, 1) (реквизит,1)
(курьер, 1) (доставил, 1) (реквизит,1) (домой, 1)	(идет, 1)
	(домой, 1) (домой, 1) (доставил, 1)
	(курьер, 1)

Программное обеспечение для обработки больших данных

В общем виде как часто бывает операции подхода MapReduce выглядят не слишком очевидно. Давайте разберём эти задачи на характерном примере. Самым распространенным является пример с подсчетом количества вхождений каждого слова в корпус текстов (например библиотеку или википедию):

- **Reduce** – функция reduce в данном случае будет рекурсивно обрабатывать список, осуществляя в нем поиск двух пар с одним ключом производя суммирования значения.

(мама, 1) (мыла, 1) (мама, 1)	(мама, 2) (мыла, 1)
(раму, 1) (реквизит,1)	(раму, 1) (реквизит,1)
(идет, 1)	(идет, 1)
(домой, 1) (домой, 1) (доставил, 1)	(домой, 2) (доставил, 1)
(курьер, 1)	(курьер, 1)

Программное обеспечение для обработки больших данных

Звучит как очень сложное решение для простой задачи.
Но такова суть высокопроизводительных параллельных вычислений. Для реализации обработки данных в рамках архитектуры HDFS подход MapReduce оказывается крайне эффективен. Принцип MapReduce был предложен компанией Google, но используется повсеместно и очень долгое время плотно ассоциировался с Apache Hadoop.

Расчет объёма продаж по регионам MapReduce:

- **Map** - По всем транзакциям сформировать пары (регион, размер оплаты), где регион – это Страна + Регион + Населенный пункт
- **Shuffle** – сформировать блоки данных по ключу Региона.
- **Reduce** – Суммировать суммы трат по одинаковым значениям ключа регион.

? Архитектура информационных систем построенных на Big Data
Контрольные вопросы:

1. Какие 3 операции входят в принцип MapReduce?

2.Почему нельзя использовать SQL-запрос для получения агрегированных показателей по массивам данных?

3. В чем смысл операции Map?

4. В чем смысл операции Shuffle?

5. В чем смысл операции Reduce?

6. Как вы можете объяснить, что вычисления в рамках MapReduce локализованы?

5. Использование языка Python в обработке больших данных

Библиотеки языка Python

Именно с изучения «питона» начинают свой путь в программировании начинающие пользователи, а профессионалы используют его для решения широкого спектра задач — от научных исследований до веб-разработки и искусственного интеллекта. Он никогда никого не разочаровывал, когда дело касалось анализа данных, визуализации, интеллектуального анализа данных и т. д. Python имеет низкий порог входа, и именно поэтому он приобрел популярность за последние несколько лет. Будучи языком программирования с открытым исходным кодом, Python также имеет обширный набор библиотек, которые идеально подходят для специалистов по обработке данных и позволяют им без каких-либо проблем выполнять практически любую задачу.

Его база библиотек насчитывает более 137000 библиотек.

Для работы с большими данными можно выделить следующие:

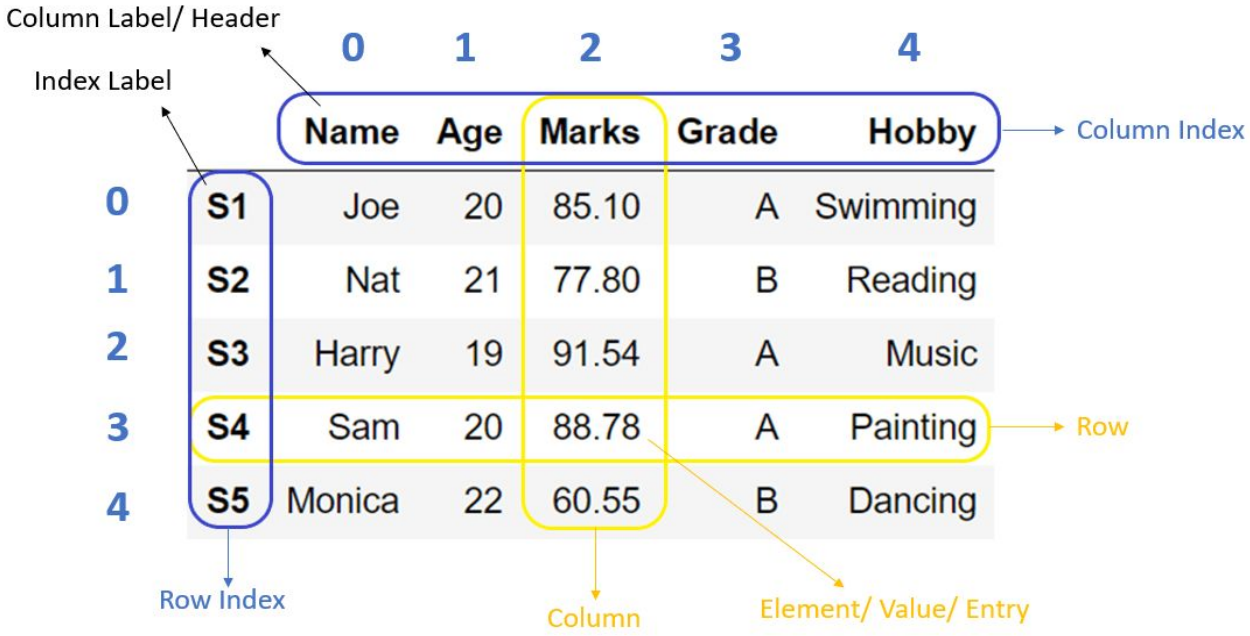
- TensorFlow
- Pandas
- Matplotlib
- NumPy
- SciPy

pandas

Pandas - библиотека для обработки и анализа данных. Она предоставляет структуры данных, такие как DataFrame и Series, позволяя удобно работать с табличными данными.

Основные возможности:

- Загрузка и сохранение данных из различных источников.
- Мощные средства для фильтрации, сортировки и группировки данных.
- Обработка пропущенных значений.
- Простой доступ и изменение данных по индексам.

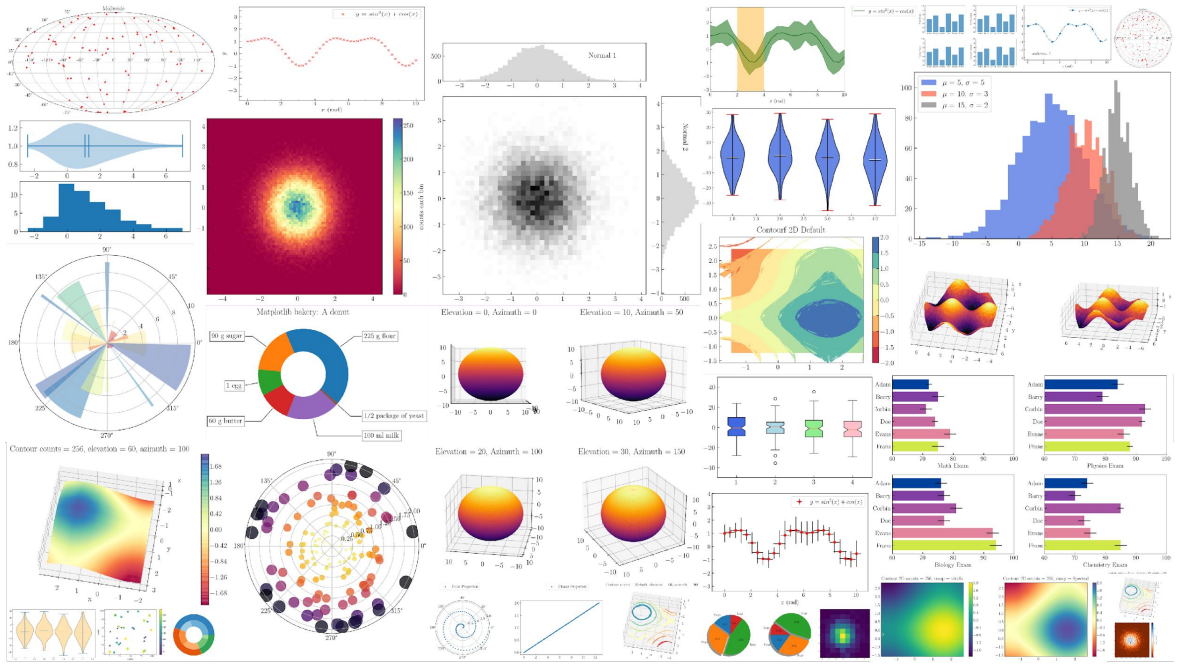


matplotlib

Matplotlib - библиотека для создания различных видов графиков и визуализации данных в Python. Она предоставляет широкий спектр инструментов для создания статических, интерактивных и анимационных графиков.

