**Выжимка самого главного, что мы должны знать:**

**Machine Learning Engineer** – это эксперт в области искусственного интеллекта. Именно он разрабатывает алгоритмы, по которым «думает» компьютер.

Где мы можем в обычной жизни встретить ИИ?

* Siri
* Олег(Тинькофф)
* Алиса(Яндекс)

Какими компетенциями должен обладать ML специалист:

* Высокие математические навыки.
* Базис программирования.
* Моделирование данных.
* SQL.
* Английский язык

Soft Skills:

* аналитический склад мышления, логика;
* коммуникативность;
* инициативность;
* внимательность к деталям.

Данные о зарплате:

Медиана проходит по точке в 165 000 рублей(специалист с 1–2-летним опытом в ML)

Junior ML-engineer - от 80 тысяч рублей.

Опытный сеньор - 200 000 рублей. В крупных международных компаниях - 330 000 рублей.

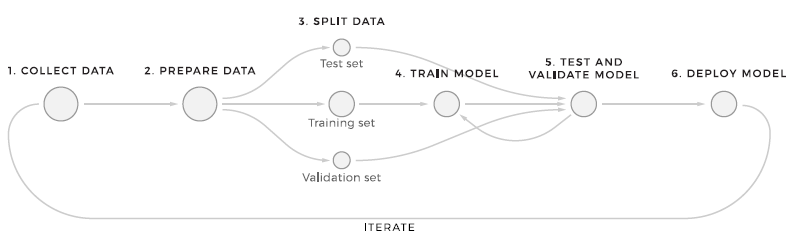
**ИИ** - это упрощенный процесс решения проблем для людей. Эта технология позволяет программному обеспечению выполнять задачи без явного участия специалиста. ИИ также включает нейронные сети и глубокое обучение.

**Машинное обучение** – это приложение искусственного интеллекта (ИИ), которое дает возможность машинам получать доступ к данным и позволяет им учиться выполнять определенные задачи. ИИ использует алгоритмы и позволяет системам находить скрытые смыслы без постороннего вмешательства.

## **Живое общение с ИИ**

Обработка естественной речи (NLP) находится на пике развития, в этой технологии разработчики добились впечатляющих успехов, позволяющих машинам создавать текстовую информацию на основе случайного начального ввода. Фактически одно такое решение NLP может писать невероятно убедительные стихи, рассказы и новостные статьи. На 2022-24 г.г. планируется, что предстоящий прогресс приведет к созданию разговорного процесса, что позволит компаниям удовлетворять конкретные потребности клиентов, задавая вопросы о продуктах или услугах компании.

**Процесс подготовки машинного обучения**

****

1. **Сбор данных**

Специальная команда дата-саентистов или людей, имеющих знания в предметной области бизнеса, определяют данные, которые будут использоваться для обучения.

1. **Подготовка данных и проектирование признаков:**

Cобранные [данные подвергаются серии преобразований](https://www.altexsoft.com/blog/datascience/preparing-your-dataset-for-machine-learning-8-basic-techniques-that-make-your-data-better/): форматирование, очистка, разметка и обогащение данных часто выполняется вручную. После подготовки данных дата-саентисты приступают к проектированию признаков. Признаки — это значения данных, которые модель будет использовать и при обучении, и в продакшене. Дата-саентисты исследуют имеющиеся данные, определяют, какие атрибуты имеют бОльшую предсказательную силу, а затем создают список признаков.

1. Выбор алгоритма:

зачастую этот этап выполняется параллельно с предыдущими, поскольку выбор алгоритма — одно из первоначальных решений в ML. В основе любой модели лежит математический алгоритм, определяющий, как модель будет находить паттерны в данных.

1. Обучение модели:

Чтобы обучить модель делать прогнозы на основании новых данных, дата-саентисты подгоняют её под имеющиеся данные, на которых она учится.

