Define Once, Evaluate Anywhere

Building Repeatable and Correct Features at Stripe

Kelley Rivoire
Data @Stripe

Outline

- ML at Stripe!
- The reality of features
- Our approach
- How we run it

Stripe

The new standard in online payments

Stripe is the best software platform for running an internet business. We handle billions of dollars every year for forward-thinking businesses around the world.

Real World ML (@Stripe)

- Stripe provides a toolkit to start and run an internet business

- Need to make decisions *quickly* and *at scale.*
- Our actions affect *real* businesses.

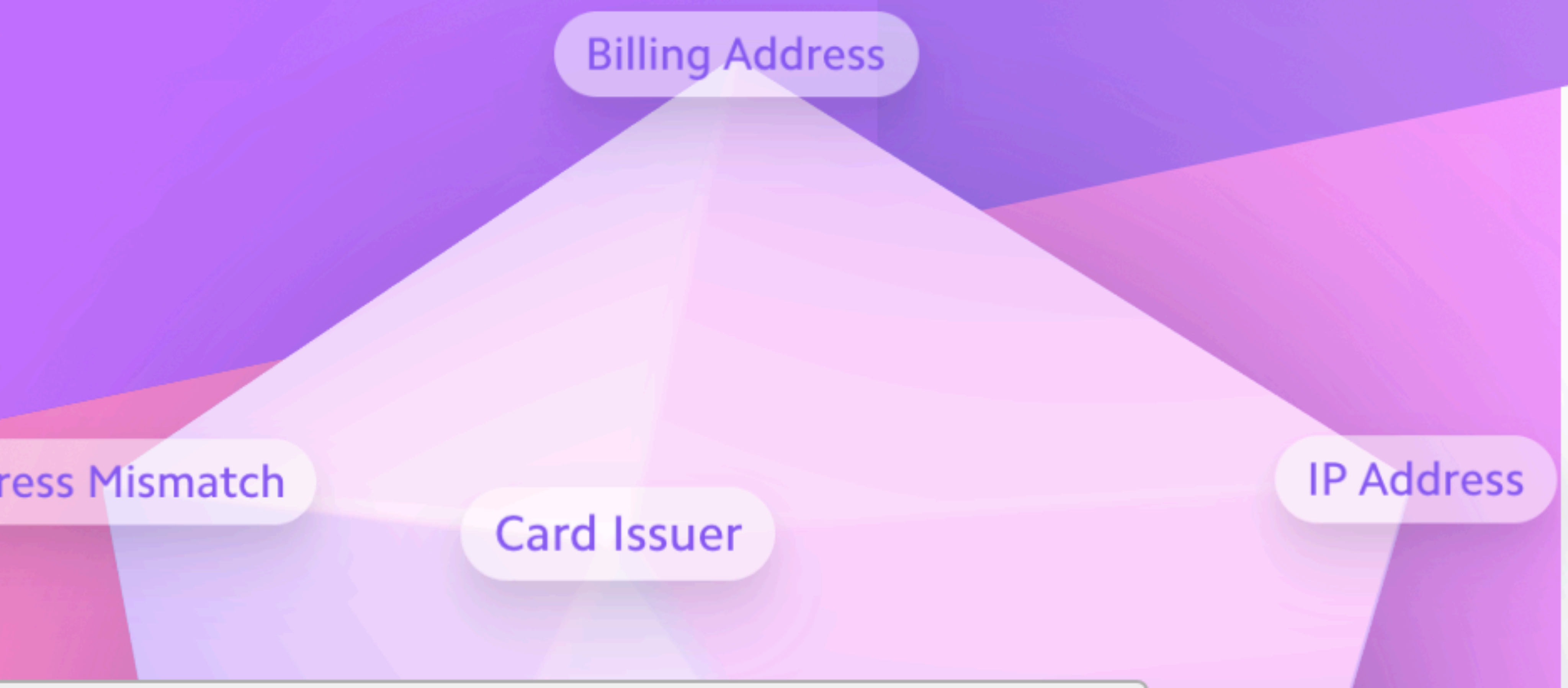
stripe

RADAR

ACTIVATE RADAR

GET IN TOUCH

Modern tools to help you beat fraud, fully integrated with your payments.



FRAUD PREVENTION DONE RIGHT

Old ways of combating fraud were never
designed for modern internet businesses

$10.00 USD BLOCKED

PAYMENT

✓ Mark as safe

Payment blocked due to high risk — Jul 26, 2016, 5:04 AM

Stripe risk evaluation: high — Jul 26, 2016, 5:04 AM

Primary risk factor: This card (0008) has been used from an unusually large number of IP addresses across the Stripe network over the last 24 hours. Learn more about risk evaluations.

stripe

Billing and subscriptions for fast-growing businesses

"We recovered **12% of revenue** through **Stripe Billing's automatic card updater** in 2017."

Bench

REDUCE DECLINED PAYMENTS BY UP TO 45%

Nearly a quarter of churn is caused by missed payments or declined cards. In 2017, Stripe's recovery tools reduced payment declines for users by 45% on average and increased revenue by 10% on average.

