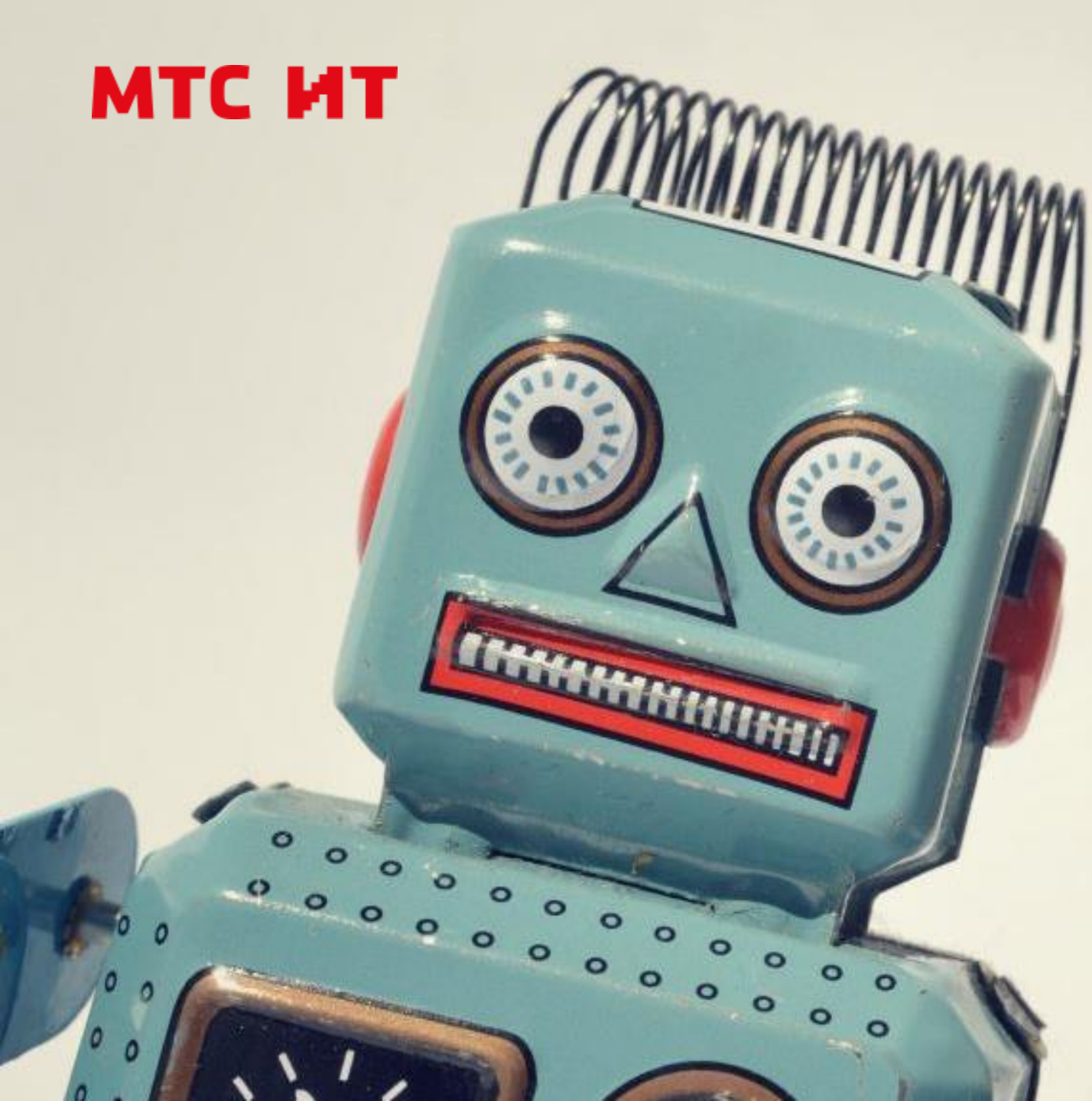
МТС IT



Помочь всем человекам.