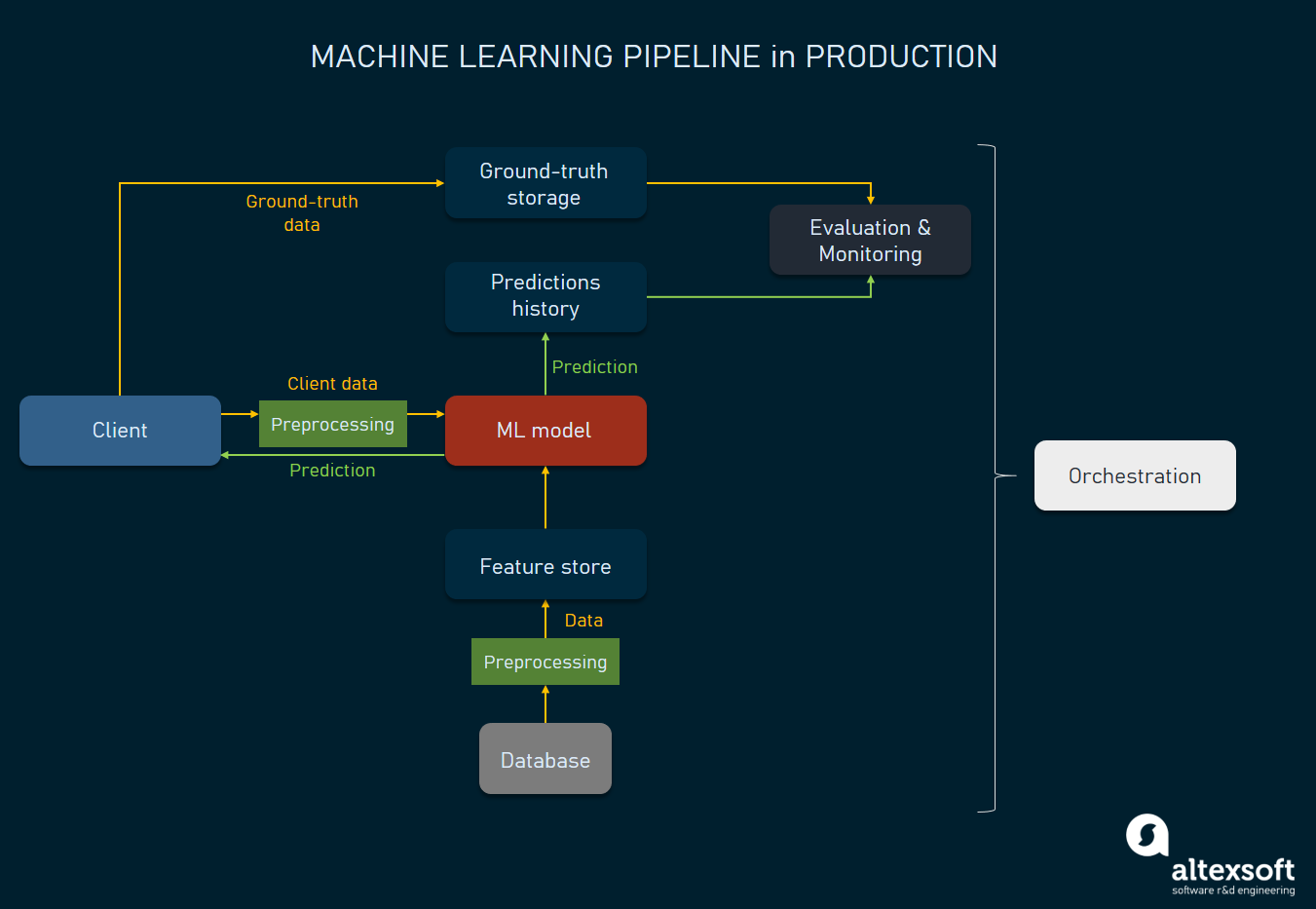
1. Тестирование и валидация:

Обученные модели тестируются на данных тестирования и валидации, чтобы убедиться в высокой точности прогнозов. После сравнения результатов тестов модель могут настраивать/модифицировать/обучать на других данных. Обучение и оценка — это итеративные этапы, продолжающиеся, пока модель не достигнет приемлемого процента правильных прогнозов.

1. Развёртывание:

Применение модели ML в продакшене. То есть, по сути, это использование модели конечным пользователем для получения прогнозов, сгенерированных на реальных данных.