Automatic card updater

Stripe works directly with card networks to update payment details with new card numbers or expiry dates.

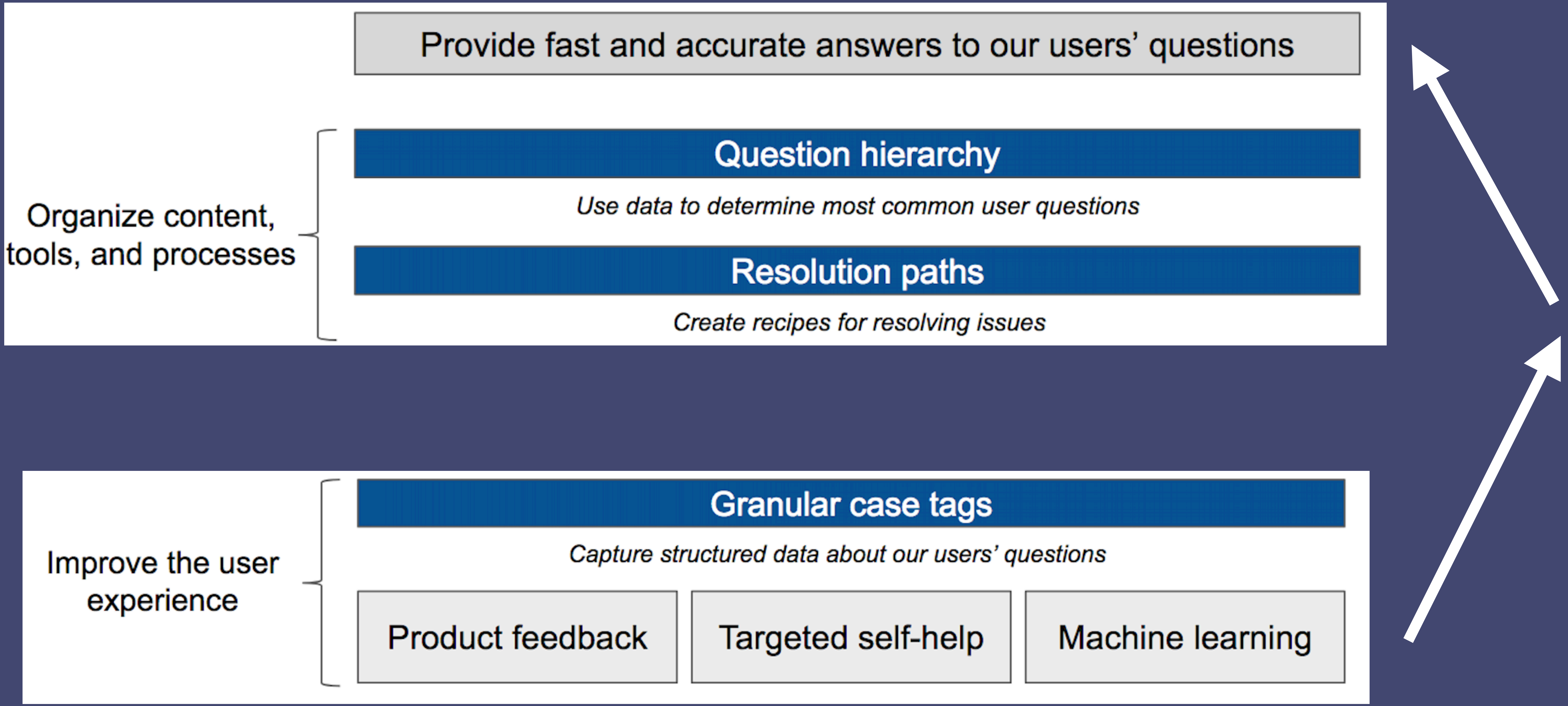
Smart retry logic

Stripe uses machine learning algorithms that train on data from across the Stripe network to optimize retry logic and minimize failed payments.

Payment reminders and overdue notices

Maximize your chances of getting paid with pre-built email reminders for missed or overdue payments.