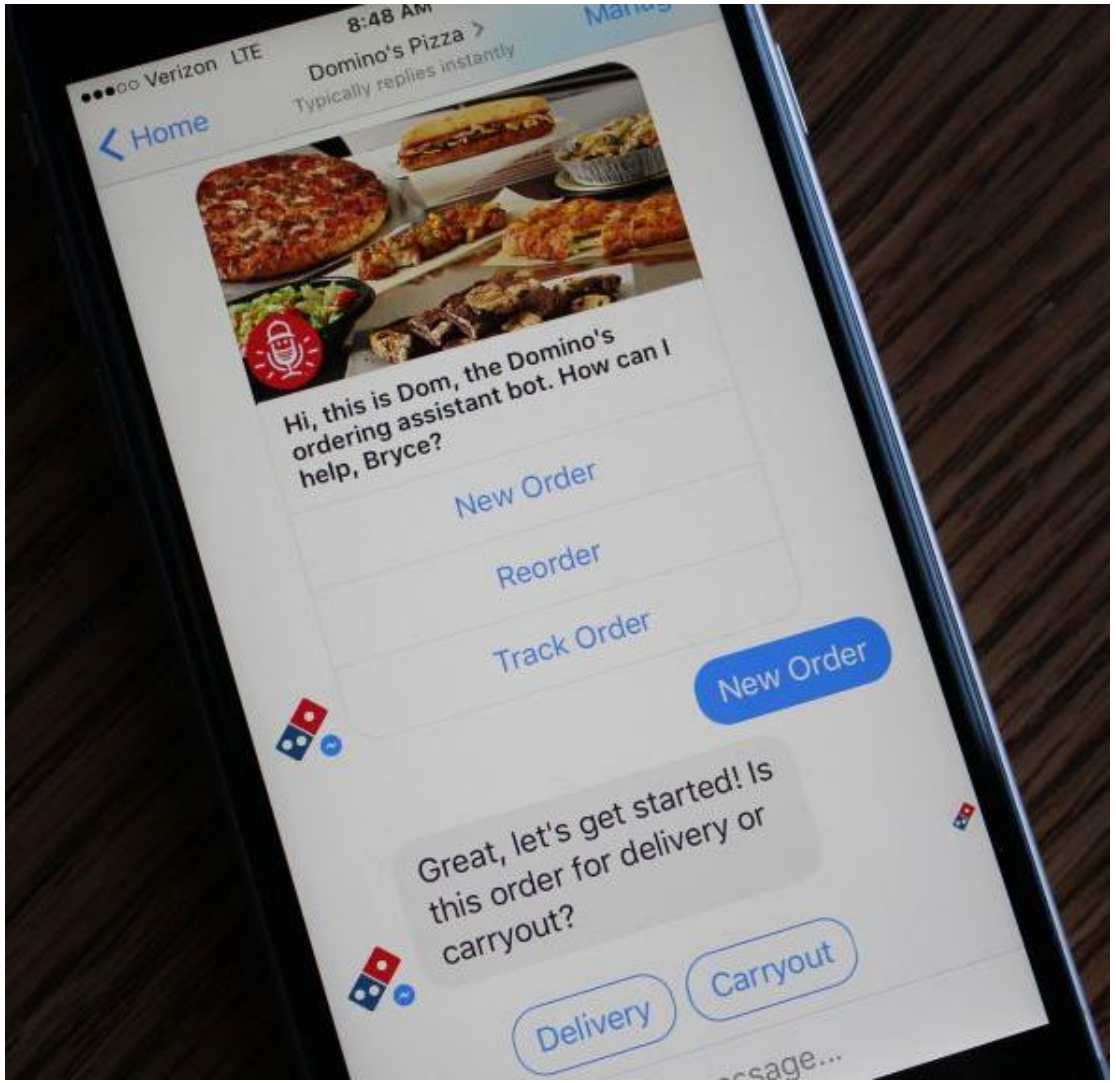
Результаты опроса западноевропейских абонентов, проведенного McKinsey: перевод взаимодействия в цифровые каналы повышает уровень удовлетворенности клиентов

Традиционные дистанционные каналы: телефон, почта или факс, электронная почта, звонок с сайта

Чат, форум, раздел «Часто задаваемые вопросы», личный кабинет, виртуальный помощник, социальные сети

Кнопочные боты

- Следуют жёсткому сценарию.
- Ограниченное число опций в одном сообщении.
- Пользователю нужно больше думать.



Боты с NLU

- Могут понять намерение пользователя по вопросу, заданному естественным языком.
- Пользователь не думает, какой пункт меню выбрать, чтобы получить нужную информацию.
- Также необходим жёсткий сценарий.

Conversational Platform

Платформа чат-бота

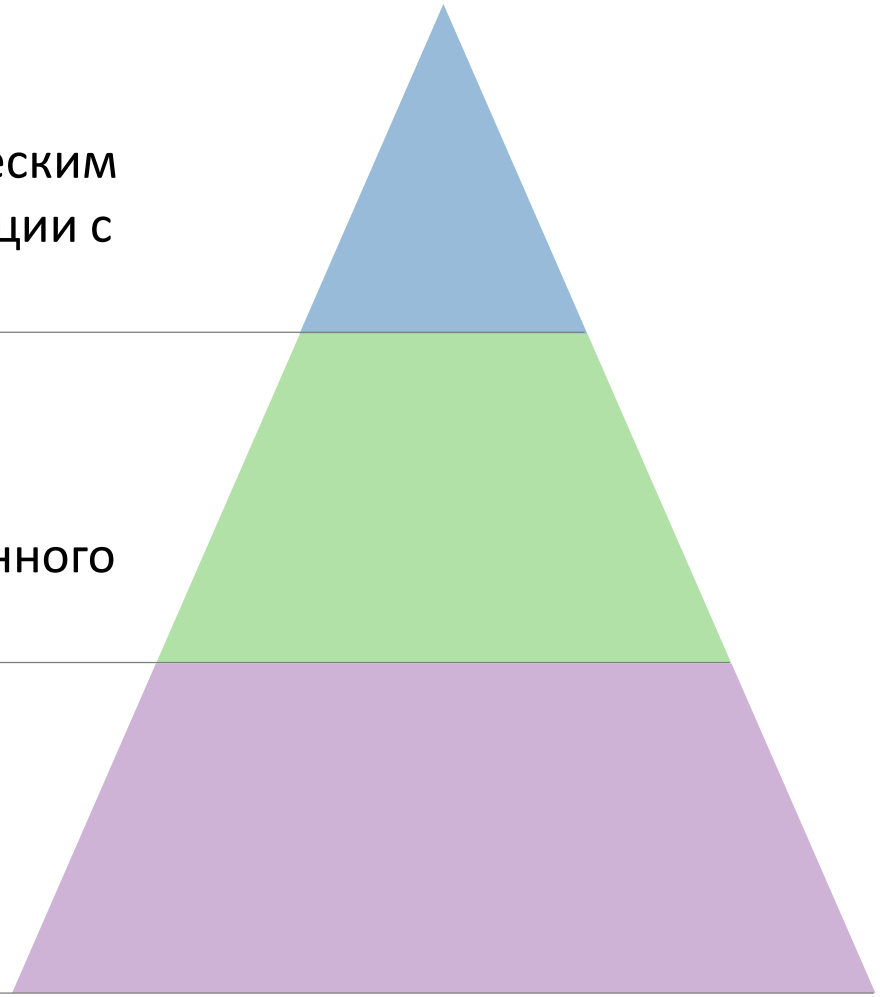
Содержит мощный движок выполнения сценариев с графическим редактором и неограниченными возможностями по интеграции с целевым ИТ-ландшафтом.

