dataminded

# Sequence data at warp speed

with Apache Ignite

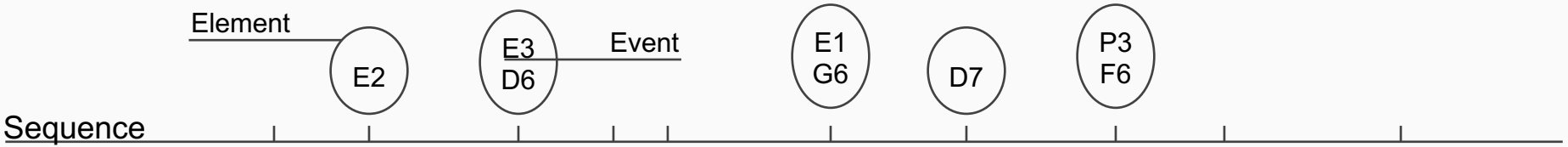
# Agenda

- Sequential Data
- How we got started...
- Introducing Apache Ignite
- Demo
- About Data Minded

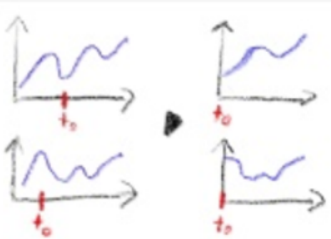
# Sequential Data

# Examples of sequence data

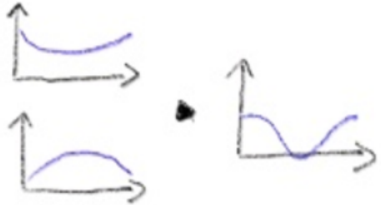
| Sequence database | Sequence | Element (Transaction) | Event (Item) |
|---|---|---|---|
| Customer | Purchase history of a given customer | A set of bought items by a customer at time *t* | Books, diary products, CDs, etc |
| Web Data | Browsing activity of a particular visitor | Ads viewed by the visitor after a single mouse click | Home page, index page, contact info, etc |
| Event Data | History of events generated by a sensor | Events triggered by a sensor at time *t* | Types of alarms generated by sensors |
| Genome sequences | DNA sequence of a particular species | An element of the DNA sequence | Bases A, T, G, C |



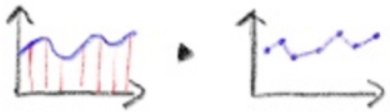
# Typical operations on sequential data



align

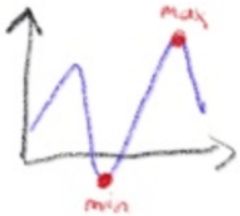


diff

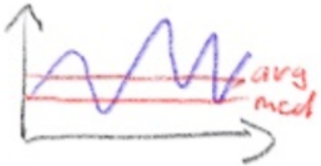


downsampling

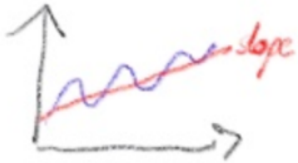
outlier



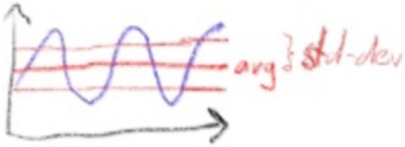

min/max




avg/med




slope




std-dev

# Goal

Interactive querying on sequence data at scale

The whole industry is focused on Spark. So we obviously got started with that...

# The first attempt on using Spark for sequence processing

Apache Spark is typically going to load the data like this: ⟹

| Timestamp | Key | Value |
|---|---|---|
| 2016-10-10 | A | 2.0 |
| 2016-10-11 | A | 3.0 |
| 2016-10-10 | B | 4.5 |
| 2016-10-11 | B | 1.5 |
| 2016-10-10 | C | 6.0 |

We call this the ‘observations’ layout.

However, the observations layout is not ideal for performing analysis. To ask most questions, you need to perform a group by on your data set, either by key or by timestamp, which is cumbersome as well as computationally expensive.