Improving our operations



A fiction about ML

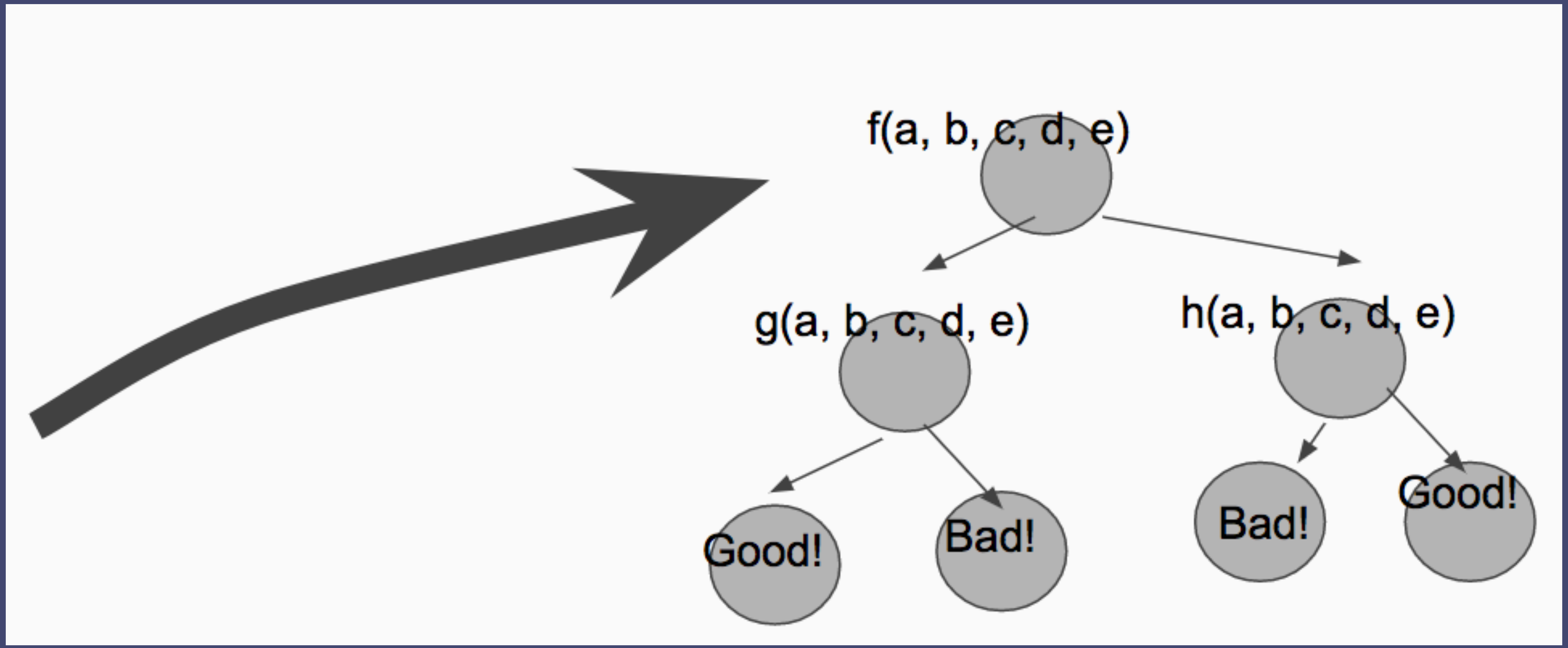
a	b	c	d	e	X
0	1	1	0	0	G
1	1	1	0	1	B
1	0	1	1	0	B
0	1	0	1	0	B
1	0	1	0	0	B

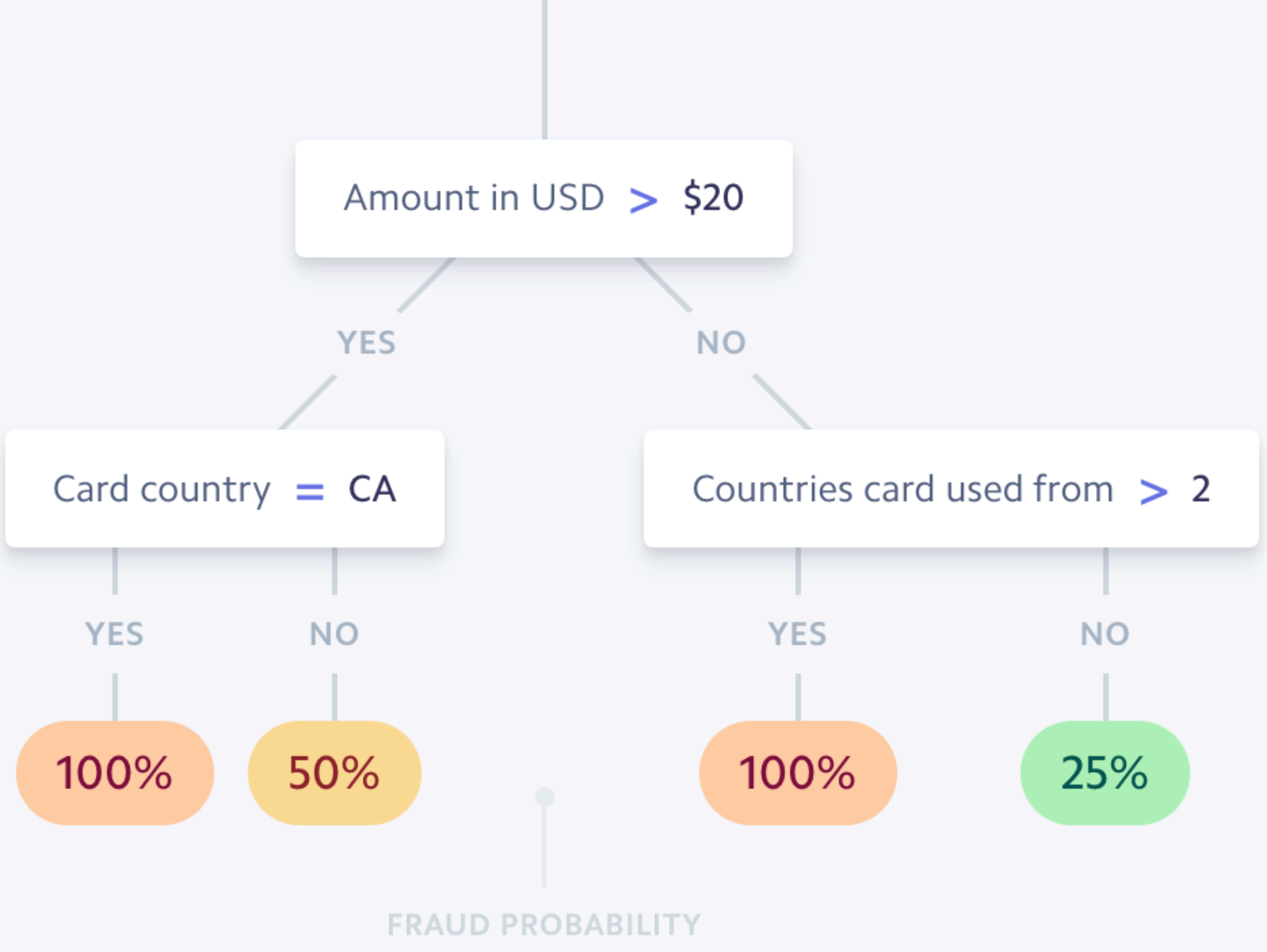
We have a beautiful table of data:
a tall matrix that represents
Ground Truth about Reality.