Когнитивные сервисы

Выполняют анализ намерения с помощью технологий машинного обучения.

Лингвистические сервисы

Низкоуровневые службы для морфологического и синтаксического анализа текстов.



Традиционный подход к пониманию текста

Морфологический и синтаксический анализ, системы на основе правил.

- Долго создавать правила.
- Нужны специально обученные лингвисты.
- Сложно дорабатывать.

Машинное обучение спешит на помощь!

- Берём логи общения с живым оператором.
- Размечаем сообщения по тематикам (без лингвистов!)
- МАГИЯ.

Облако или on-premise?

- Для быстрого старта облако отлично работает.
- Если в компании много задач и предвидится масштабирование, то уже не очень.
- С Microsoft LUIS мы бы платили около 700 000 руб. в месяц только за чат-бота клиентского сервиса.

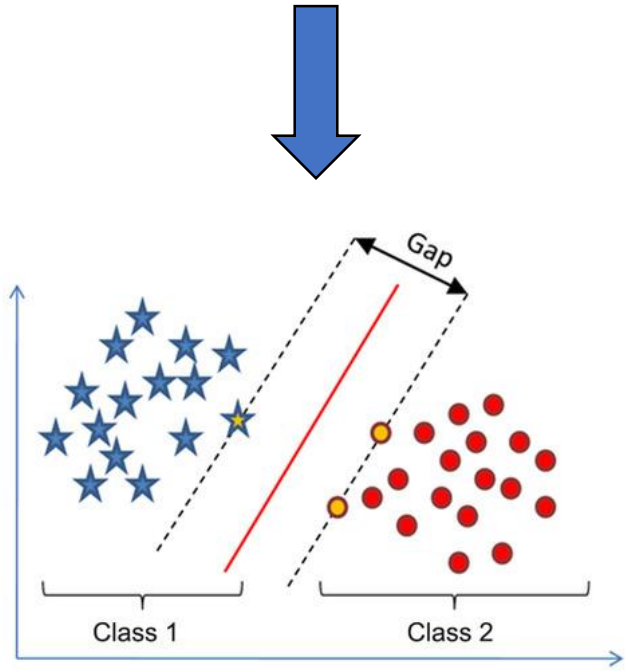
Что делать с текстом?

- Алгоритмы машинного обучения работают с векторами и матрицами.
- В каждой фразе разное количество слов, а в каждом слове – разное количество букв.
- Сначала мы научимся переводить любую фразу в вектор фиксированной длинны, а потом обучим классификатор правильно обрабатывать эти векторы.

$$\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} \quad \begin{bmatrix}1&0&0\\0&0&1\\0&1&0\end{bmatrix}$$

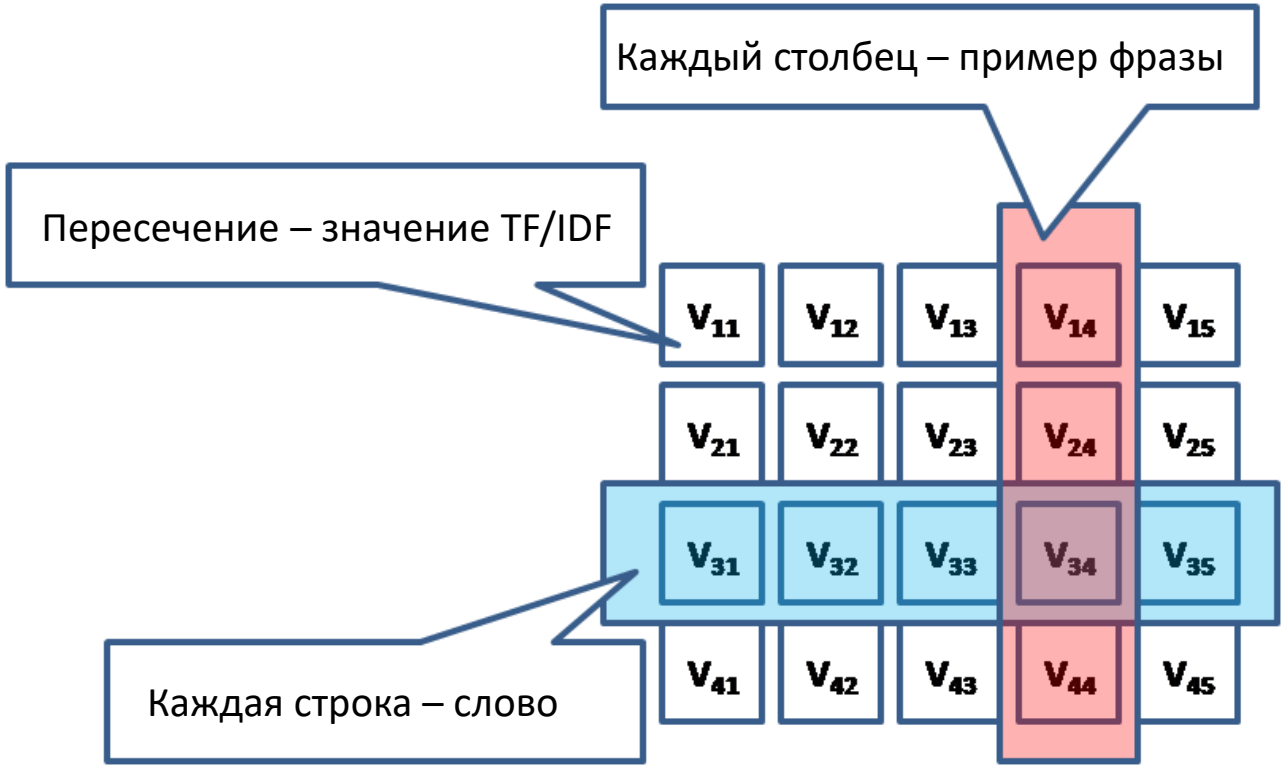
$$\begin{bmatrix}0&1&0\\1&0&0\\0&0&1\end{bmatrix} \quad \begin{bmatrix}0&1&0\\1&-1&1\\0&1&0\end{bmatrix} \quad \begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}$$

$$\begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix} \quad \begin{bmatrix}0&0&1\\0&1&0\\1&0&0\end{bmatrix}$$