When you only got a mug and all your problems are nails

# There are a few Spark-based options that try to solve this issue

**Sandy Ryza developed Spark-TS at Cloudera**

Promising but:

- Immature
- Developed by 1 person
- Sandy left Cloudera and Spark-TS is no longer a Cloudera project

**Another solution is Huohua but is only in conceptual phase.**

Good presentation online and a Github Repository with only a README

# Better instants layout

| Timestamp | A | B | C |
|---|---|---|---|
| 2016-10-10 | 2.0 | 4.5 | 6.0 |
| 2016-10-11 | 3.0 | 1.5 | NaN |

The “instants” layout is ideal for much of traditional machine learning—for example, if you want to build a supervised learning model that predicts one variable based on contemporaneous values of the others

# Best: “timeseries” layout

In the third layout, which is most central to spark-ts, each object in the RDD stores a full univariate series.

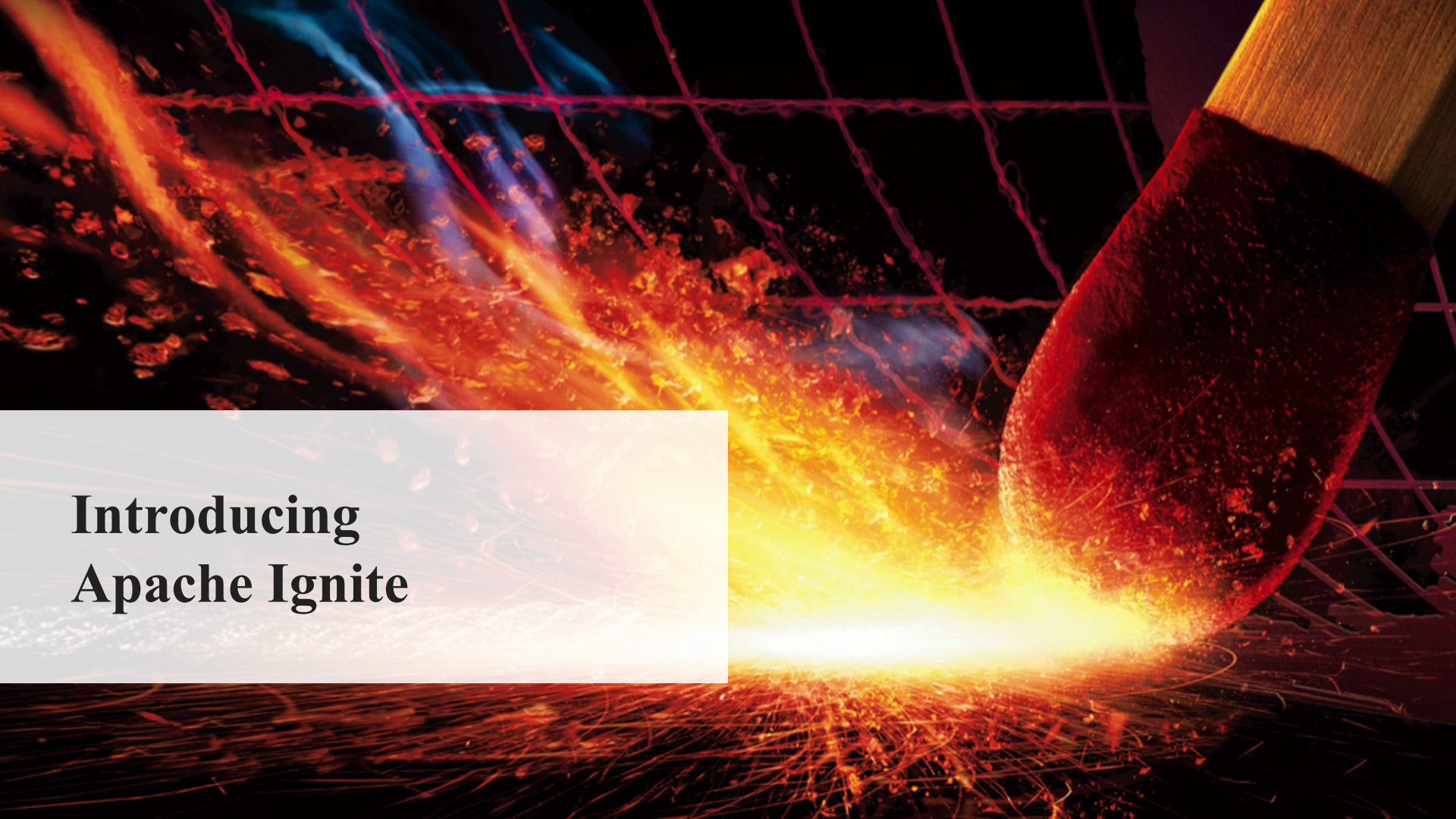
| DateTimeIndex: [2016-10-10, 2016-10-11] | |
|---|---|
| **Key** | **Series** |
| A | [2.0, 3.0] |
| B | [4.5, 1.5] |
| C | [6.0, NaN] |