stripe

A fiction about ML

a	b	c	d	e	X
0	1	1	0	0	G
1	1	1	0	1	B
1	0	1	1	0	B
0	1	0	1	0	B
1	0	1	0	0	B





Amount in USD > $20
YES
NO
Card country = CA
Countries card used from > 2
YES
NO
YES
NO
100%
50%
100%
25%
FRAUD PROBABILITY

Reality

Feature engineering: turn a giant pile of serialized data into a sane matrix to feed to a training algorithm.

Amount in USD	Card country	Countries card used from (24h)	Fraud?
$10.00	US	1	No
$10.00	CA	2	No
$10.00	CA	1	No
$10.00	US	1	Yes
$30.00	US	1	Yes
$99.00	CA	1	Yes
$15.00	CA	3	Yes
$70.00	US	1	No

Key challenges

- There are many different data stores and event streams. How do we integrate them?
- How to produce a historical view of state *when a prediction would have been made*? Time-aware joins are easy to get wrong.
- How to prevent “label leakage” with labels leaking into training data?
- How to make sure *data for training* is consistent with *data for scoring*?
- How to share *code* to generate data for training and scoring?

stripe

Training on future data

Feature idea: fraud rate by e-mail!

Compute fraud rates

Both charges disputed as fraud!!

kelley@stripe.com makes a charge on business B

kelley@stripe.com makes a charge on business A

stripe

Features are used in rules, too!

Add a rule for blocking payments

All payments that pass this test will be blocked automatically

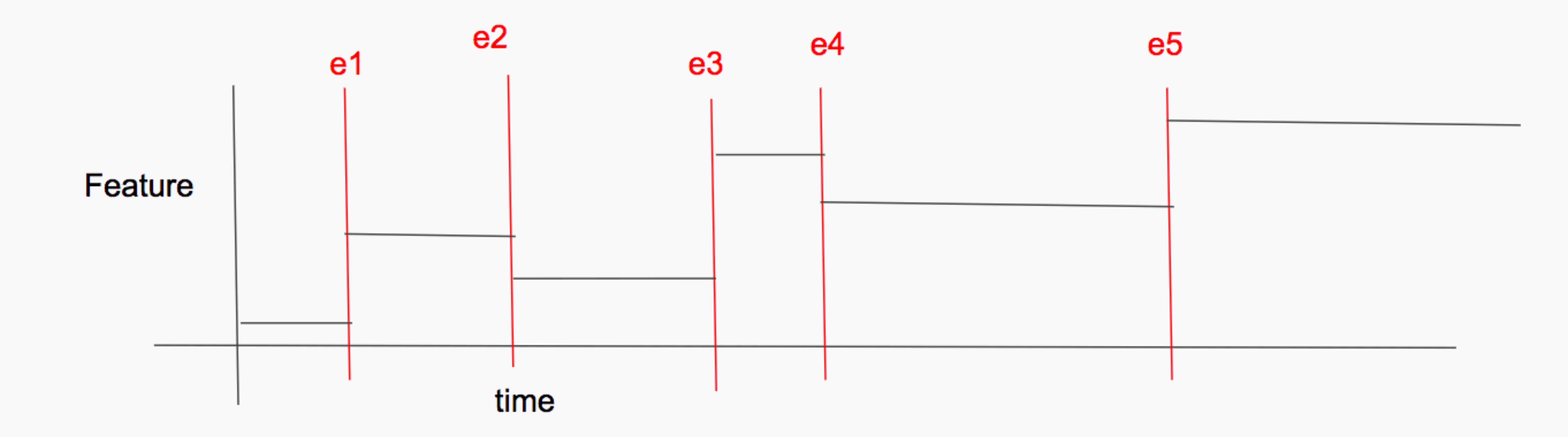
```
Block if :card_funding: = 'prepaid' and :amount_in_usd: >
1000.00 and :card_country: != 'US'
```

