effective

effectivedev_agency

КАК DART ЗАМЕНИЛ PYTHON: АНАЛИЗ ВРЕМЕННЫХ РЯДОВ НА FLUTTER



Владислав Зюзько

Flutter Engineer,
Effective

О СЕБЕ

- Приехал из Омска, сотрудник компании Effective
- 3 года коммерческой мобильной разработки на Flutter
- Веду собственный проект по разработке приложения Sweet Limit



Владислав Зюзько

Flutter Engineer в Effective

effective

План доклада

#1 Задача: предсказать опасный уровень сахара в крови

#2 Почему не Python на сервере

#3 Статистика на устройстве

#4 Нативное ядро и пакет-прослойка

#5 Архитектура: от C до Dart

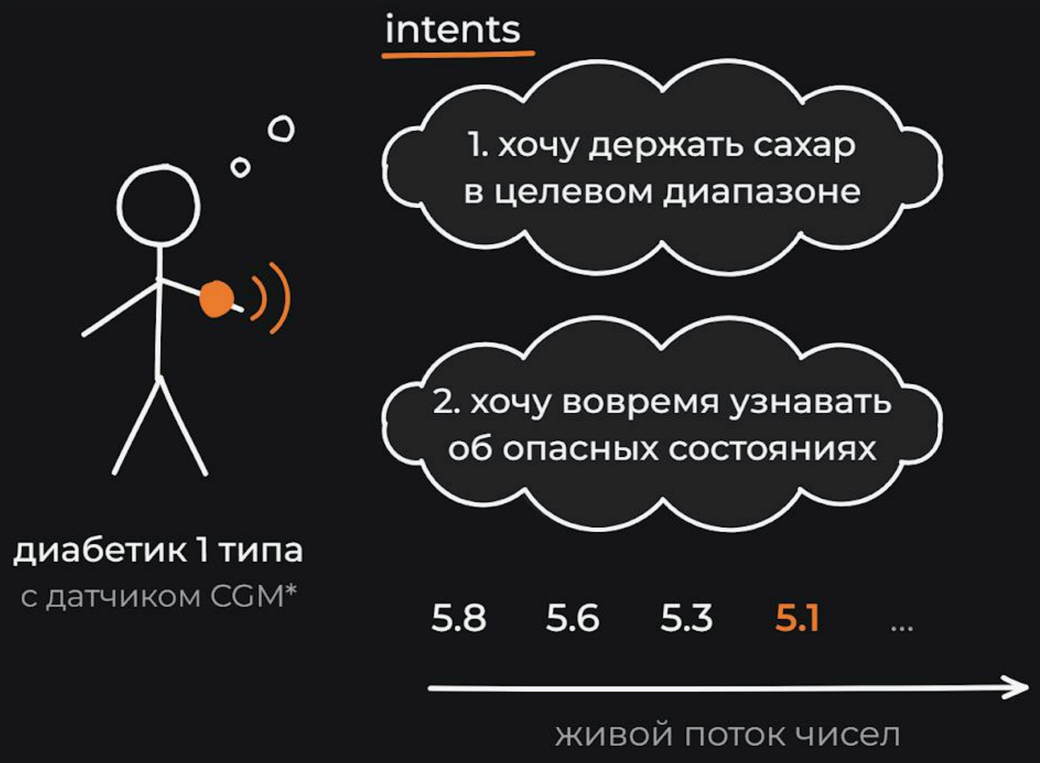
#6 Метрики, гипотезы, выводы

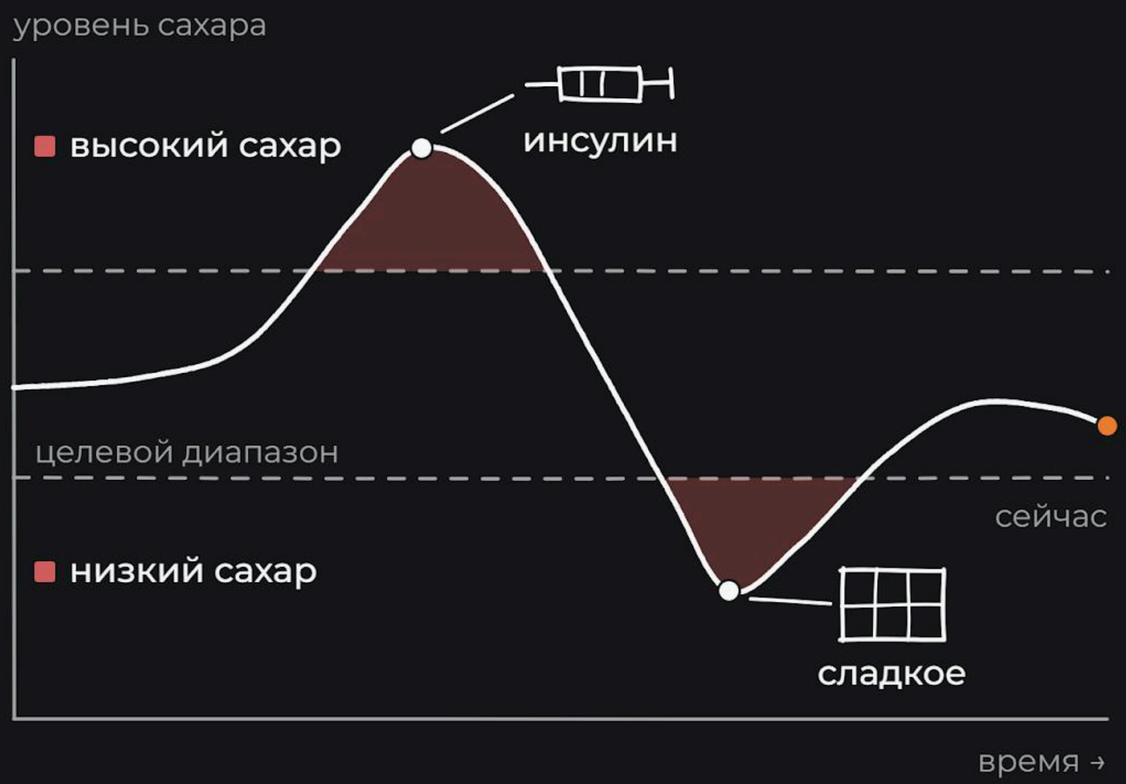
01

ЗАДАЧА: ПРЕДСКАЗАТЬ ОПАСНЫЙ УРОВЕНЬ САХАРА В КРОВИ

Система раннего предупреждения

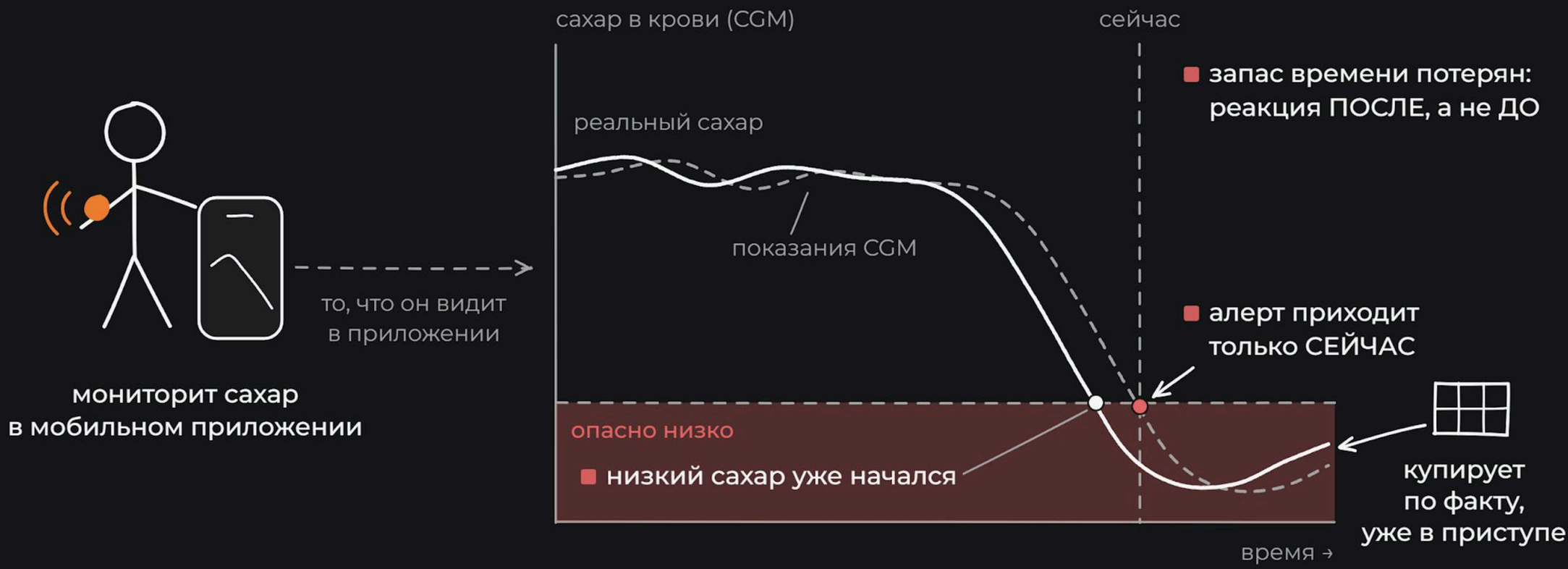