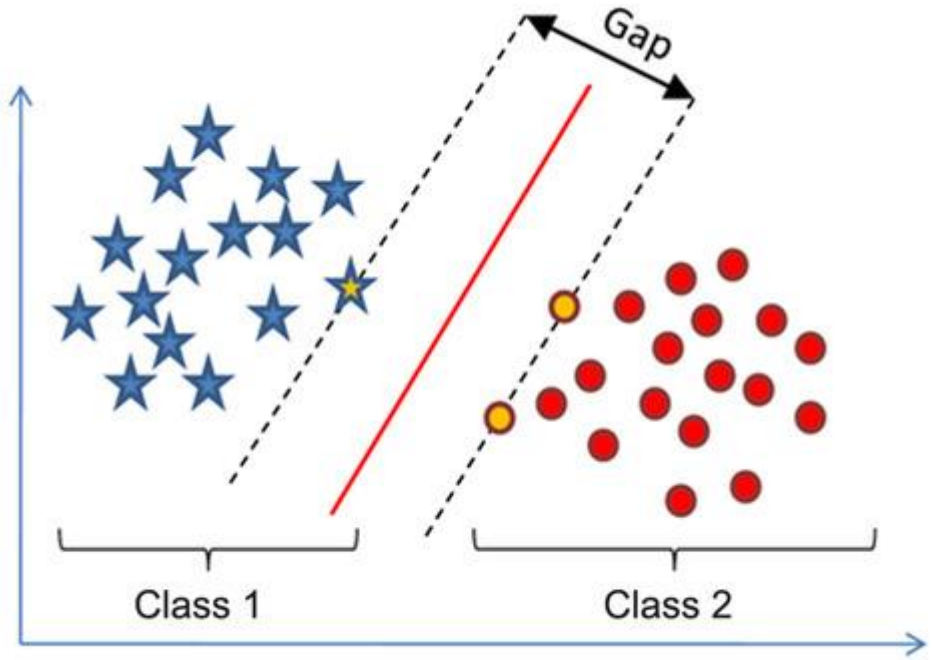
Простой алгоритм – TF/IDF

- Сначала составляем «словарь» из всех слов, которые встречаются в примерах.
- В каждом примере для каждого слова считаем количество его вхождений.
- Редким словам даём больший вес, частым – меньший.
- Каждая фраза становится вектором, каждый элемент которого – количество вхождений конкретного слова в конкретной фразе.



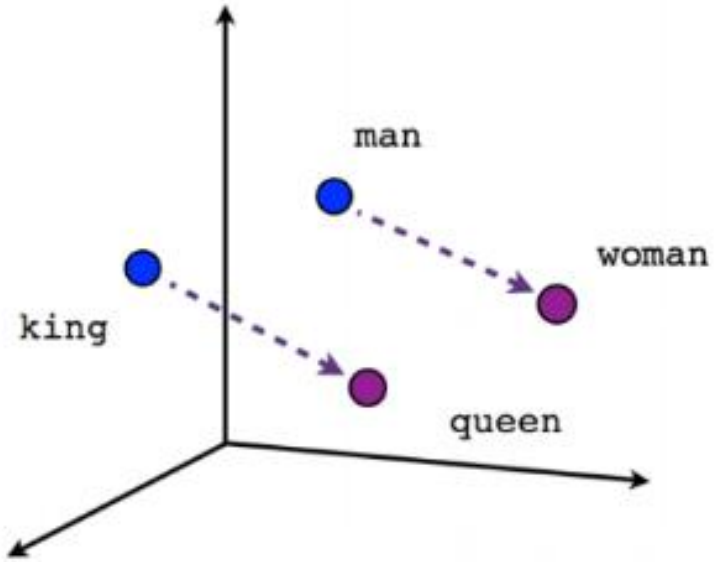
Простой классификатор – SVM

- Support Vector Machines – достаточно простой, но в то же время мощный классификатор, который может давать хорошие результаты в сочетании с TF/IDF.
- Быстро обучается и работает с нелинейными случаями.