1,323 payments would have matched this rule

1,181	28	134
Blocked or declined	Refunded or disputed	Successful

ADD AND ENABLE

Features and events



The input matrix to models are Features attached to Events

a	b	c	d	e	X
0	1	1	0	0	G
1	1	1	0	1	B
1	0	1	1	0	B
0	1	0	1	0	B
1	0	1	0	0	B

- At an event, we can lookup a feature value (which exists at all times)
- With the event and the feature we can either train or evaluate

We require all data inputs to be **evented data.**

stripe

Core types: Event, Feature

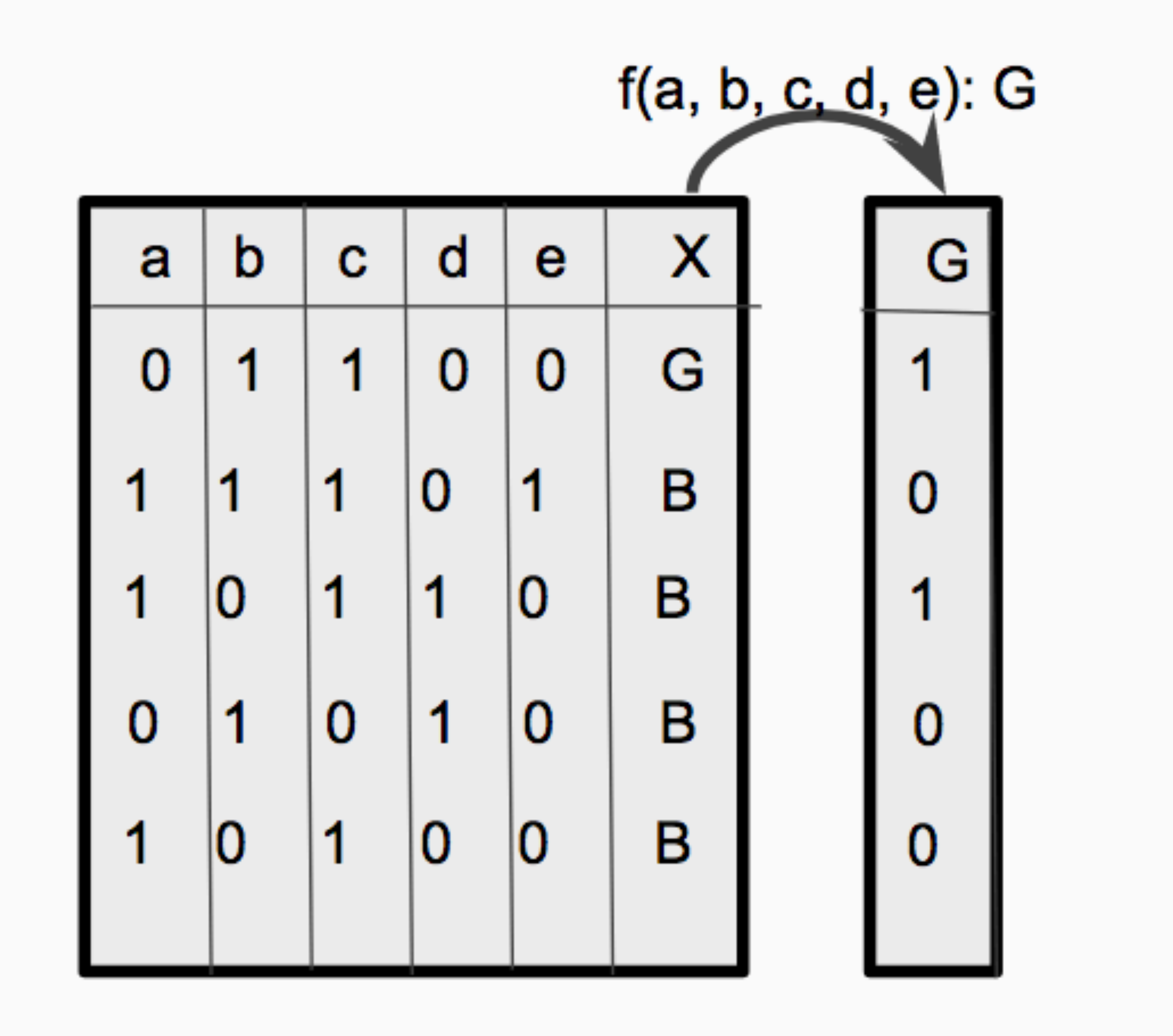
```scala
sealed trait Event[A] {
  def map[B](fn: A => B): Event[B]
  def mapWithTime[B](fn: (A, Time) => B): Event[B]
  def concatMap[B](fn: A => Iterable[B]): Event[B]
  def filter(fn: A => Boolean): Event[A]
  def ++(that: Event[A]): Event[A]
}

object Event {
  def empty[A]: Event[A]
  def source[A](name: String): Event[A]
}

sealed trait Feature[K, V] {
  def map[W](fn: V => W): Feature[K, W]
  def product[W](that: Feature[K, W]): Feature[K, W]
}

object Feature {
  def constant[K, V](v: V): Feature[K, V]
}
```