ЖИЗНЬ С ДИАБЕТОМ ПЕРВОГО ТИПА



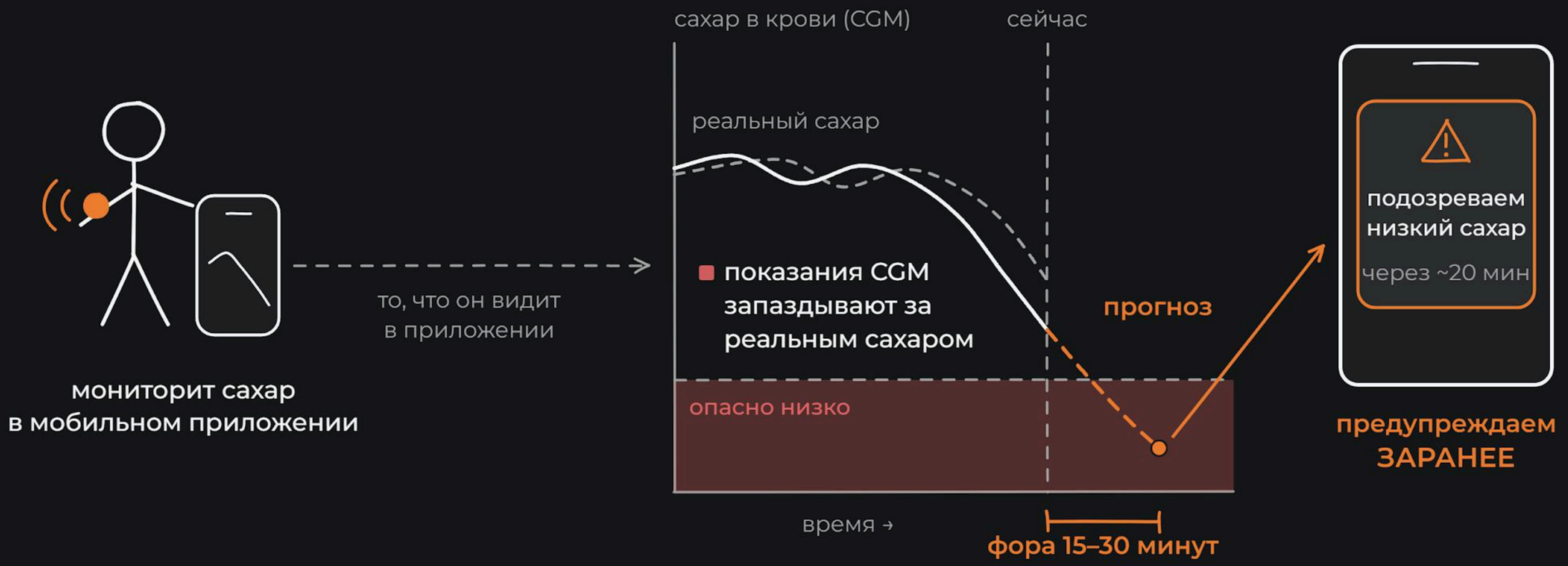


! CGM — Continuous Glucose Monitoring, непрерывный мониторинг глюкозы

ПРОБЛЕМА ФАКТИЧЕСКИХ ТРЕВОГ



РЕШЕНИЕ — ДАЁМ ПРОГНОЗ НА N МИНУТ

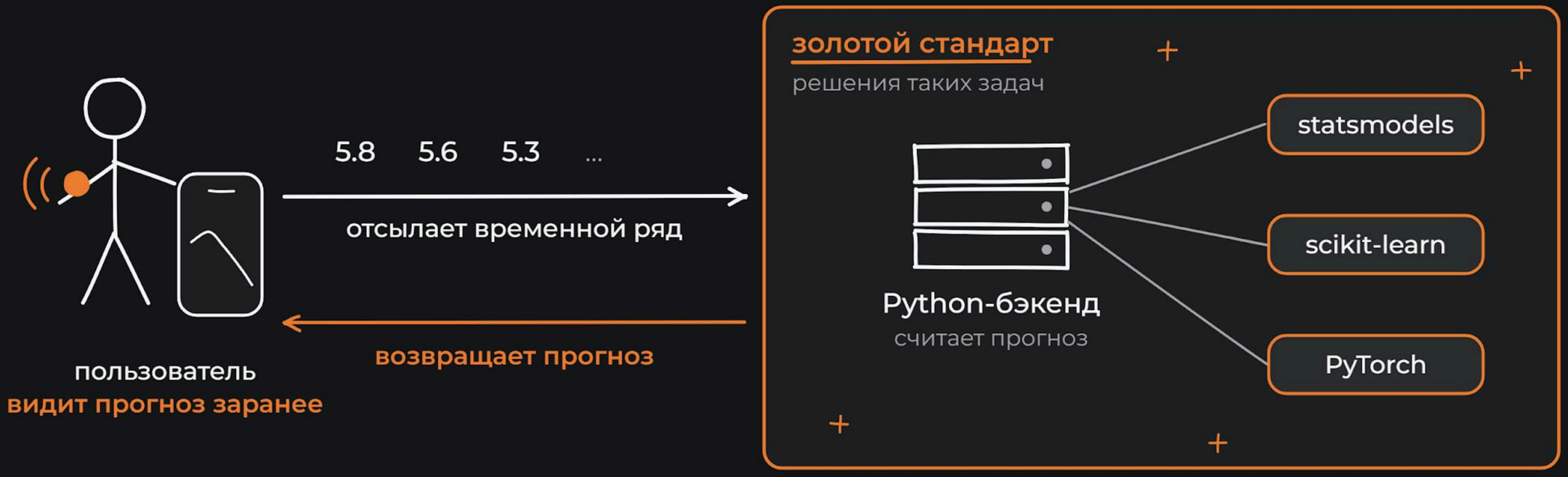


02

ПОЧЕМУ НЕ PYTHON НА СЕРВЕРЕ

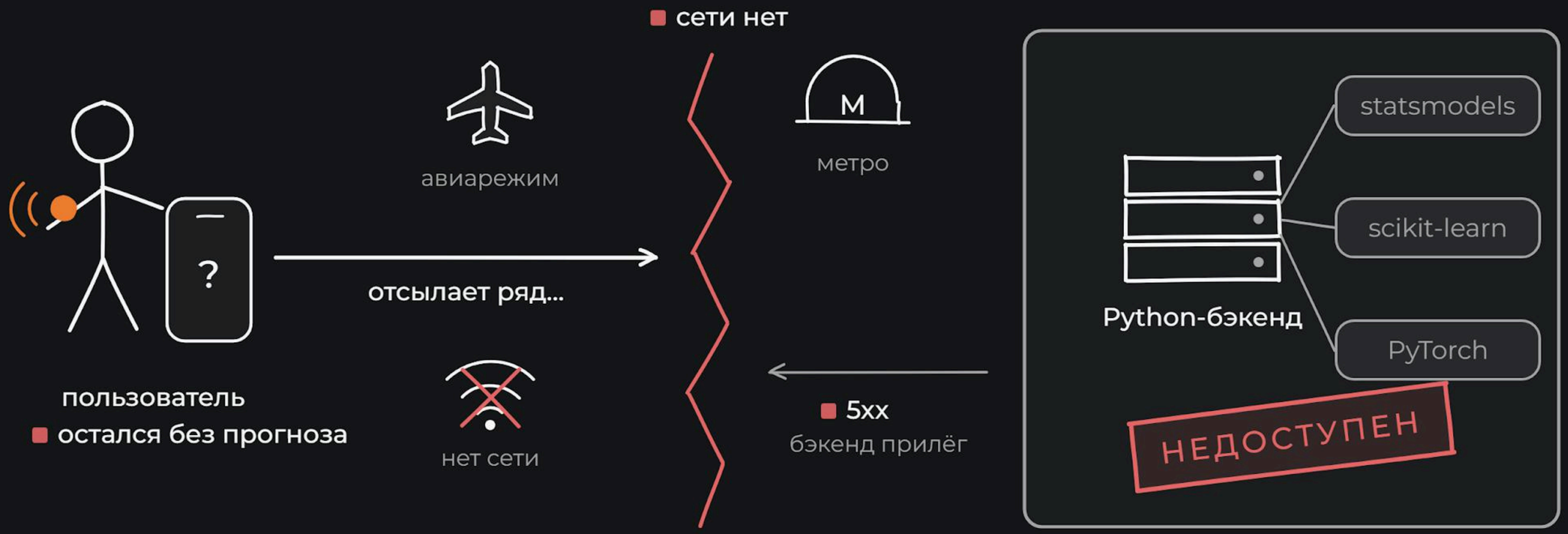
Обоснование Edge ML кейса

КАК ТАКИЕ ЗАДАЧИ РЕШАЮТСЯ СЕЙЧАС



классическая
клиент-серверная схема

СЕТЬ КАК НЕСТАБИЛЬНЫЙ КОМПОНЕНТ



РЕШЕНИЕ — ПРИ НЕСТАБИЛЬНОЙ СЕТИ РЯД ПРОГНОЗИРУЕТСЯ УСТРОЙСТВОМ



03

РЕШЕНИЕ: СТАТИСТИКА НА УСТРОЙСТВЕ

Весь прогноз – офлайн в Flutter приложении

ПОЧЕМУ PYTHON НЕ ДРУЖИТ С FLUTTER

Flutter-приложение

Dart, AOT-компиляция

официального моста нет

только сторонние обвязки — без гарантий

Python

- CPython-рантайм
- вся stdlib