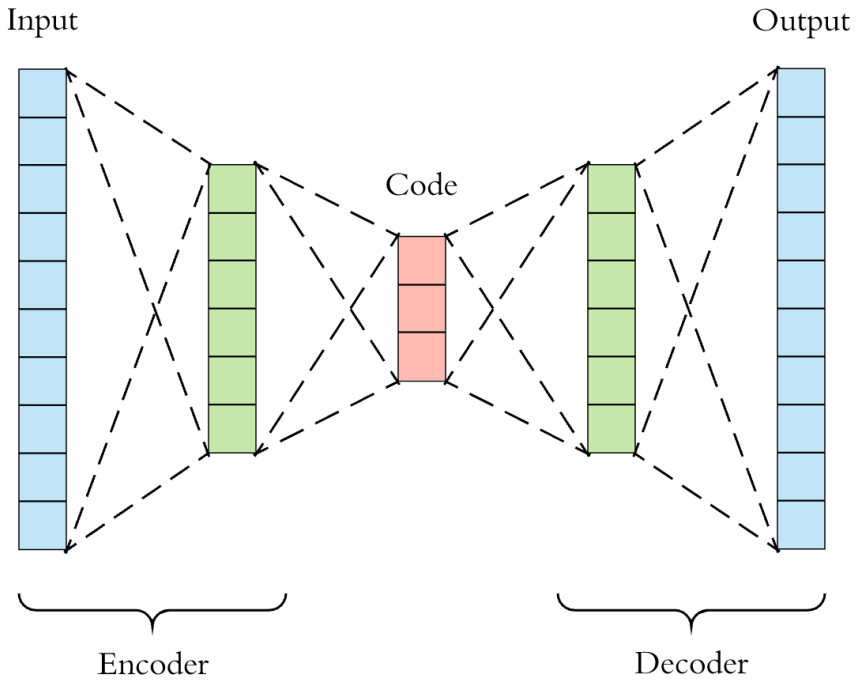
Векторное представление слов с word2vec

- Нейросеть выучивает векторные представления слов на большом корпусе неразмеченного текста без участия человека.
- Такие векторы обладают интересными свойствами. Например, близкие по значению слова располагаются близко в векторном пространстве. Это даёт классификатору «знание» о синонимах.
- Направления в векторном пространстве также имеют смысл. Это позволяет делать преобразования слов без участия традиционного тезауруса.



Более сложные модели

- Чтобы модель могла учитывать порядок слов в предложении, можно использовать рекуррентные нейронные сети.
- Обучение такой сети занимает много времени, и это невыгодно, если нужно часто добавлять новые примеры и классы.
- Модель можно разбить на две части: рекуррентный автоэнкодер, который учится упаковывать фразу в один вектор, и перцептрон, который можно быстро обучить классификации такого вектора.



Почему полностью своё решение?

- Для русского языка мало готовых компонентов. Почти все на Питоне.
- Мы честно написали прототип на Rasa NLU (https://rasa.com). Самая сложная модель, которая была доступна на тот момент — усреднённый word2vec.
- Своё решение легко поддерживать и развивать.

Пайплайны

```
public IChainablePipeline CreateMorphologyPipeline()
{
    return new Pipeline(NluRequirements.Input)
            .Chain(Item<Tokenizer>())
            .Chain(Item<NumberSplitter>())
            .Chain(Item<MorphologyAnalyzer>()) // First stab at lemmatization
            .Chain(Item<SpellChecker>()) // Correct spelling in non-lemmatized words
            .Chain(Item<MorphologyAnalyzer>()); // Lemmatize once again after spellcheck
}
```

Сначала мы сделали универсальные пайплайны для задач машинного обучения.

А потом появился ML .NET и мы поняли, что идея была правильной :)

FastText

Мы форкнули официальный FastText и сделали из него MSVC библиотеку с экспортируемыми функциями.

Потом написали .NET-обёртку, чтобы можно было спокойно использовать из любого проекта.

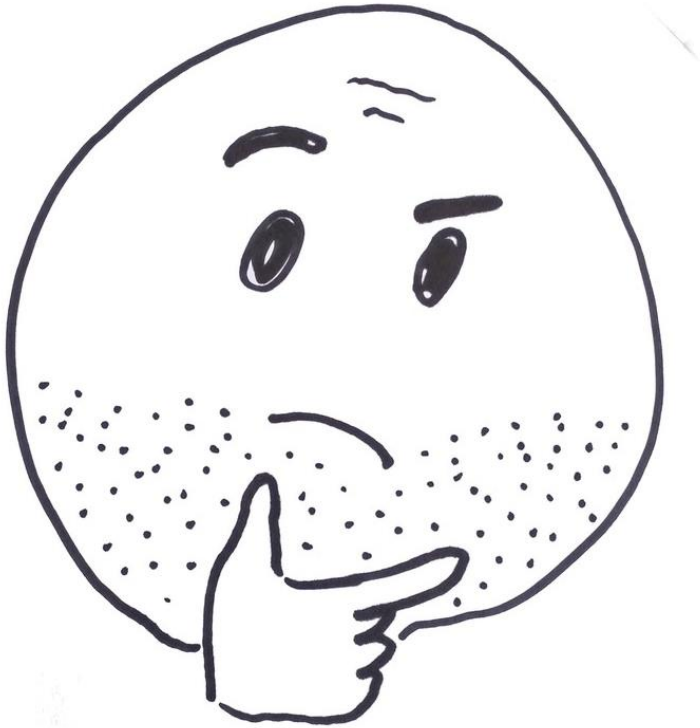
В результате, FastText — это просто ещё один элемент пайплайна, который кладёт в контекст результат классификации.

А что если обучать не только на диалогах?

Мы сделали универсальный движок для классификации любого текста.

Больше примеров — лучше результат.

Есть GUI, который позволяет супер быстро заливать и размечать примеры.