Events are things that pop out of Kafka!

Features are about a subject of type K. We can partition updates to feature by the K, e.g. K=user, merchant, tweetid, contentid, etc...

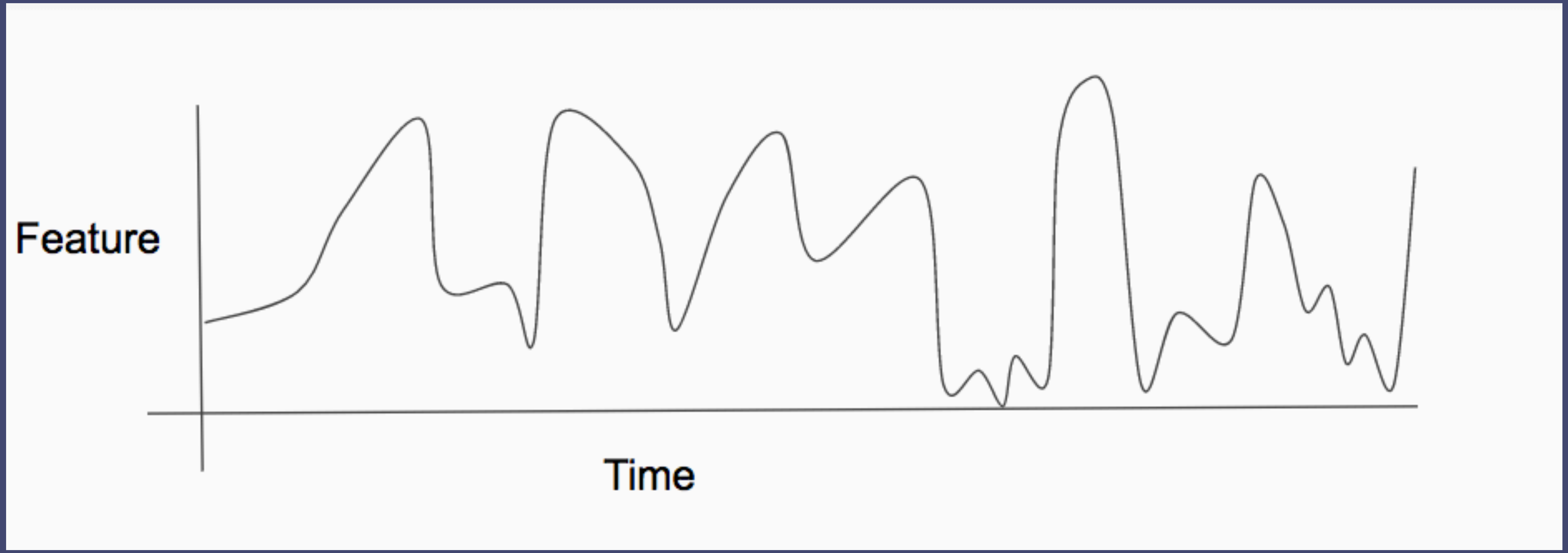
stripe

Feature.map creates new columns from old



- E.g. from Feature[Merchant, (TotalChargeCount, TotalChargeAmount)] we can use .map to get average charge amount.