- NumPy / SciPy (нативные)

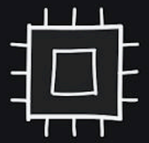
весь этот багаж едет в APK



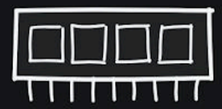
+десятки МБ



медленная граница



ест CPU



и память пользователя

ТРИ УРОВНЯ СЛОЖНОСТИ В ПРОГНОЗИРОВАНИИ ВРЕМЕННОГО РЯДА



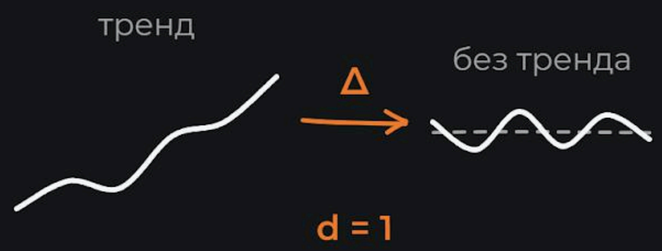
ARIMA (P, D, Q): ИЗ ЧЕГО СОСТОИТ

AR
авторегрессия



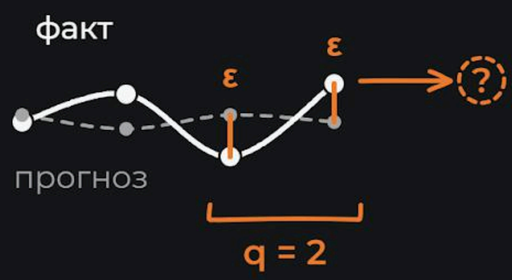
p — сколько прошлых значений берём

I
интегрирование



d — сколько раз берём разность

MA
скользящее среднее



q — сколько прошлых ошибок берём

TIME SERIES В DART

СКОЛЬЗЯЩИЕ СРЕДНИЕ, EWMA

Пишутся за вечер.
Детерминированно

ARIMA НА PUB.DEV

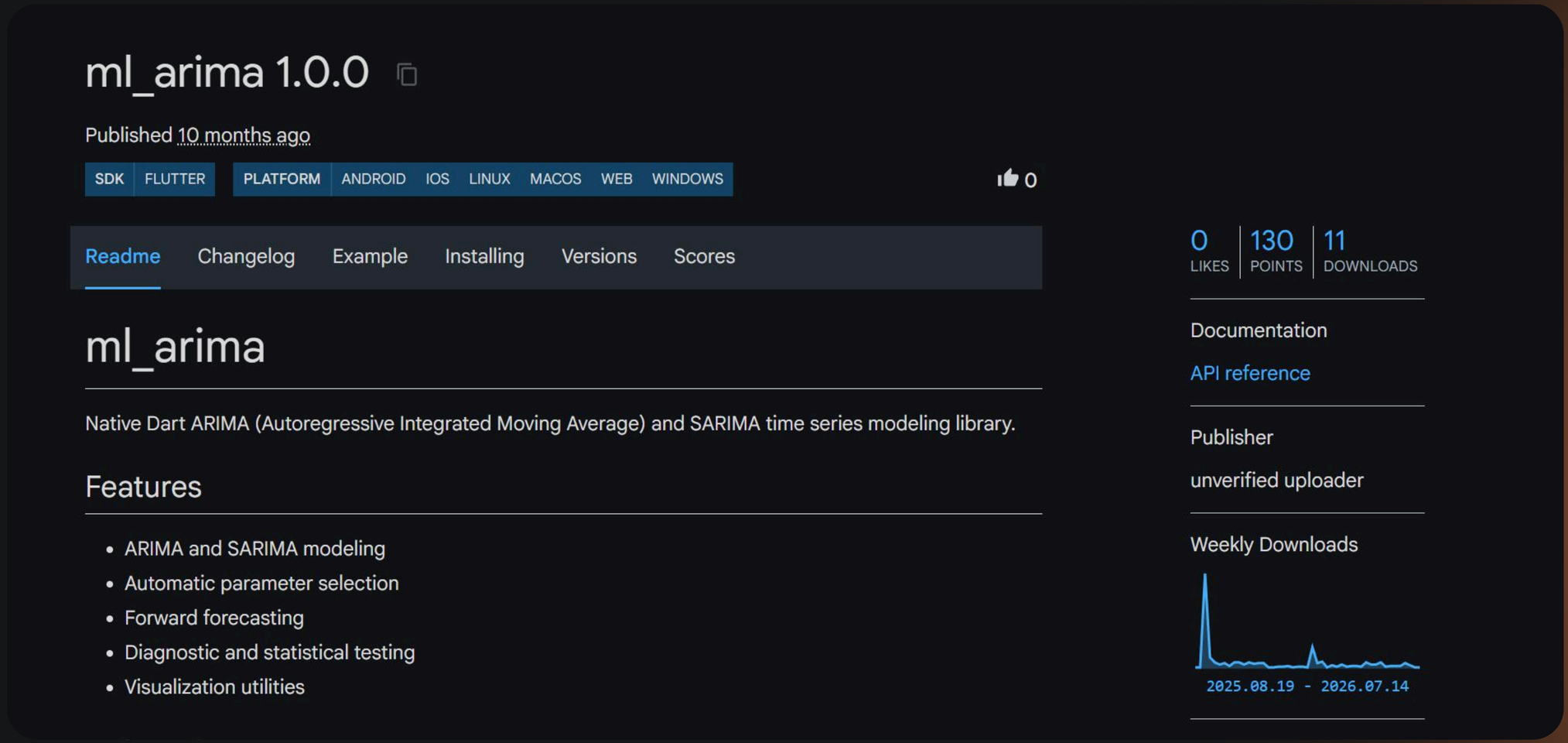
Почти ничего

ML_ARIMA

Есть, но без валидации
и продакшена

! Проверенной временем ARIMA на чистом Dart нет.