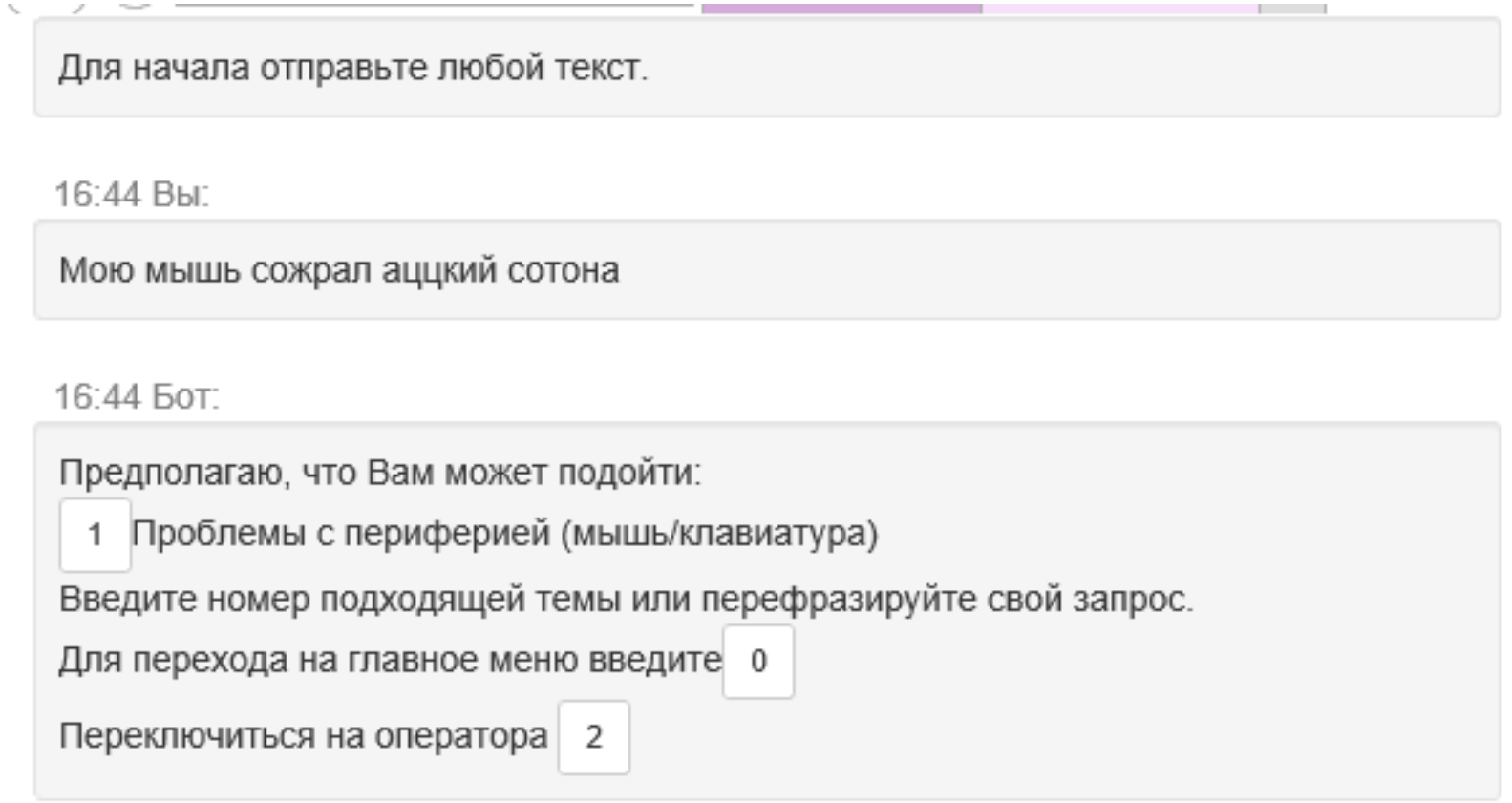
Чат-бот Единого Хелпдеска

У Единого Хелпдеска МТС уже был бот, который умел работать только с цифровым меню типа «текстовый IVR».

Мы выгрузили базу заявок Единого Хелпдеска и обучили классификатор на тексте из описания заявок.

Система обучилась достаточно хорошо, чтобы понимать проблемы пользователя в чате.

Действительно работает.

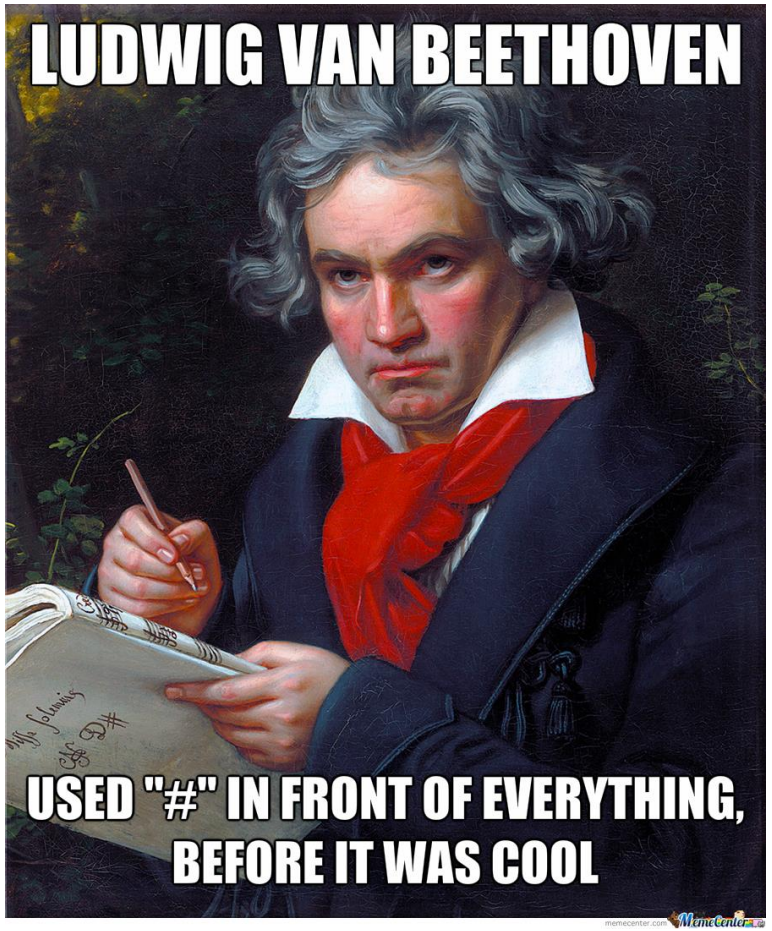


Где ещё может пригодиться NLU?