# Apache Spark with a series-based approach showed difficulties in implementation

**Findings**

Series-based approach does not match with Spark’s typical data model causing memory and shuffle issues.

Unable to store large series in Parquet-files.

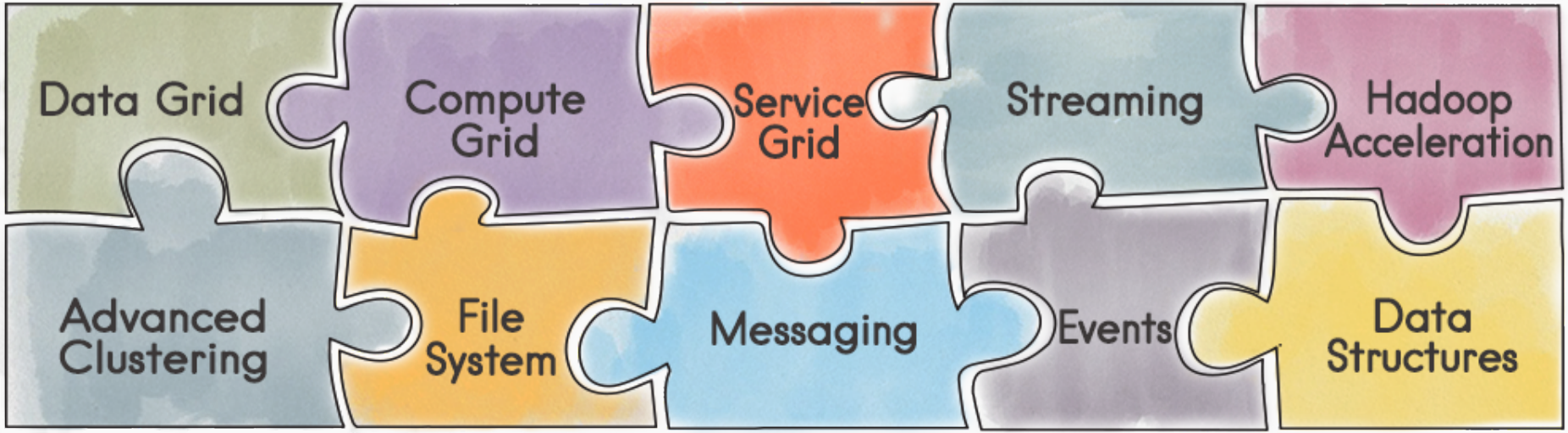

Introducing
Apache Ignite

# Why Apache Ignite?

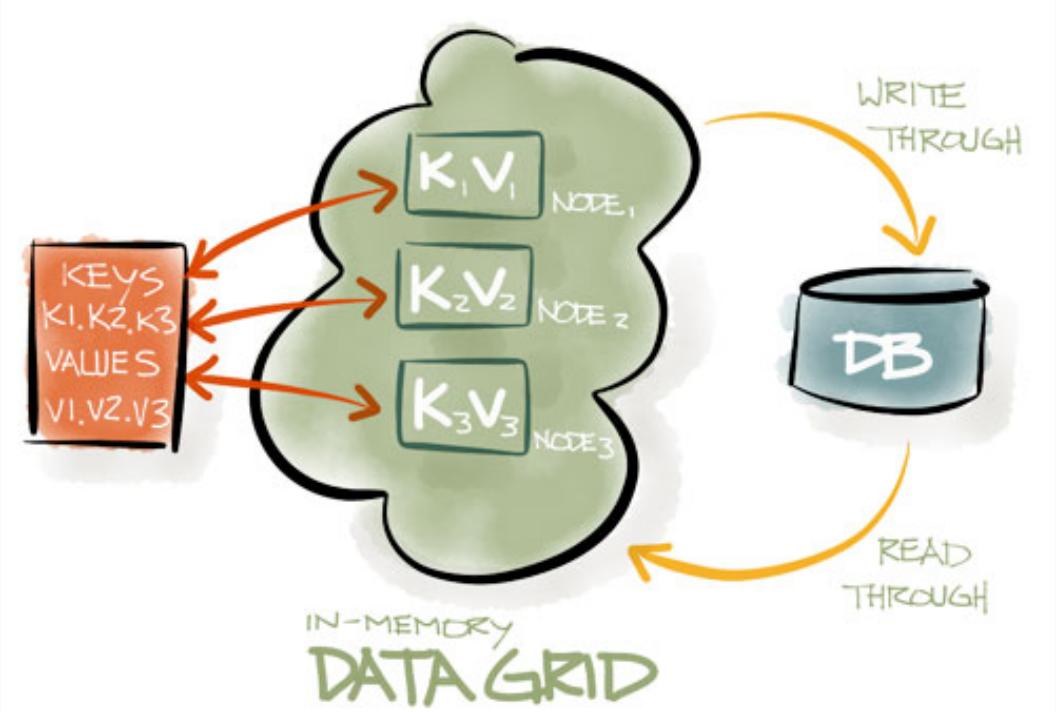
Apache Ignite could provide us with a high level of control on how we want our data to be partitioned.

It also has other niceties like read- and write-through cache behaviour, an API that feels like Java’s Native concurrency API and an easy setup