TIME SERIES B DART



04

НАТИВНОЕ ЯДРО И ПАКЕТ-ПРОСЛОЙКА (CTSA И TSERIES)

CTSA: ARIMA НА ЧИСТОМ ANSI C

МОДЕЛИ

ARIMA, SARIMA, SARIMAX, Auto-ARIMA

ЗАВИСИ-МОСТИ

0

37 .c + 37 .h, без Fortran

МАТЕМАТИКА

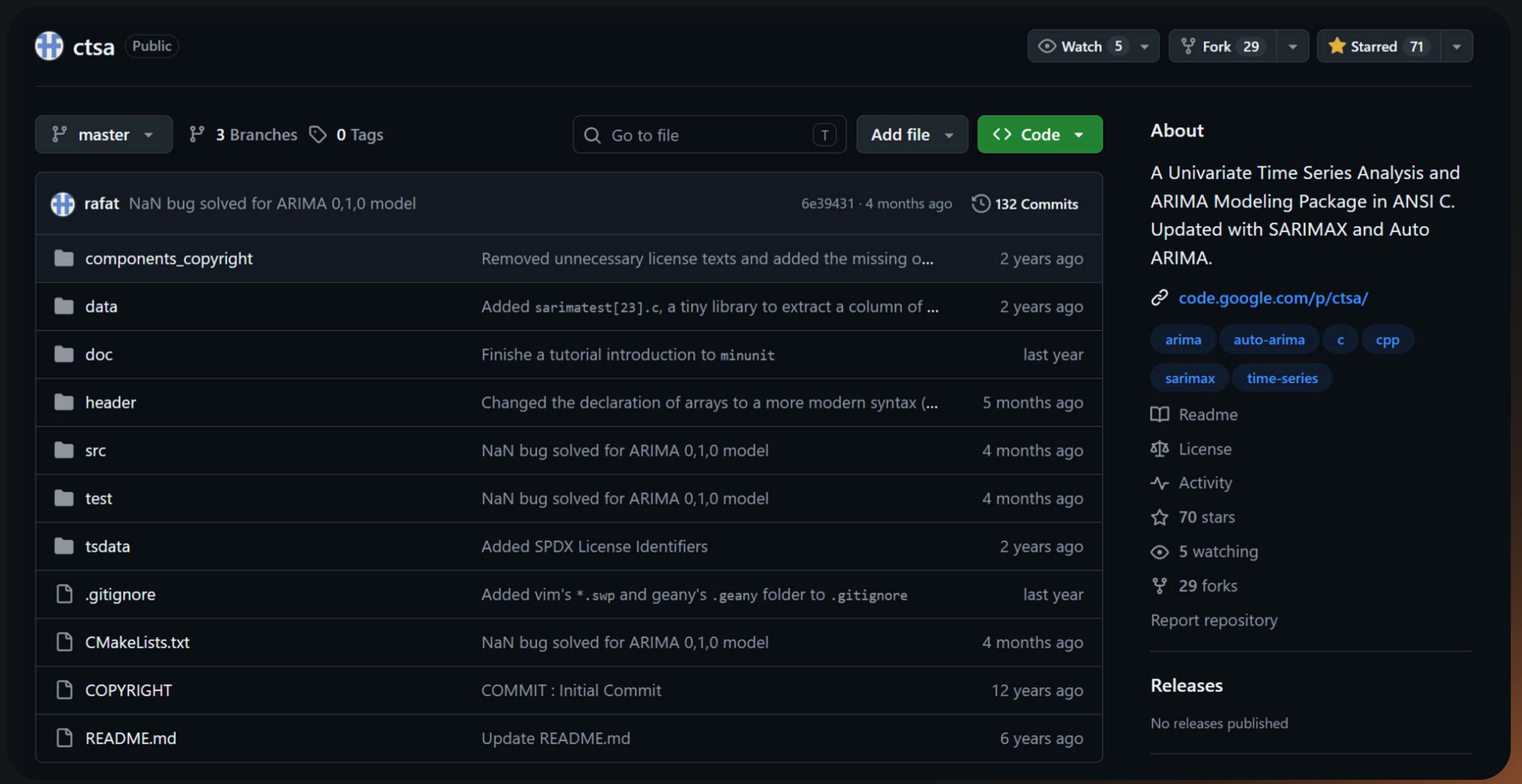
Своя линейная алгебра и оптимизатор

СБОРКА

Тот же clang из NDK. Чужого рантайма нет

Автор — Rafat Hussain (wavelib, 443★ на GitHub)

CTSA



ctsa Public

Watch 5 | Fork 29 | Starred 71

master | 3 Branches | 0 Tags | Go to file T | Add file | Code

rafat NaN bug solved for ARIMA 0,1,0 model — 6e39431 · 4 months ago — 132 Commits

components_copyright	Removed unnecessary license texts and added the missing o...	2 years ago
data	Added `sarimatest[23].c`, a tiny library to extract a column of ...	2 years ago
doc	Finishe a tutorial introduction to `minunit`	last year
header	Changed the declaration of arrays to a more modern syntax (...	5 months ago
src	NaN bug solved for ARIMA 0,1,0 model	4 months ago
test	NaN bug solved for ARIMA 0,1,0 model	4 months ago
tsdata	Added SPDX License Identifiers	2 years ago
.gitignore	Added vim's `*.swp` and geany's `.geany` folder to `.gitignore`	last year
CMakeLists.txt	NaN bug solved for ARIMA 0,1,0 model	4 months ago
COPYRIGHT	COMMIT : Initial Commit	12 years ago
README.md	Update README.md	6 years ago

About

A Univariate Time Series Analysis and ARIMA Modeling Package in ANSI C. Updated with SARIMAX and Auto ARIMA.

code.google.com/p/ctsa/

arima | auto-arima | c | cpp | sarimax | time-series

Readme
License
Activity
70 stars
5 watching
29 forks
Report repository

Releases

No releases published

CTSA: ДОВЕРЯЙ, НО ПРОВЕРЯЙ

РИСК	СТРАХОВКА
Открытые issue про корректность	Пин проверенного коммита
Автор молчит годами	Свой golden-value harness
Тесты — демо без ассертов	Сверка с R на AirPassengers