Основные возможности:

- Графики линий, точечные диаграммы, гистограммы, круговые диаграммы и другие виды графиков.
- Настраиваемый внешний вид графиков, включая цвета, шрифты, подписи и легенды.
- Возможность создания подграфиков и сложных композиций.

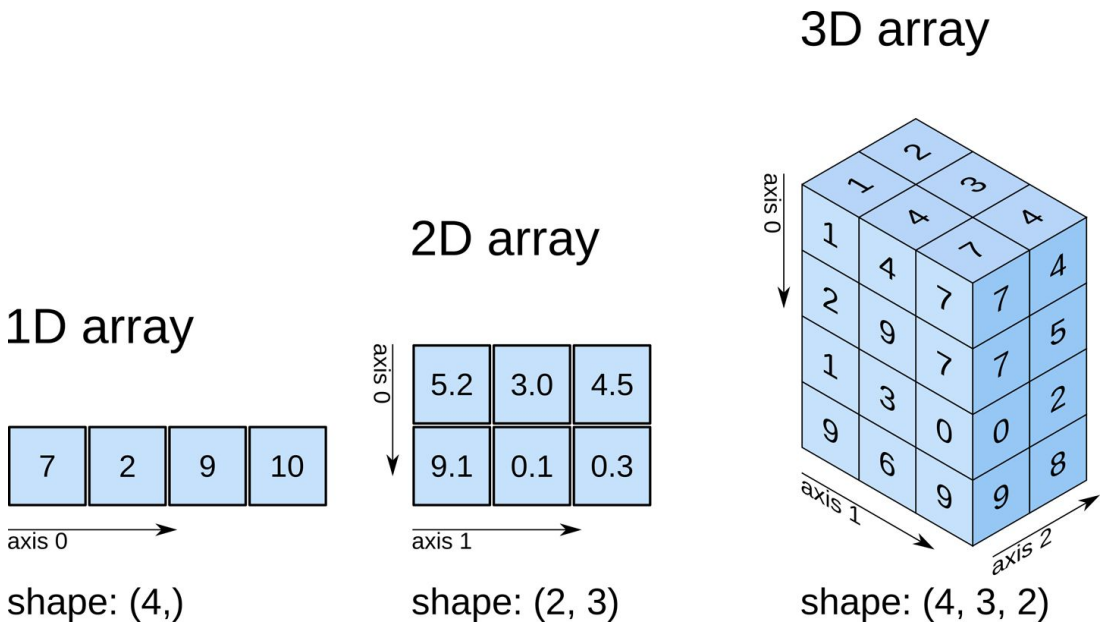


NumPy

NumPy - библиотека для работы с массивами и матрицами, предоставляя высокоэффективные операции над ними. Она является основой для многих других библиотек в области научных вычислений в Python.

Основные возможности:

- Многомерные массивы (ndarrays) с эффективными операциями над ними.
- Математические функции для выполнения операций на массивах.
- Инструменты для интеграции с кодом на C/C++ и Fortran.