## **Конвейер машинного обучения в продакшене**



### **Вызов модели из клиента приложения**

Для описания потока продакшена мы воспользуемся в качестве начальной точки *клиентом приложения*. При наличии приложения, для которого модель генерирует прогнозы, конечный пользователь будет взаимодействовать с ним через клиент. Работа модели запускается, когда пользователь совершает определенное действие или предоставляет входящие данные. Например, если алгоритм машинного обучения занимается рекомендациями продуктов на веб-сайте онлайн-магазина, то клиент (веб-приложение или мобильное приложение) будет отправлять подробности текущей сессии, например, какие продукты или разделы продуктов сейчас изучает пользователь.

**Клиент приложения:** отправляет данные серверу модели.

### **Получение дополнительных данных из хранилища признаков**

Хотя данные получаются со стороны клиента, некоторые дополнительные признаки могут также храниться в отдельной базе данных — *хранилище признаков (feature store)*. Это хранилище признаков предоставляет модели быстрый доступ к данным, доступ к которым невозможен с клиента. Например, если онлайн-магазин рекомендует продукты, купленные другими пользователями со схожими вкусами и предпочтениями, хранилище признаков предоставит модели соответствующие признаки.

В свою очередь, хранилище признаков получает данные от других хранилищ (пакетно или в реальном времени) при помощи [потоков данных](https://www.altexsoft.com/blog/real-time-analytics/). Пакетная обработка — обычный способ извлечения данных из баз данных, получение требуемой информации по частям.

**Хранилище признаков:** предоставляет модели дополнительные признаки.

### **Предварительная обработка данных:**

Приходящие от клиента приложения данные поступают в сыром формате. Чтобы модель смогла прочитать эти данные, нам нужно обработать их и преобразовать в признаки, которые может потребить модель. Этот процесс базовых преобразований данных называется *предварительной обработкой данных (data preprocessing)*.

**Препроцессор данных:** данные, переданные от клиента приложения и хранилища признаков, форматируются, из них извлекаются признаки.

### **Генерация прогнозов**

Наконец, когда модель получит все признаки от клиента и хранилища признаков, она генерирует прогноз и отправляет его клиенту, а также отдельной базе данных для дальнейшей оценки.

**Модель:** прогноз отправляется клиенту приложения.

### **Сохранение эталонных данных и данных прогнозов**

Ещё один тип данных, которые мы хотим получить — это эталонные данные. Эти данные используются для оценки прогнозов, сделанных моделью, и для дальнейшего совершенствования модели. Эталонными данными можно назвать те данные, в которых мы уверены, например, реальный продукт, который когда-то купит посетитель. Для хранения этой информации используется база эталонных данных.

Однако сбор эталонных данных не всегда возможен, а иногда его нельзя автоматизировать. Например, продукт, купленный посетителем, будет эталонными данными, с которыми можно сравнить прогнозы модели. Но если посетитель увидел вашу рекомендацию и купил продукт в каком-то другом магазине, вы не сможете собрать подобные эталонные данные.

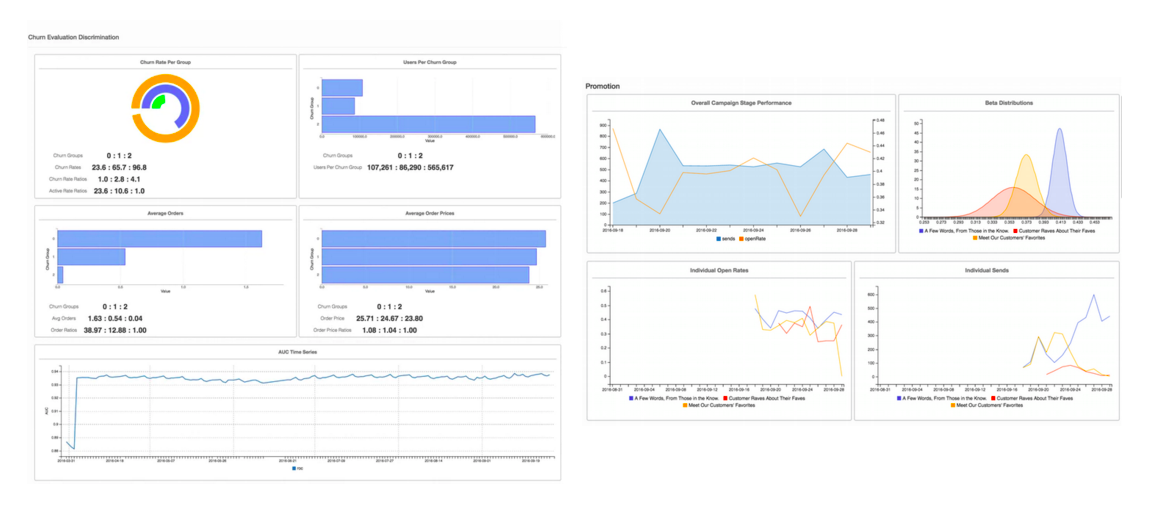
**База эталонных данных:** хранит эталонные данные.