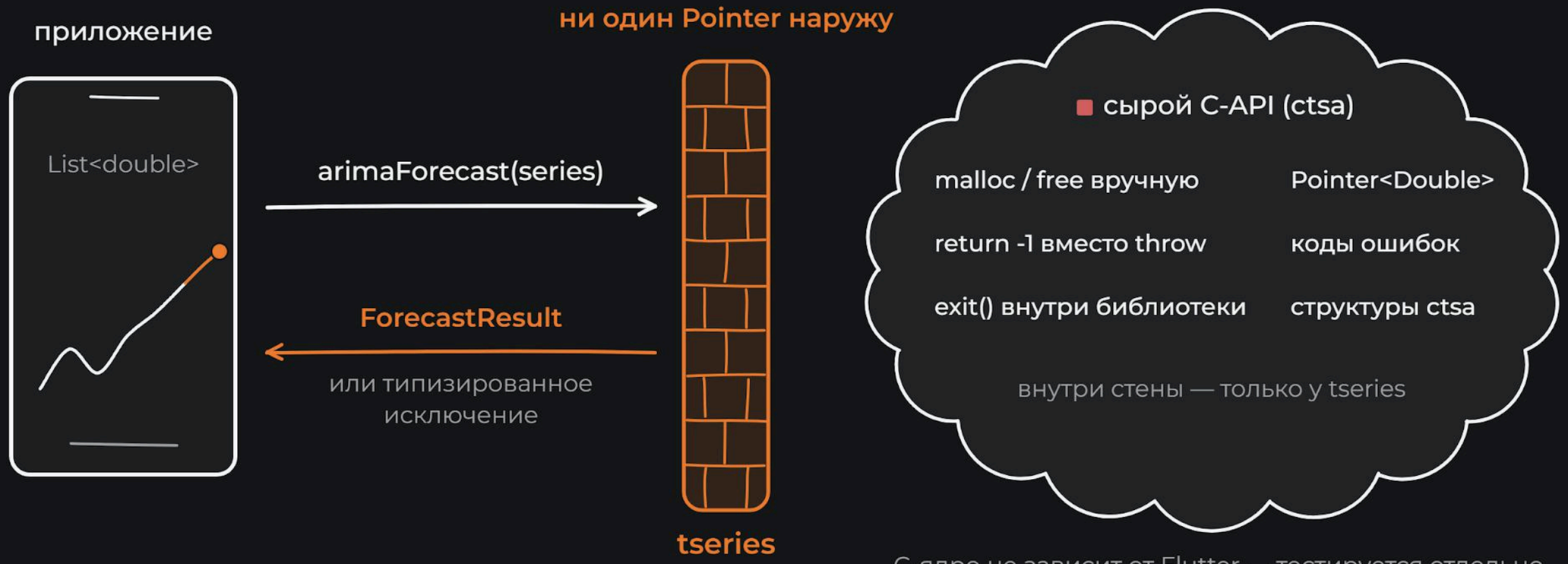
<1%

расхождение коэффициентов с R

4+ знака

совпадают σ^2 и loglik

ПАКЕТ TSERIES: ЧТОБЫ ПРИЛОЖЕНИЕ НЕ ЗНАЛО ПРО УКАЗАТЕЛИ



C-ядро не зависит от Flutter — тестируется отдельно

ВЕСЬ ПУБЛИЧНЫЙ API — ОДНА СТРОКА

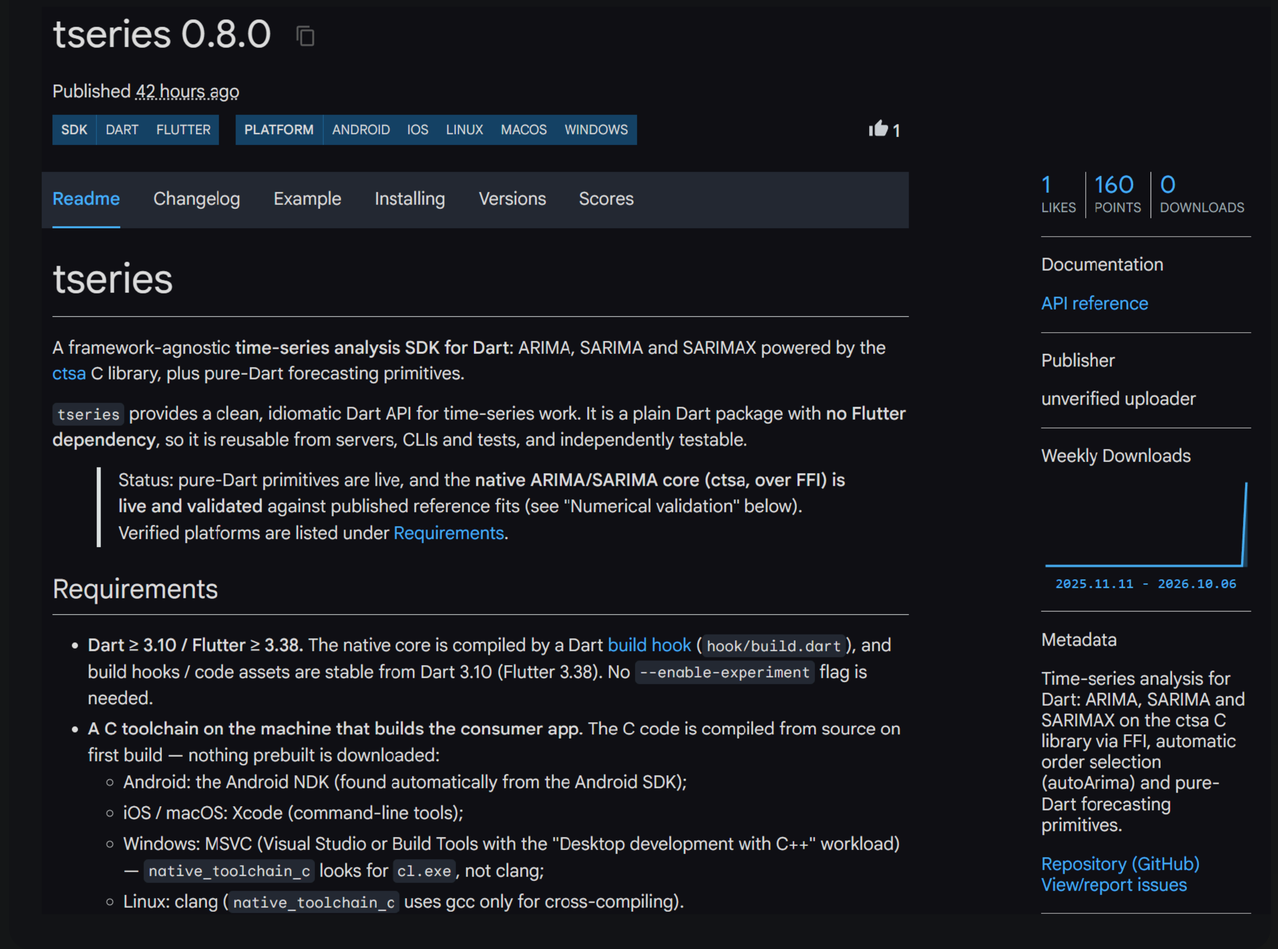
Одна строка обычного Dart

api.dart

```dart
// List<double> на входе — прогноз на выходе. Ни Pointer, ни malloc, ни кодов возврата.
final forecast = arimaForecast(series, orders: ladder, horizon: 6);
//                              ▲ история          ▲ на 30 минут вперёд
```

МЫ НА PUB.DEV!

effectivedev_agency



tseries 0.8.0

Published 42 hours ago

SDK DART FLUTTER | PLATFORM ANDROID IOS LINUX MACOS WINDOWS | 1

Readme Changelog Example Installing Versions Scores

tseries

A framework-agnostic **time-series analysis SDK for Dart**: ARIMA, SARIMA and SARIMAX powered by the ctsa C library, plus pure-Dart forecasting primitives.

`tseries` provides a clean, idiomatic Dart API for time-series work. It is a plain Dart package with **no Flutter dependency**, so it is reusable from servers, CLIs and tests, and independently testable.

> Status: pure-Dart primitives are live, and the **native ARIMA/SARIMA core (ctsa, over FFI) is live and validated** against published reference fits (see "Numerical validation" below). Verified platforms are listed under Requirements.

Requirements

- **Dart ≥ 3.10 / Flutter ≥ 3.38.** The native core is compiled by a Dart build hook (`hook/build.dart`), and build hooks / code assets are stable from Dart 3.10 (Flutter 3.38). No `--enable-experiment` flag is needed.
- **A C toolchain on the machine that builds the consumer app.** The C code is compiled from source on first build — nothing prebuilt is downloaded:
  - Android: the Android NDK (found automatically from the Android SDK);
  - iOS / macOS: Xcode (command-line tools);
  - Windows: MSVC (Visual Studio or Build Tools with the "Desktop development with C++" workload) — `native_toolchain_c` looks for `cl.exe`, not clang;
  - Linux: clang (`native_toolchain_c` uses gcc only for cross-compiling).

1 LIKES | 160 POINTS | 0 DOWNLOADS

Documentation

API reference

Publisher

unverified uploader

Weekly Downloads

2025.11.11 - 2026.10.06

Metadata

Time-series analysis for Dart: ARIMA, SARIMA and SARIMAX on the ctsa C library via FFI, automatic order selection (autoArima) and pure-Dart forecasting primitives.