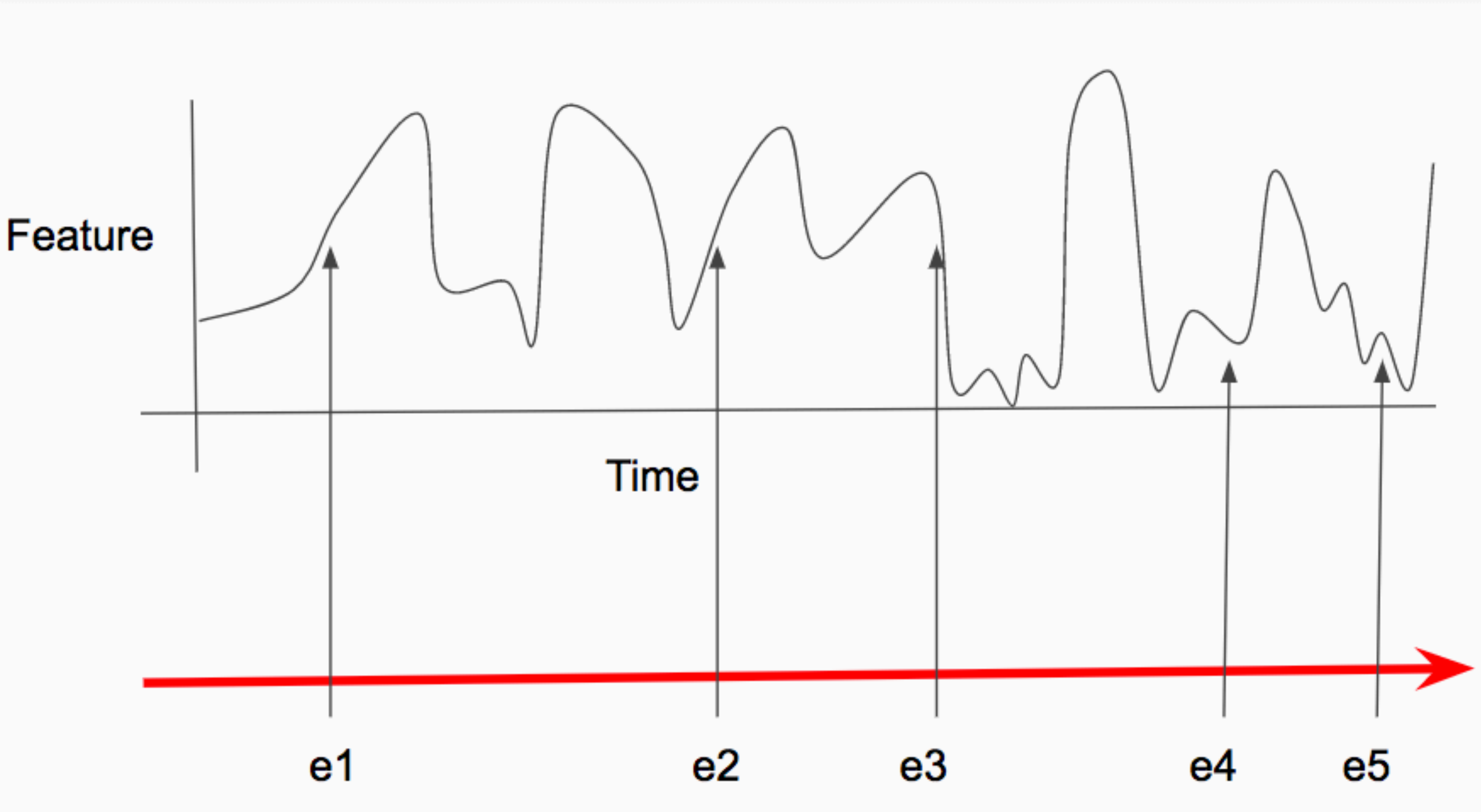
Event.lookup reads Features




```
object Event {
  def lookup[K, V, W](
    e: Event[(K, V)],
    f: Feature[K, W]): Event[(K, (V, W)]
}
```

stripe

Event.lookup reads Features



When generating training data, it is **critical** that the events see the value of the feature **as it was at the event's time**.

- very tedious to do by hand.
- keeping this declarative the system can manage these lookups correctly.
- Call this "temporal consistency"

Example features

```
val barks = Event.source[Bark]("barks")
 val jumps = Event.source[Jump]("jumps")
 val howls = Event.source[Howl]("howls")

 val hasBarked: Feature[String, Boolean] =
   Feature
     .latest(barks.mapN(b => (b.name, ())))
     .mapN(_.isDefined)

 val averageVolume: Feature[String, AveragedValue] =
   Feature
     .sum(
       barks.map(b => (b.name, AveragedValue(b.decibels))) ++
         howls.map(h => (h.name, AveragedValue(h.decibels))))
     .name("averageVolume")

 val averageVolume2: Feature[String, AveragedValue] =
   Feature
     .sum(barks.map(b => (b.name, AveragedValue(b.decibels))))
     .zip(Feature.sum(howls.map(h => (h.name, AveragedValue(h.decibels)))))
     .map { case (x, y) => x + y }
     .name("averageVolume2")

 val isAnimal: Feature[String, Boolean] =
   Feature.const[String, Boolean](true)
```

But how do you actually run it?