# Apache Ignite: In-Memory Data Fabric



# Apache Ignite: Data Grid

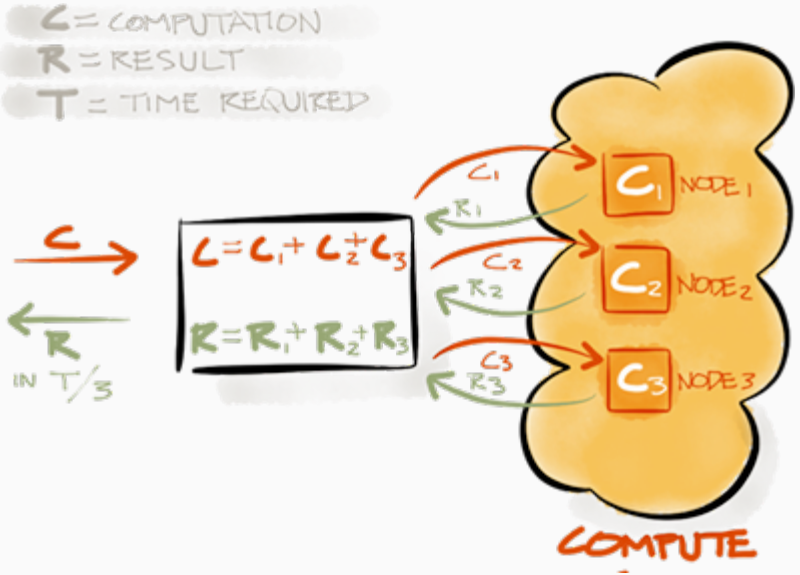



<(datetime_index, series_key), [sequence]>

# Apache Ignite: Compute Grid

Computations can be launched as simple Java Callable's that return a Future<T>

In an interactive environment this is way more flexible than launching Spark



# The sweet spot:
# Co-locate compute with data

Apache Ignite has the ability to execute callables or threads on the node where the data resides in the Data Grid