Repository (GitHub)
View/report issues

ССЫЛКА

GitHub

ССЫЛКА

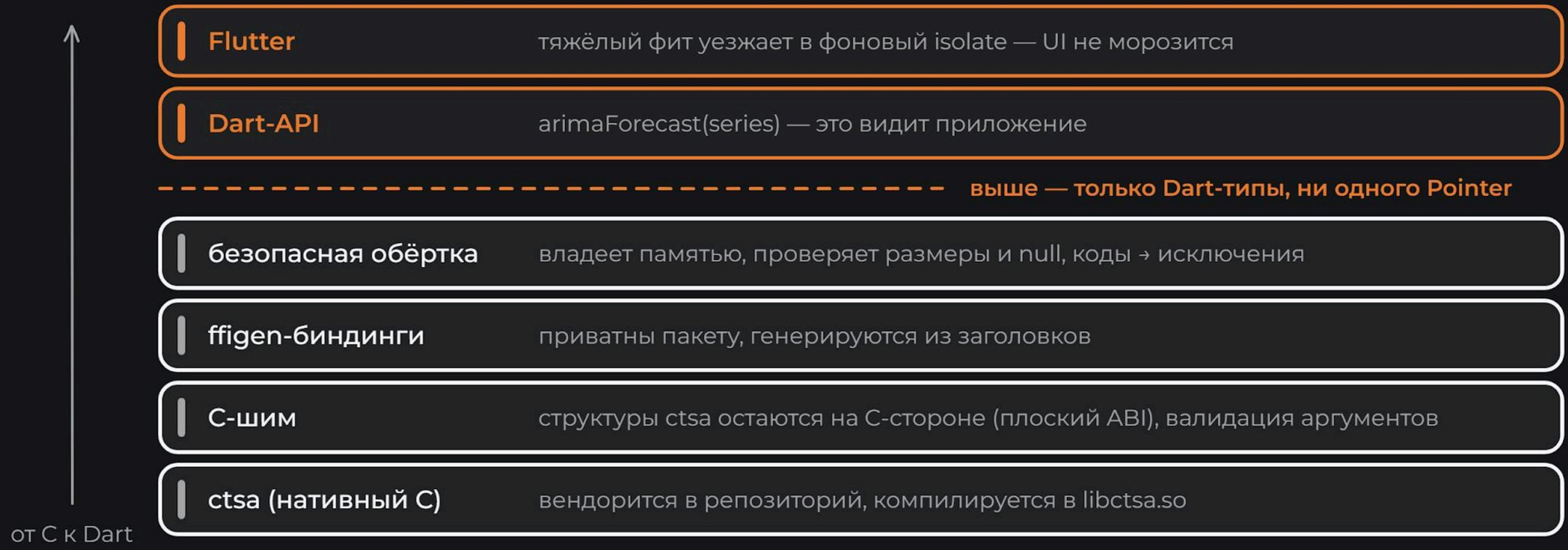
pub.dev

05

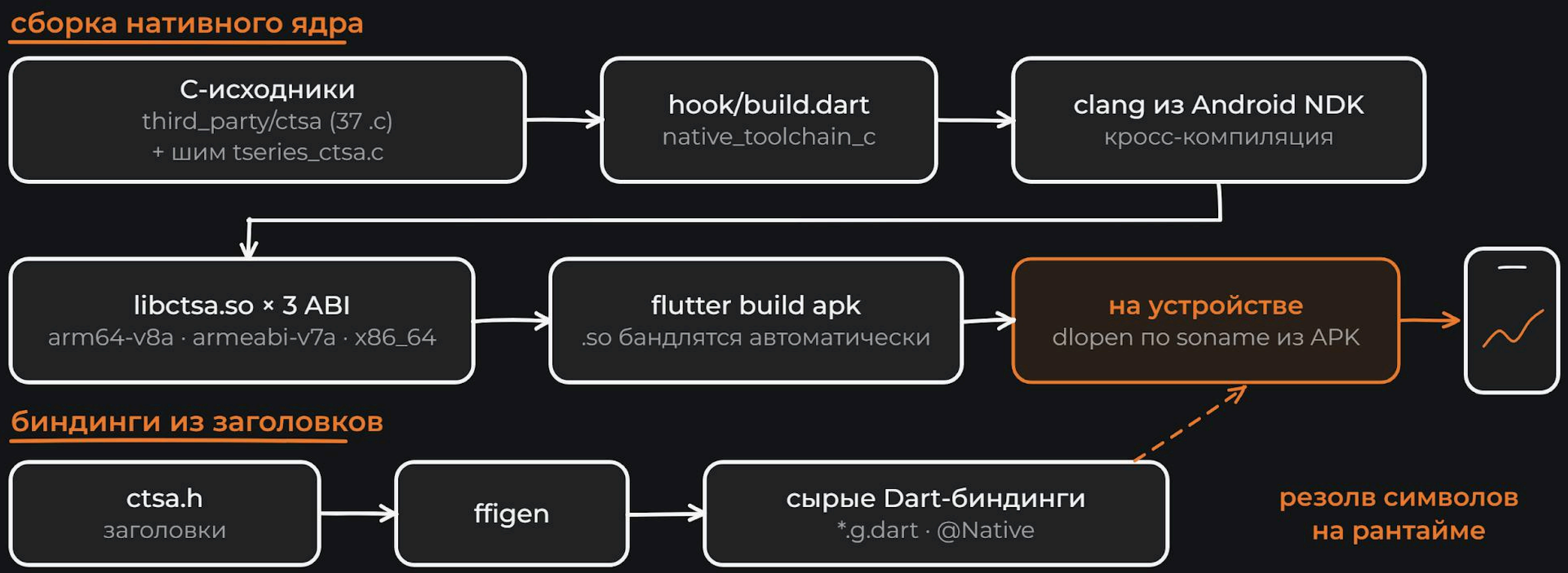
АРХИТЕКТУРА: ОТ C ДО DART

Как series доходит до C и возвращается прогнозом

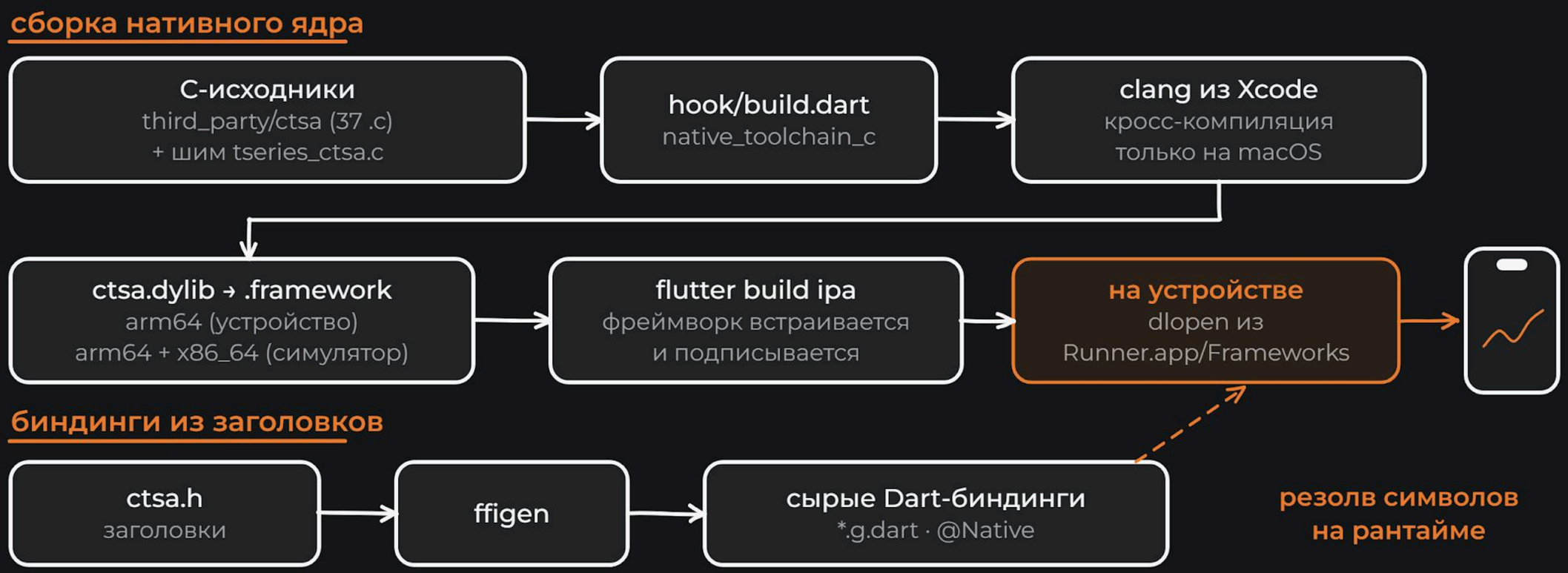
ОТ C ДО ЧИСТОГО DART: СЛОИ TSERIES



СБОРКА TSERIES ПОД ANDROID



СБОРКА TSERIES ПОД IOS



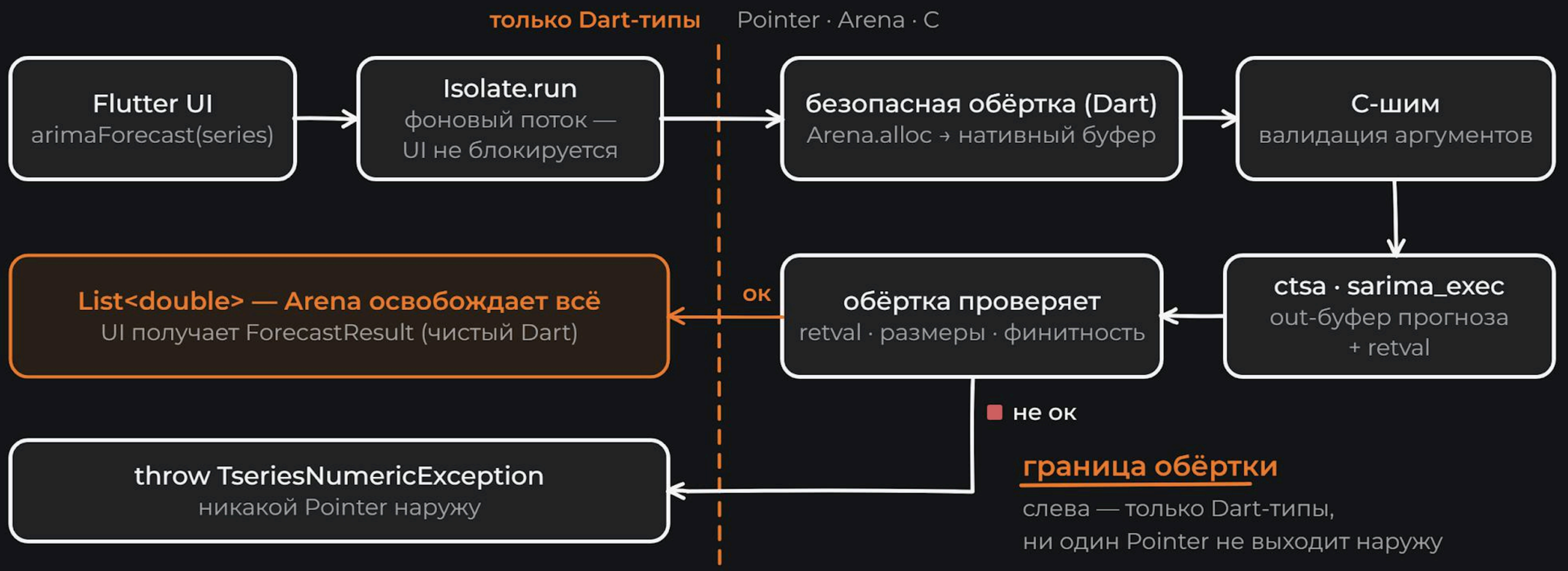
ХУК БИЛДА

Кросс-компиляция C под три ABI и упаковка в APK — несколько строк декларативного Dart.