Везде, где есть текст, и вы хотите принимать решения на основе его содержания.

Классификатор может обучиться давать тексту любую характеристику, которую вы для него придумаете.

Например, можно выявлять эмоциональную окраску текста или предсказывать хэштеги по описанию фоточки в Instagram :)

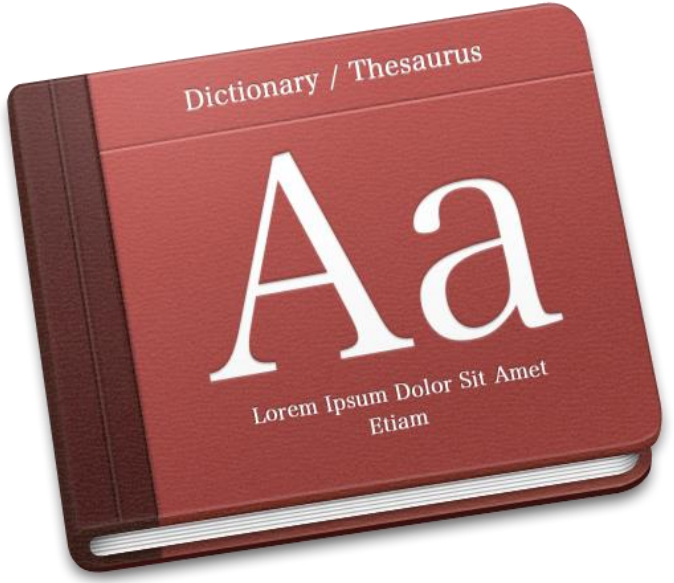


Лингвистика

Можно просто засунуть текст в FastText и получить хороший результат по классификации.

«Мы ели суп, а ели шумели» — стандартный пример, с омонимией. Надо что-то делать.

А ещё люди пишут с ошибками. Сначала мы хотели распечатывать самые смешные, но потом у нас закончилась бумага.



Grammar Engine

Решили использовать библиотеку Grammar Engine: https://github.com/Koziev/GrammarEngine. В сравнении с AOT эта библиотека показалась нам более продуманной и расширяемой.

Написали свою .NET-обёртку для словарного движка.

Разбираем сообщения от пользователя, снимаем омонимию, тегируем части речи, лемматизируем фразу. На вход классификатора подаются токены в исходной форме:

«Мы ели суп, а ели шумели» → «я есть суп а ель шуметь»

Сценарный движок чат-бота

В платформе нужна возможность динамически настраивать сложные сценарии.

На Microsoft Bot Framework «из коробки» можно написать специализированного бота. Чтобы изменить сценарий, нужно изменить код и задеплоить бота заново.

Мы написали свой сценарный движок, который работает по принципу направленного графа.

Экосистема чат-бота

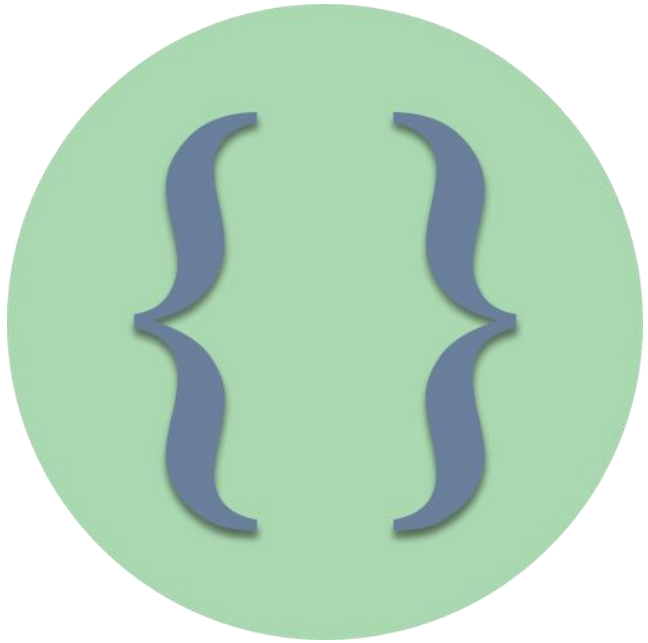
Просто «болталка» никому не нужна. Бот должен что-то делать. Возникают вопросы:

- Как бот будет интегрироваться с ИТ-ландшафтом?
- Кто будет пополнять базу сценариев, и как оценивать эффективность этих сценариев?

Даже котику надоело подгружать новые данные в эксельках.

Шаблоны и выражения

- Мы встроили Roslyn прямо в движок бота.
- Бот может выбрать нужный переход на основе любого C#-выражения.
- В любом месте сообщения можно использовать любое C#-выражение с помощью {{шаблонов}}.