### **Мониторинг и оценка модели**

Модель, работающая на продакшен-сервере, будет иметь дело с реальными данными и предоставлять пользователям прогнозы. При мониторинге нам важно следующее:

* обеспечивать сохранение высокой точности прогнозов относительно эталонных данных.
* внимательно изучать точность и быстродействие модели.
* понимать, когда модели нужно повторное обучение.

Инструменты мониторинга часто создаются из библиотек [визуализации данных](https://www.altexsoft.com/blog/data-visualization-tools-types-techniques/), обеспечивающих наглядные визуальные метрики точности. Их интерфейс может походить на аналитический дэшборд:



**Инструменты мониторинга:** предоставляют метрики о точности прогнозов и показатели моделей.

### **Оркестраторы**

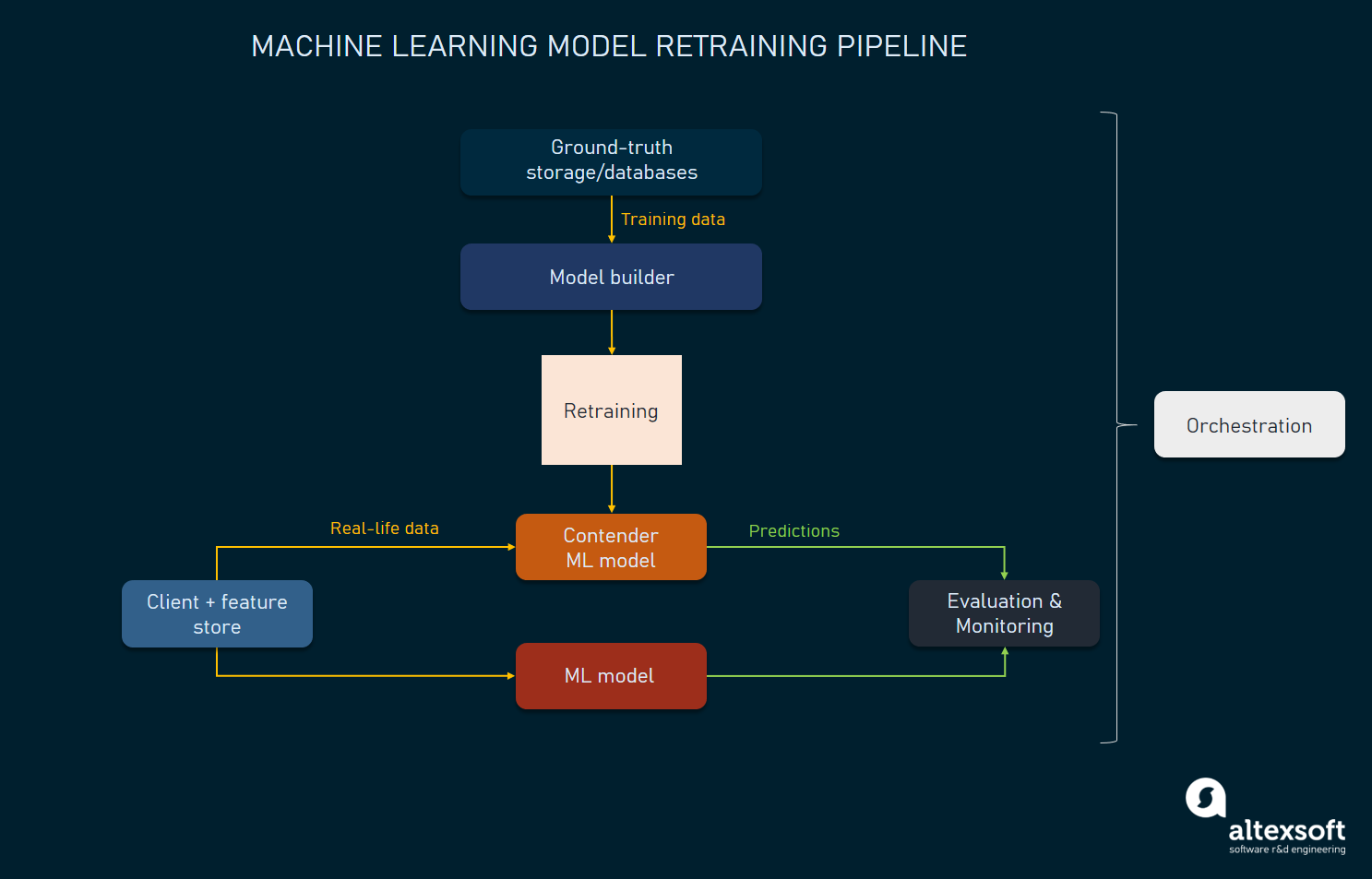
Оркестратор — это инструмент, выполняющий все процессы машинного обучения на всех этапах. То есть он обеспечивает полный контроль над развёртыванием моделей на сервере, управляет их работой, потоками данных и активацией процесса обучения/повторного обучения.