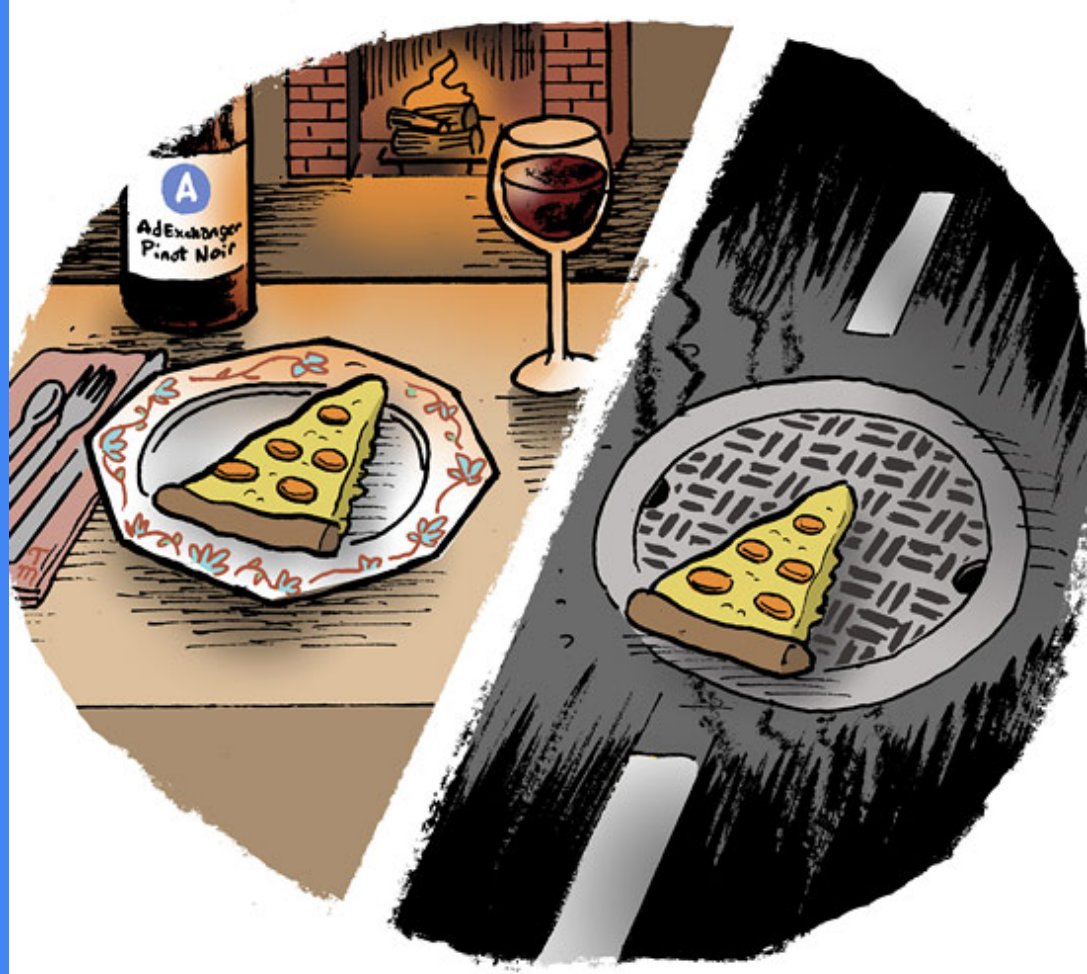
The system is called:

- Affinity Call (Java Callable)
- Or Affinity Run (Java Thread)

Time for some demonstration!