- Once we have the AST, we have several backends that can evaluate a feature, either a total history or evaluate at a point in time, given the Event source

- E.g. interpreter, map/reduce-like backend, push-based realtime backend

Map/reduce-like backend

```
private[features] def innerCompile[K, V](
    memo: Memo,
    feature: Compiler.NonMappedFeature[K, V]
): (Memo, FinalAggregated.Unmapped[K, V]) =
  memo.getFeature(feature) {

    feature match {
      case Named(name, feat, _) =>
        innerCompile(memo, feat)
      case Latest(LatestFeature(event, o, ser)) =>
        val (m1, ex) = innerCompile(memo, event)
        (m1, ex.latest(o, ser))
      case Sum(SumFeature(event, o, m, ser)) =>
        val (m1, ex) = innerCompile(memo, event)
        (m1, ex.sum(o, m, ser))
      case Zip(lhs, rhs) =>
        val (m1, ex1) = innerCompile(memo, lhs)
        val (m2, ex2) = innerCompile(m1, rhs)
        (m2, ex1 zipUnmapped ex2)
    }
  }
```

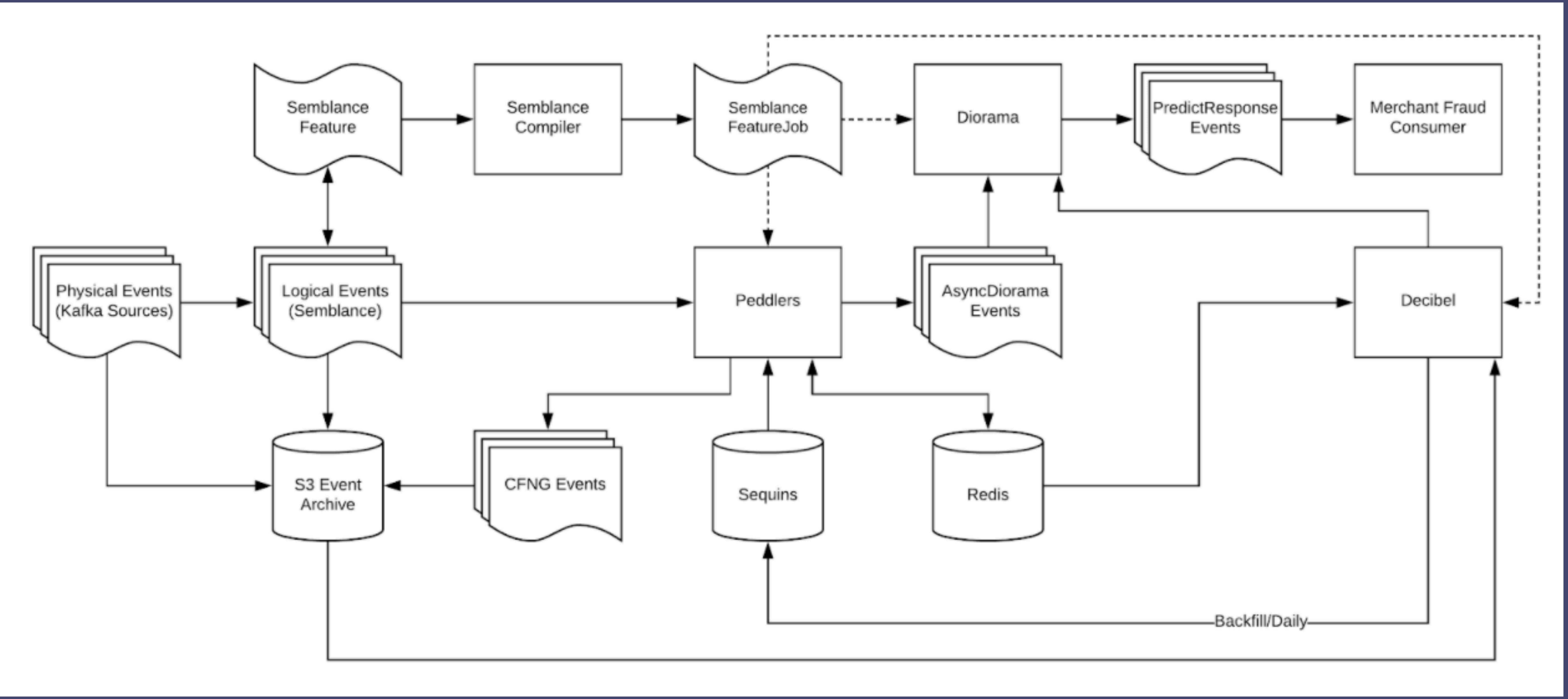
Do you use it?

- Yes! We use this to generate, e.g., features that score our fraud models

- The most complex graphs have around 1400 feature/event nodes.

- We can update features for very complex feature graphs in around 60ms p99 which can involve updating more than 100 keys.

How does it fit together?



stripe

Summary

- This system gives a minimal and principled API for feature engineers.
- The principled nature means the backend system has a lot of power to optimize or run in different environments (easy to change how we compute, without changing what we compute).
- Solves the problem of separating business logic completely from the implementation details.
- Frees the feature engineer from having to worry about temporal consistency.

stripe

Come work with me!

- Stripe is hiring for a lot of interesting data and ML roles!
- We use data technology to track and move money.
- We are building state-of-the-art ML infrastructure for feature engineering, model training and evaluation.

Thanks!

Special thanks to Oscar Boykin, Erik Osheim, Sam Ritchie, Travis Brown

Machine Learning Infrastructure @Stripe