Оркестраторы — это инструменты, выполняющие по графику скрипты и запускающие все задачи, связанные с моделью машинного обучения в продакшене. Популярными оркестраторами моделей ML являются [Apache Airflow](https://airflow.apache.org/), [Apache Beam](https://beam.apache.org/) и [Kubeflow Pipelines](https://www.kubeflow.org/docs/pipelines/overview/pipelines-overview/).

**Оркестратор:** отправляет команды для управления всем процессом.

## Конвейер повторного обучения модели ML

Модели обучаются на предыдущих данных, которые со временем устаревают. Точность прогнозов начинает снижаться, это можно отследить при помощи инструментов мониторинга. Когда точность становится слишком низкой, нам нужно повторно обучить модель на новых массивах данных. Этот процесс можно выполнять по графику, чтобы повторное обучение проходило автоматически.



***Как выполняется повторное обучение моделей***

### **Повторное обучение моделей**

Повторное обучение — это ещё одна итерация жизненного цикла модели, в которой, по сути, используются те же методики, что и при обучении. При снижении точности прогнозов можно обучить модель на обновлённых массивах данных, чтобы она могла обеспечивать более точные результаты.При повторном обучении обычно используется тот же алгоритм, но ему передаются новые данные.

Все процессы, выполняемые на этапе повторного обучения до развёртывания модели на продакшен-сервере, управляются оркестратором. То есть мы можем управлять массивом данных, подготавливать алгоритм и начинать обучение. Для повторного обучения при помощи предоставления входящих данных используется сборщик модели. По сути, это автоматизирует процесс обучения, чтобы мы могли выбрать на этапе оценки наилучшую модель.

**Оркестратор:** отправляет модели на повторное обучение. Формирует новые массивы данных. Передаёт данные, собранные в базах эталонных данных/хранилищах признаков.

**Сборщик модели:** повторно обучает модели по заданным свойствам.

### **Оценка моделей-претендентов и отправка их в продакшен**

Прежде чем повторно обученная модель сможет заменить старую, её нужно оценить, сравнив с базовыми метриками: точностью, скоростью работы и так далее.

*Программа оценки (evaluator)* — это ПО, проверяющее, готова ли модель к внедрению в продакшен. Оно может предоставлять метрики точности прогнозов или сравнивать новые обученные модели со старыми при помощи реальных и эталонных данных. Результаты модели-претендента могут отображаться при помощи инструментов мониторинга. Если модель-претендент превосходит параметры свой предшественницы, её можно развёртывать в продакшене. Цикл завершается.

**Программа оценки:** выполняет оценку обученных моделей, чтобы определить, лучше ли её прогнозы, чем у базовой модели.

**Оркестратор:** развёртывает модели в продакшене.