Система плагинов

Кодовая миграция здорового человека:

```
PM> Add-Migration FooBar
```

Кодовая миграция с архитектурой плагинов:

```
PM> Add-Migration -ProjectName MTSBotFramework.Plugins.Foris -StartupProjectName MTSBotFramework -
ConfigurationTypeName MTSBotFramework.Plugins.Foris.BotData.Migrations.Configuration FooBar
```

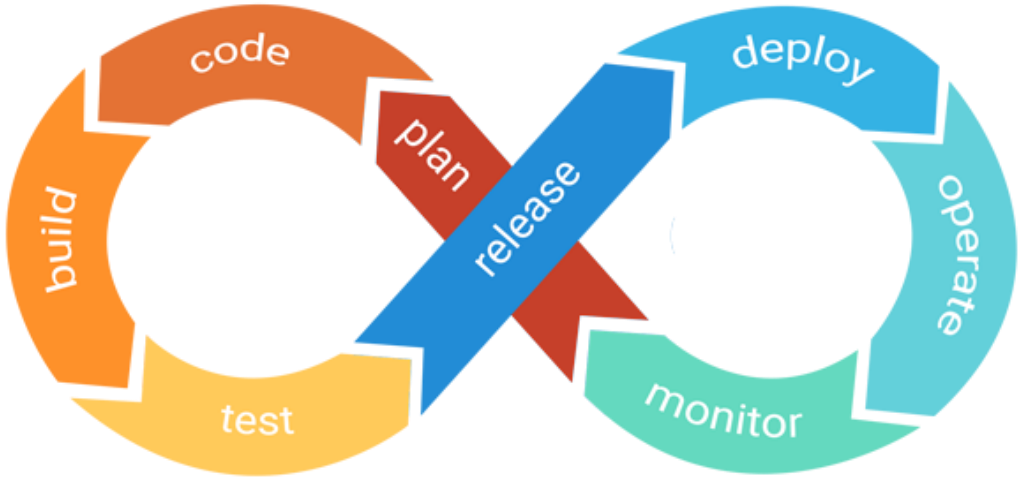
Плагин может добавлять свои классы в контекст сценария и шаблонизатор, может иметь свои таблицы в общей БД и свои контроллеры Web API.

Метрики и постоянная обратная связь

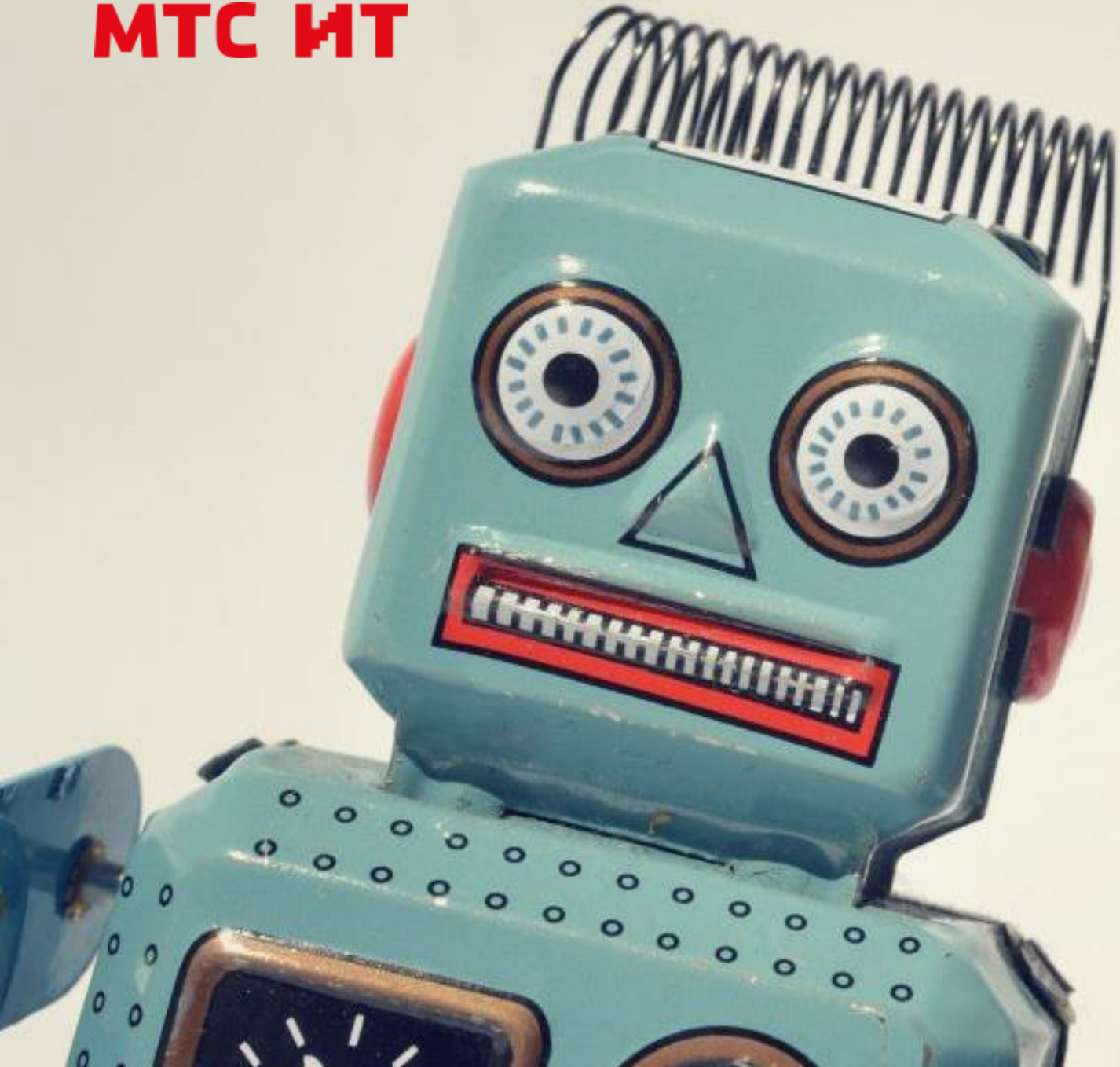
Нужно выбрать и постоянно контролировать ключевые метрики.

Такой процесс нужно использовать не только на этапе поставки кода, но и при разработке сценариев.

Мониторинга много не бывает!



МТС IT



Спасибо!

oleg.tarasov@mts.ru