hook_build.dart

```dart
// hook/build.dart
final builder = CBuilder.library(
  name: 'ctsa',
  assetName: 'ctsa',                    // → package:tseries/ctsa
  sources: ['native/tseries_ctsa.c', ...?_ctsaSources],  // 37 файлов ctsa
  libraries: _mathLibrariesFor(os),      // ← эта строка стоила нам белого экрана
  std: 'c11',
);
await builder.run(input: input, output: output);
```

КАК SERIES ДОХОДИТ ДО C И ВОЗВРАЩАЕТСЯ ПРОГНОЗОМ



ARENA: TRY/FINALLY ДЛЯ НАТИВНОЙ ПАМЯТИ

Вся нативная память живёт в Arena и чистится на любом пути, включая throw, — ни один байт не утечёт

arena.dart

```dart
return using((arena) {
  final seriesPtr   = arena<Double>(n)..asTypedList(n).setAll(0, series);
  final forecastPtr = arena<Double>(horizon);
  final status = b.tseries_sarima_fit(seriesPtr, n, /* ... порядки ... */, forecastPtr);
  if (status ≠ TSERIES_OK) _throwForStatus(status, 'sarimaFit'); // Pointer не течёт наружу
  return forecastPtr.asTypedList(horizon).toList();              // → чистый List<double>
});
```

ТЯЖЁЛЫЙ ФИТ — В ФОНОВЫЙ ISOLATE

FFI-вызов блокирует поток => делаем дорогие операции в изоляте

isolate.dart

```dart
final forecast = await Isolate.run(() ⇒ arimaForecast(series, orders: ladder, horizon: 6));
```

06

МЕТРИКИ, ГИПОТЕЗЫ, ВЫВОДЫ

Подкрепляем теорию практикой

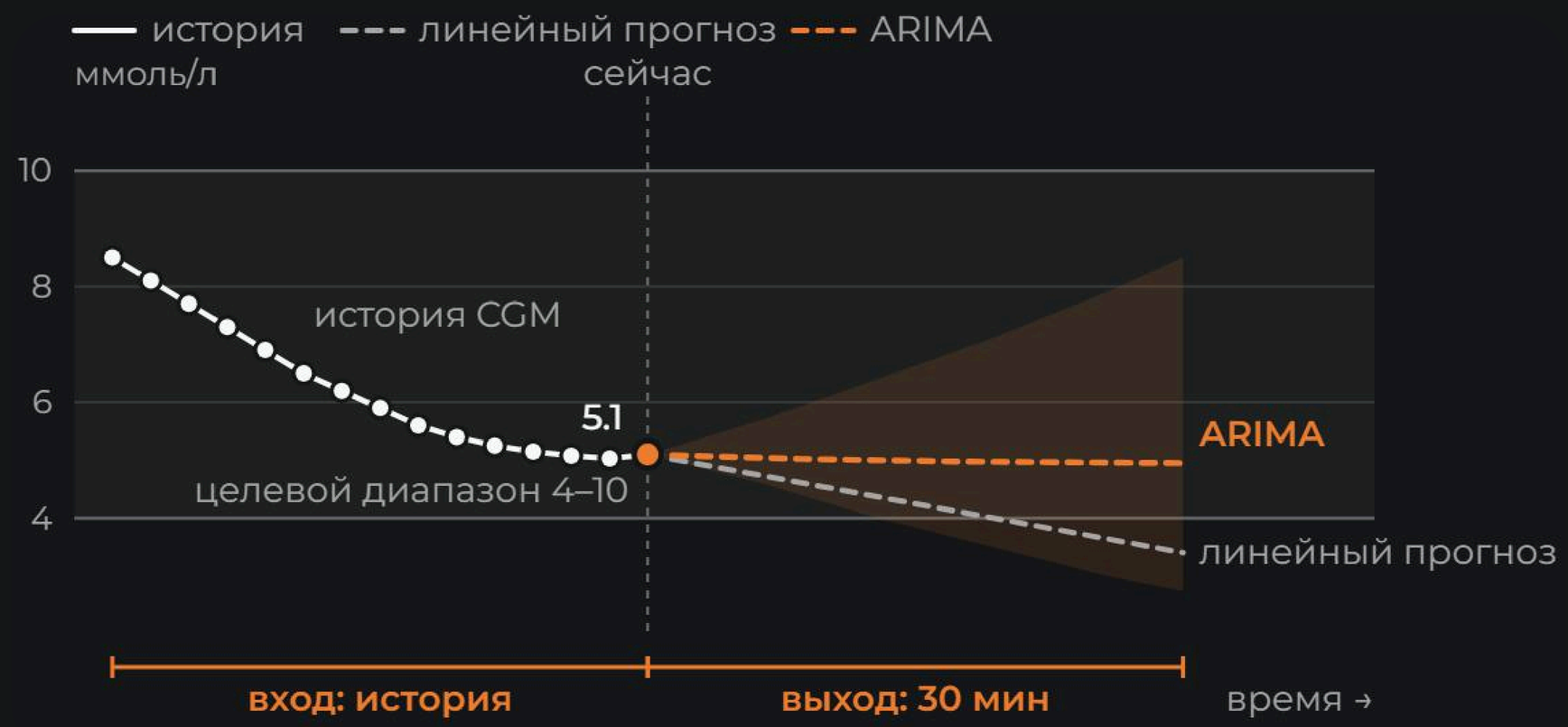
ВХОД: ЧАСЫ.
ВЫХОД: 15–30 МИНУТ ВПЕРЁД

ВХОД

Короткая история:
несколько точек
для линейной,
12 часов для ARIMA

ВЫХОД

Несколько точек
на 15–30 минут
вперёд



ЛЕСТНИЦА МОДЕЛЕЙ: ОТ ЛЁГКОГО К ТЯЖЁЛОМУ

Persistence
«через 30 минут — как сейчас»
baseline для skill

Линейная экстраполяция
прямая по N точкам
(+ свип окна N)
так работают помпы PLGS

Демпфированный Хольт
сглаживание с затухающим трендом
сетка 72 конфигурации

ARIMA
автокорреляция приращений
ядро ctsa через tseries



легче → тяжелее, точнее

SKILL: МОДЕЛЬ ПРОТИВ «ЗАВТРА КАК СЕГОДНЯ»