# The benefits

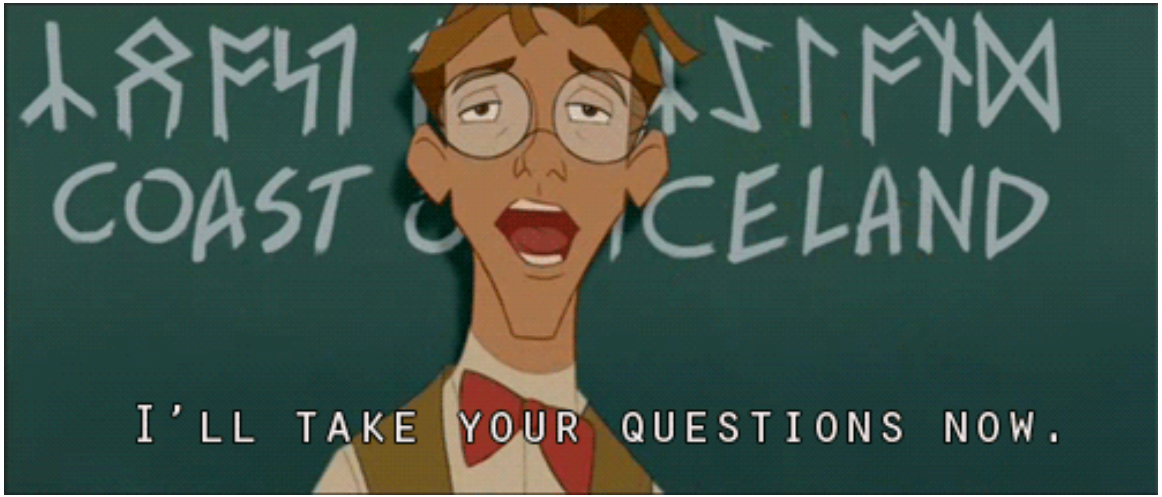
# Processing sequences without loss of context



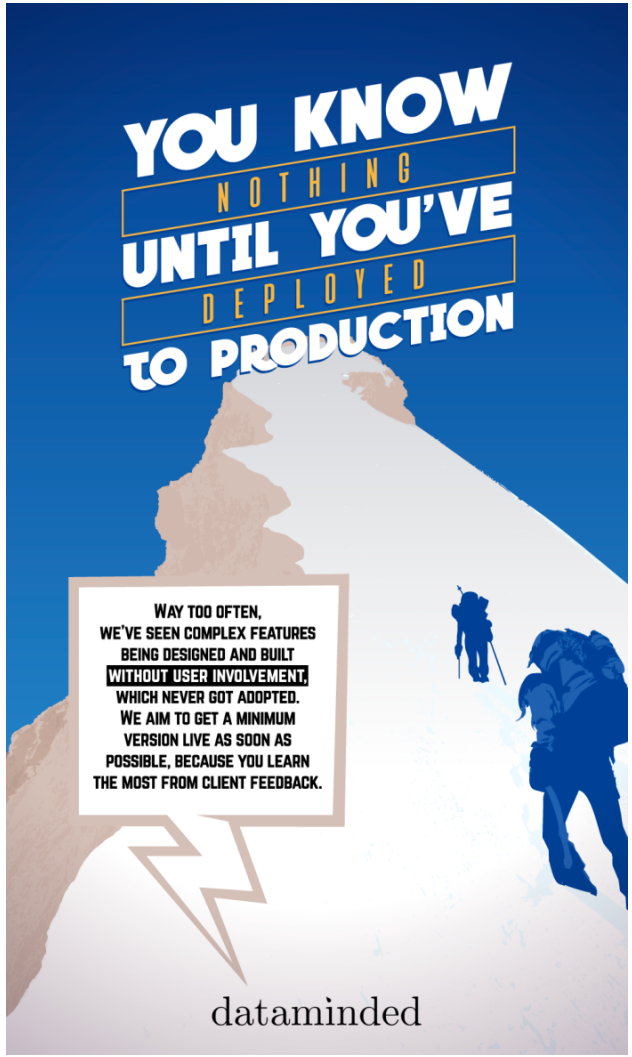
Collocating related sequences results in 0 network transfer when analyzing them