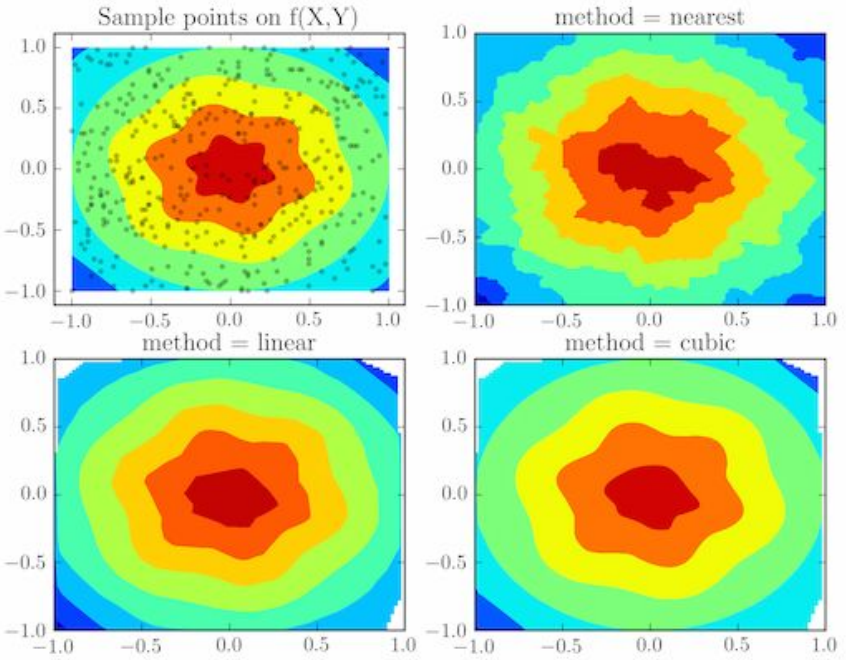


SciPy

SciPy - библиотека для выполнения научных и технических вычислений в Python. Она предоставляет функционал поверх NumPy, включающий различные методы оптимизации, обработку сигналов, статистику и многое другое.

Основные возможности:

- Модули для оптимизации, интерполяции, интеграции и решения дифференциальных уравнений.
- Функции для обработки сигналов, анализа спектров и статистических методов.
- Инструменты для работы с изображениями и звуками.



TensorFlow

Фреймворк TensorFlow представляет собой инструмент относительно легкого уровня сложности, который обеспечивает быстрое создание нейронных сетей различных уровней сложности. Он идеально подходит для новичков благодаря множеству примеров и предварительно созданных моделей машинного обучения, готовых к интеграции в различные приложения. В то же время, опытным разработчикам TensorFlow предоставляет многочисленные настройки и API для оптимизации процесса обучения.

TensorFlow поддерживает несколько языков программирования, основным из которых является Python. Кроме того, существуют отдельные пакеты для языков C/C++, Golang и Java. Возможности TensorFlow не ограничиваются только этими аспектами. Библиотеку можно использовать для обучения моделей на смартфонах и умных устройствах с использованием TensorFlow Lite, а также для создания корпоративных нейронных сетей с помощью TensorFlow Extended.

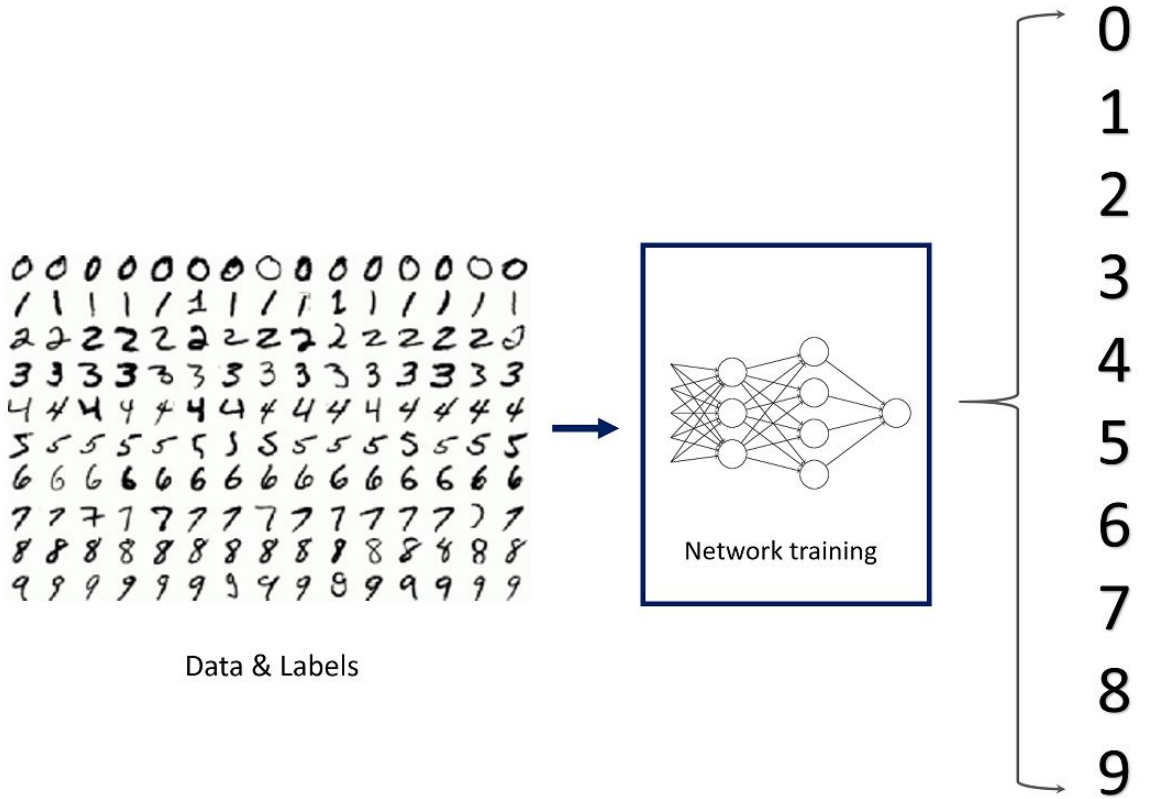
Для создания простой нейронной сети на TensorFlow достаточно понимания нескольких основных принципов:

- Принципы машинного обучения;
- Процесс обучения нейронных сетей и используемые методы;
- Общий взгляд на процесс обучения в TensorFlow.

TensorFlow

Создание простой нейронной сети

Задача: построить нейронную сеть, которая будет по картинке классифицировать цифру



TensorFlow

Создание простой нейронной сети

Шаг 1 – импортирование библиотеки TensorFlow

```
import tensorflow as tf
print("TensorFlow version:", tf.__version__)
```

```
TensorFlow version: 2.14.0
```

TensorFlow

Создание простой нейронной сети

Шаг 2 – загрузка набора данных для обучения и тестирования

Исходные данные: база данных MNIST — объёмная база данных образцов рукописного написания цифр.

```
mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 0s 0us/step
```

TensorFlow

Создание простой нейронной сети

Шаг 3 – построение модели и определение архитектуры нейронной сети

```
model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10)
])
```

TensorFlow

Создание простой нейронной сети

Шаг 4 – компиляция модели и выбор функции потерь, оптимизатор и метрики

```
model.compile(optimizer='adam',
              loss=loss_fn,
              metrics=['accuracy'])
```

TensorFlow

Создание простой нейронной сети

Шаг 5 – обучение модели, загружаем данные в модель и проводим обучение

```
model.fit(x_train, y_train, epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2929 - accuracy: 0.9161
Epoch 2/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.1391 - accuracy: 0.9585
Epoch 3/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.1030 - accuracy: 0.9689
Epoch 4/5
1875/1875 [==============================] - 9s 5ms/step - loss: 0.0845 - accuracy: 0.9738
Epoch 5/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0738 - accuracy: 0.9770
<keras.src.callbacks.History at 0x7e94b5922fe0>
```

TensorFlow

Создание простой нейронной сети

Шаг 6 – оценка модели

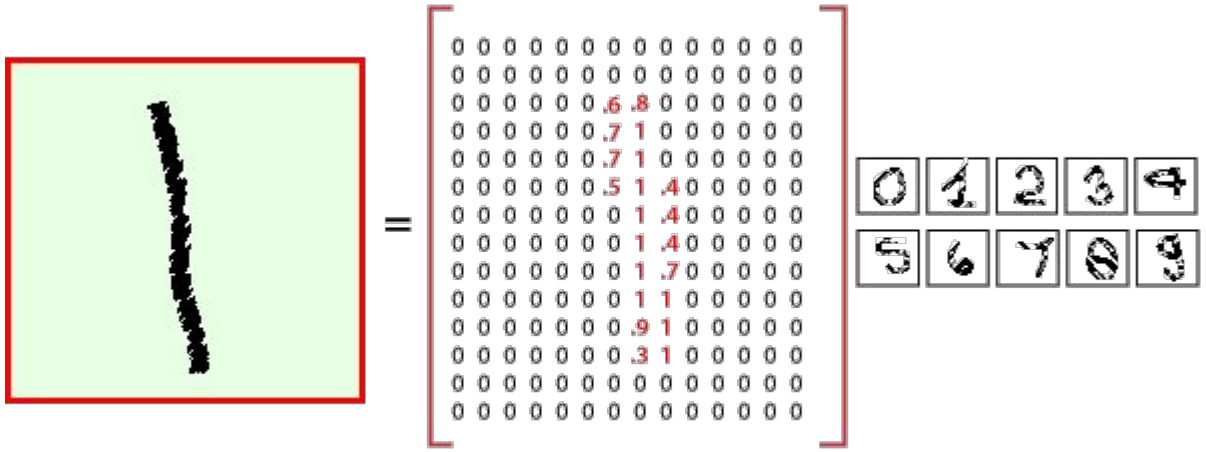
```
[11] model.evaluate(x_test,  y_test, verbose=2)

     313/313 - 1s - loss: 0.0745 - accuracy: 0.9774 - 596ms/epoch - 2ms/step
     [0.07447012513875961, 0.977400004863739]
```