skill = 1 – RMSE модели / RMSE persistence

0

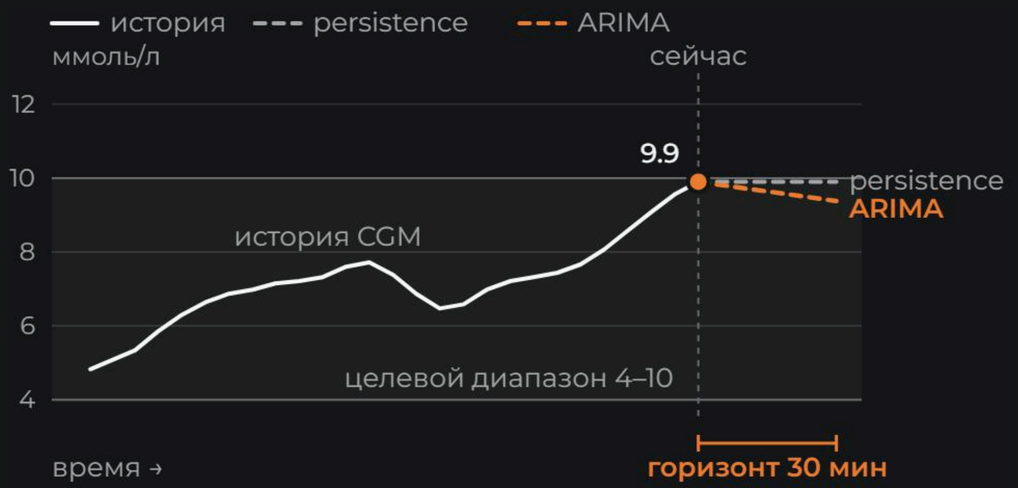
не лучше persistence

> 0

добавляет информацию

< 0

вредит

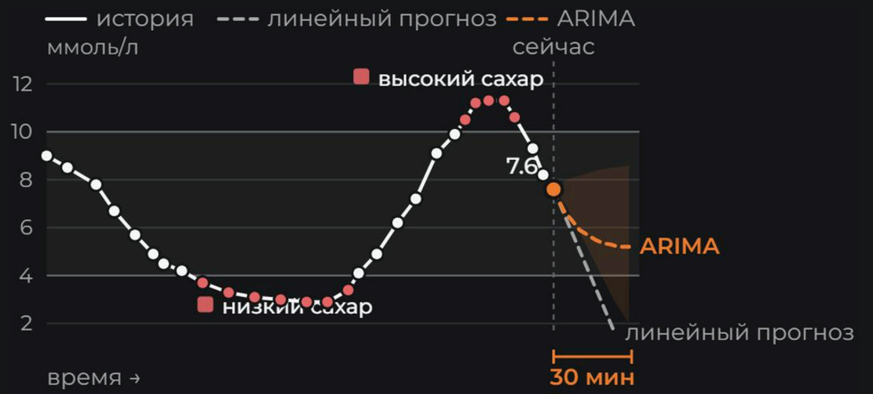


! Низкий RMSE сам по себе ничего не значит: за 5 минут сахар почти не меняется

432 КОНФИГУРАЦИИ ХОЛЬТА: SKILL ОКОЛО НУЛЯ

72 набора параметров × 6 горизонтов. Лучший Хольт: +0.05

ГОРИЗОНТ	МОДЕЛЬ	RMSE, ммоль/л	SKILL
15 мин	Persistence	0.66	—
	Лучший Holt	0.62	+0.05
	ARIMA(2,1,1)	0.48	+0.27
30 мин	Persistence	1.16	—
	Лучший Holt	1.12	+0.04
	ARIMA(2,1,1)	1.03	+0.11



! 1366 origins, один набор точек.

Три модели, 15 и 30 минут

МЕТРИКА	PERSISTENCE	ЛИНЕЙНЫЙ	ARIMA
RMSE 15м	0.66	0.65	0.48
MAE 15м	0.45	0.46	0.33
MARD 15м	7.4 %	8.0 %	5.7 %
Skill 15м	—	0.01	0.27
RMSE 30м	1.16	1.34	1.03
MAE 30м	0.79	0.96	0.70
MARD 30м	13.0 %	16.5 %	11.8 %
Skill 30м	—	-0.16	0.11

ARIMA fit-failure rate: 0.0 %

ПОДРОБНЕЕ ПРО МЕТРИКИ

Хольт: ТОП-5 из 72 по skill за 15 минут

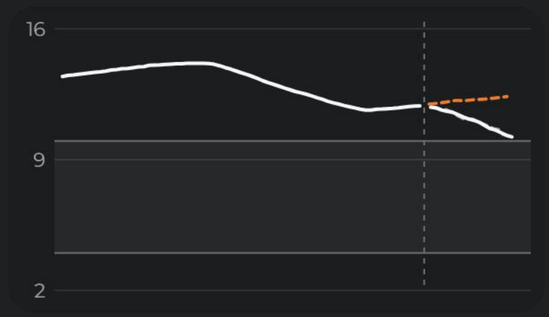
ПАРАМЕТРЫ	RMSE 15М	SKILL 15М	SKILL 30М
α 0.7 · β 0.2 · …	0.62	0.05	0.04
α 0.7 · β 0.2 · …	0.62	0.05	0.02
α 0.7 · β 0.2 · …	0.64	0.02	0.02
α 0.7 · β 0.2 · …	0.64	0.02	-0.03
α 0.7 · β 0.2 · …	0.66	-0.01	-0.01

Линейный: (N = 4 — текущий дефолт)

ОКНО	RMSE 15М	SKILL 15М	SKILL 30М
N = 4	0.65	0.01	-0.16
N = 6	0.74	-0.13	-0.23
N = 9	0.87	-0.33	-0.35
N = 12	1.01	-0.54	-0.47
N = 18	1.26	-0.91	-0.63

БОЛЬ, С КОТОРОЙ СТОЛКНУЛИСЬ

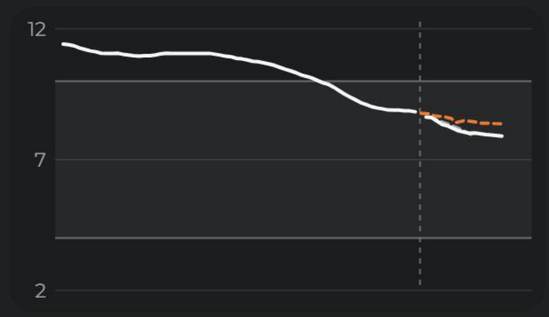
— факт (CGM) - - - ARIMA



23:41, 7 окт. Худший

+2.38 ммоль/л

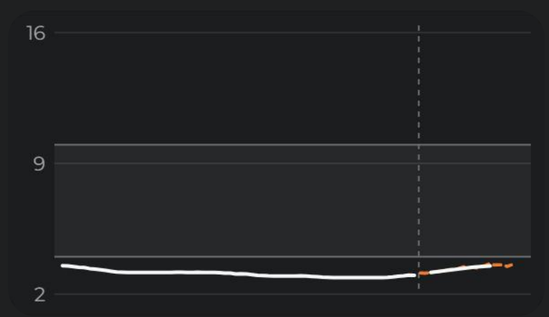
p100 · ARIMA(2,1,1) · лин.: 1.70



19:56, 7 окт. Типичный

+0.42 ммоль/л

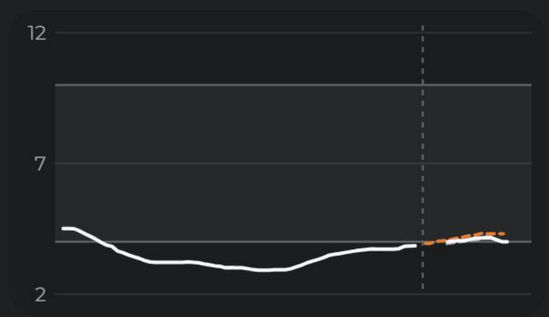
p48 · ARIMA(2,1,1) · лин.: 0.42



03:56, 6 окт. Лучший

+0.00 ммоль/л

p0 · ARIMA(2,1,1) · лин.: 0.46 · низкий сахар



04:36, 6 окт. Типичный

+0.44 ммоль/л

p49 · ARIMA(2,1,1) · лин.: 0.42

ЧТО СОШЛОСЬ

ТОЧНОСТЬ

skill > 0
на всех используемых горизонтах у ARIMA

ЗНАЧИМОСТЬ

Интервал skill
не накрывает ноль

ВЫБОР МОДЕЛИ

ARIMA дала
лучшие метрики

ПОРОГ ЗАРАНЕЕ

До эксперимента.
Не подкручивали

ЧТО ОПРОВЕРГЛОСЬ

ГИПОТЕЗА

- Чем шире окно линейной, тем точнее
- Линейная где-то обгонит persistence
- На 30 минутах всё неразличимо
- Хольт в нуле — одномерное бессильно

НА ДЕЛЕ

- RMSE растёт с окном. Лучший N = 4
- Persistence чаще всего лучше
- ARIMA даёт положительные метрики
- ARIMA смогла

ЗОНЫ РОСТА

N=1

Один человек, один сенсор,
больше месяца наблюдений

УЧЁТ ВНЕШНИХ СОБЫТИЙ

Скачки зачастую связаны
с внешними событиями

НЕЙРОСЕТЕВОЙ ПОДХОД

Локальные RNN / LSTM сети

ВЕРДИКТ: ИСПОЛЬЗОВАТЬ ARIMA ИМЕЕТ СМЫСЛ

+0.31

skill на 15 минутах:
ошибка меньше
на треть

+0.11

skill на 30 минутах:
не навредили,
и есть смысл смотреть

ПРЕДЕЛ

Модель использует данные
только с одного ряда

ДАЛЬШЕ

Использовать инсулин
и углеводы как входные
данные, переход к ARIMAX

ЗАДАВАЙТЕ ВОПРОСЫ!

ССЫЛКА

GitHub

ССЫЛКА

Telegram

Владислав Зюзько

Flutter Engineer в Effective