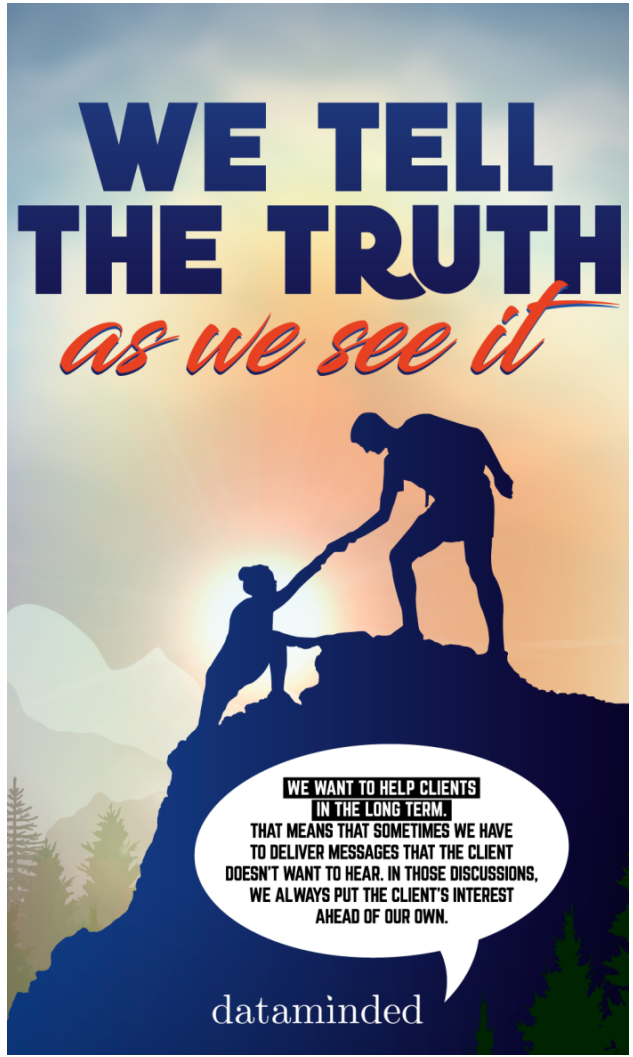
# Analyzing a single sequence is fast and convenient


Overall very high
level of control


COAST
CELAND
I'LL TAKE YOUR QUESTIONS NOW.


Code is a
LIABILITY,
experience
IS AN ASSET
TECHNOLOGY ITSELF IS NEVER THE SOLUTION.
IT'S A TOOL TO DELIVER BUSINESS VALUE. WE START FROM THE BUSINESS QUESTION AND WE ITERATE FAST TO FIND THE RIGHT SOLUTION. WE REFACTOR OUR CODE CONSTANTLY AS WE BUILD EXPERIENCE.
dataminded


YOU KNOW
NOTHING
UNTIL YOU'VE
DEPLOYED
TO PRODUCTION
WAY TOO OFTEN, WE'VE SEEN COMPLEX FEATURES BEING DESIGNED AND BUILT WITHOUT USER INVOLVEMENT, WHICH NEVER GOT ADOPTED. WE AIM TO GET A MINIMUM VERSION LIVE AS SOON AS POSSIBLE, BECAUSE YOU LEARN THE MOST FROM CLIENT FEEDBACK.
dataminded


WE TELL
THE TRUTH
as we see it
WE WANT TO HELP CLIENTS IN THE LONG TERM.
THAT MEANS THAT SOMETIMES WE HAVE TO DELIVER MESSAGES THAT THE CLIENT DOESN'T WANT TO HEAR. IN THOSE DISCUSSIONS, WE ALWAYS PUT THE CLIENT'S INTEREST AHEAD OF OUR OWN.
dataminded