TensorFlow

Создание простой нейронной сети

Итогом является модель нейронной сети, которая может определить рукописную цифру с точностью в 98%. Теперь мы можем в неё загрузить картинку рукописной цифры и модель определит, что это за цифра!



Самостоятельная работа

В рамках выполнения первого задания для слушателей без активных навыков разработки программного обеспечения. Необходимо самостоятельно изучить ПРИНЦИП РАБОТЫ сортировки слиянием.

1. Визуализация (Merge sort!!!) - https://www.cs.usfca.edu/~galles/visualization/ComparisonSort.html
2. https://neerc.ifmo.ru/wiki/index.php?title=%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0_%D1%81%D0%BB%D0%B8%D1%8F%D0%BD%D0%B8%D0%B5%D0%BC

Домашнее задание

1. **Если навыков разработки программного обеспечения нет.**

1.1 Задача сформировать понимание работы распределенных вычислений. Необходимо выполнить групповое задание на сортировку слиянием. ВАЖНО! Задание выполняется в ручном режиме! Результат должен быть сформирован последовательно, через общий документ.

1.2 Задача посчитать общую стоимость товаров по брендам в разделе электроники. Расчет необходимо производить по принципу классической задачи о вхождении слов в тексты большого корпуса документов.

2. **Если есть навыки разработки программного обеспечения.**

По инструкции развернуть локальный Hadoop кластер на своей машине.

Ссылка на инструкцию – https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html

По шагам приложить скриншоты получивших действий.

Финальным результатом служит документ развернутый на 3 узлах.

Список использованных источников и литературы:

1. Документация Apache Hadoop - https://hadoop.apache.org/docs/stable/

2. Видео материалы по работе архитектуры HDFS - https://www.youtube.com/c/DataflairWS/videos (English)

3. Вводный материал о big data - https://www.oracle.com/ru/big-data/what-is-big-data/

4. Финансовый отчет Netflix - https://ir.netflix.net/ir-overview/profile/default.aspx

5. Оперативная оценка пользовательских трат за октябрь 2021 года. Сбербанк - https://www.sberbank.ru/common/img/uploaded/files/pdf/analytics/sberindex_oct.pdf

6. Big Data Challenges in Genome Informatics - https://www.researchgate.net/publication/324055666_Big_Data_Challenges_in_Genome_Informatics

7. MapReduce общая схема - https://blog.sqlauthority.com/2013/10/09/big-data-buzz-words-what-is-mapreduce-day-7-of-21/

Список использованных источников и литературы:

8. Язык программирования Python – https://www.python.org/

9. Библиотека Pandas – https://pandas.pydata.org/

10. Библиотека Matplotlib – https://matplotlib.org/

11. Библиотека Numpy – https://numpy.org/

12. Библиотека SciPy – https://scipy.org/

13. Библиотека TensorFlow – https://www.tensorflow.org/?hl=ru

14. Пример нейронной сети в google colab – https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/quickstart/beginner.ipynb#scrollTo=7NAbSZiaoJ4z

Спасибо за внимание!

ИНСТИТУТ РЕГИОНАЛЬНОГО РАЗВИТИЯ

НОЦ ИНЖЕНЕРИЯ БУДУЩЕГО

Ростех

РОСКОСМОС

РЖД