SPAMMERS VS. DATA: MY EVERYDAY FIGHT

Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO

ABOUT LOVOO

LOVOO is a dating and social app. Thus, people are trying to engage into various sorts of activities with other users ;).

STORY TIME

I got some bad news.

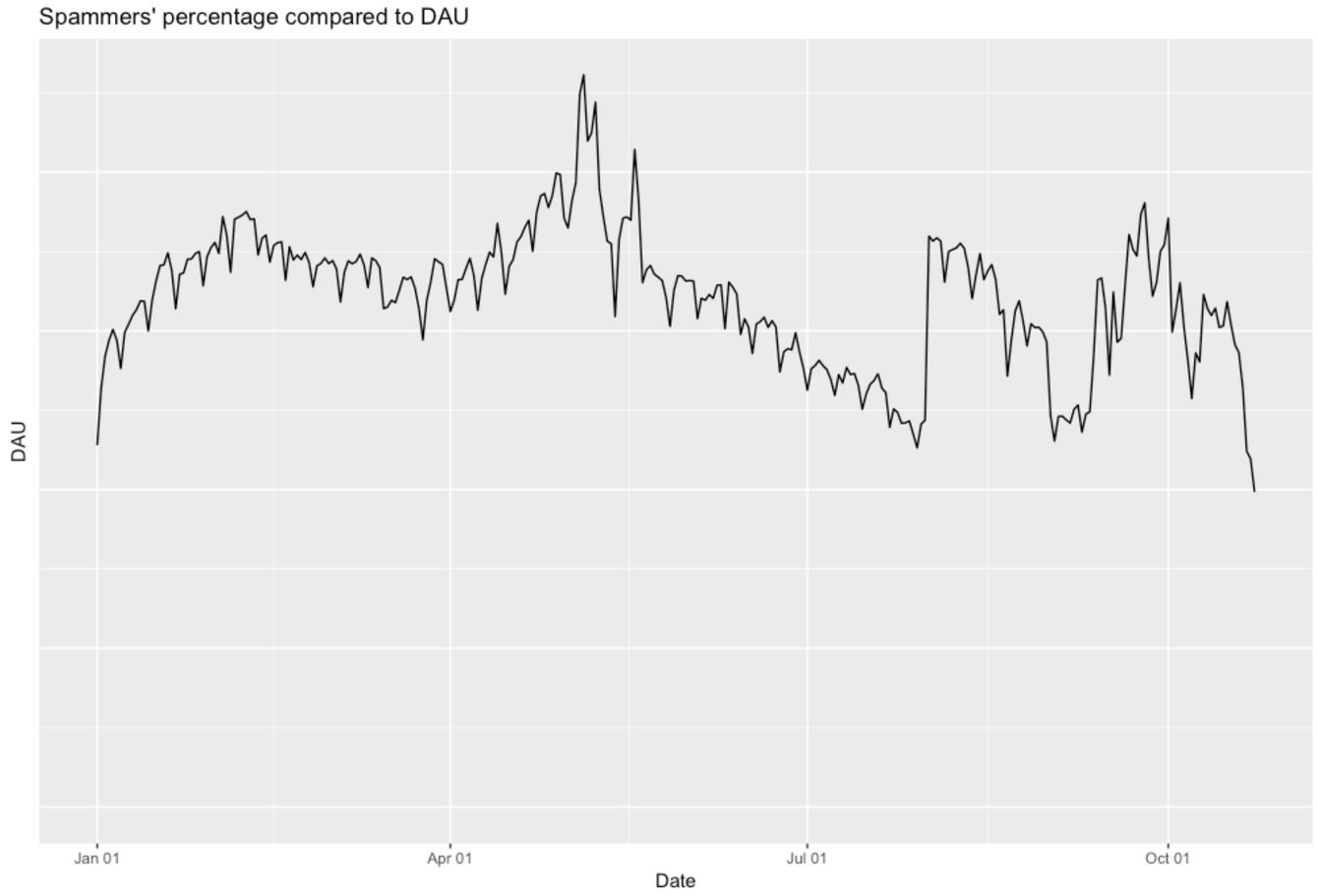
STORY TIME

Where there are real users, chances are that spammers will be there.

STORY TIME

Their goal is to use our app to **advertise** other platforms and to try to **scam** others.

STORY TIME


Spammers' percentage compared to DAU
DAU
Jan 01
Apr 01
Jul 01
Oct 01
Date

STORY TIME

The spammers evolve! They are constantly trying to bypass and beat our systems.

STORY TIME

And I do the same.

STORY TIME

So, it is a fight, in which my main weapon is **data**.

STORY TIME

The purpose of this talk is to present the **tools**, **algorithms**, and **architecture** we use to **fight** the spammers from our platform and to avoid the **proliferation** of them.

AGENDA

- STORY TIME
- ANTISPAM
- DATA + ALGORITHMS + MACHINE LEARNING
- SOME NUMBERS
- RECAP AND CONCLUSION :(

ANTISPAM

Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO

ANTISPAM

- LOVOO's solution to detect and block spammers
- It uses a mix of machine learning and heuristics to identify the spammy profiles
- It is not a monolithic system; it is made of over 15+ components
- Written in Go
- It is amazing ❤️

STACK


GO


Google Cloud


APACHE
kafka


python


scikit
learn


K


TensorFlow


R

ANTISPAM ARCHITECTURE

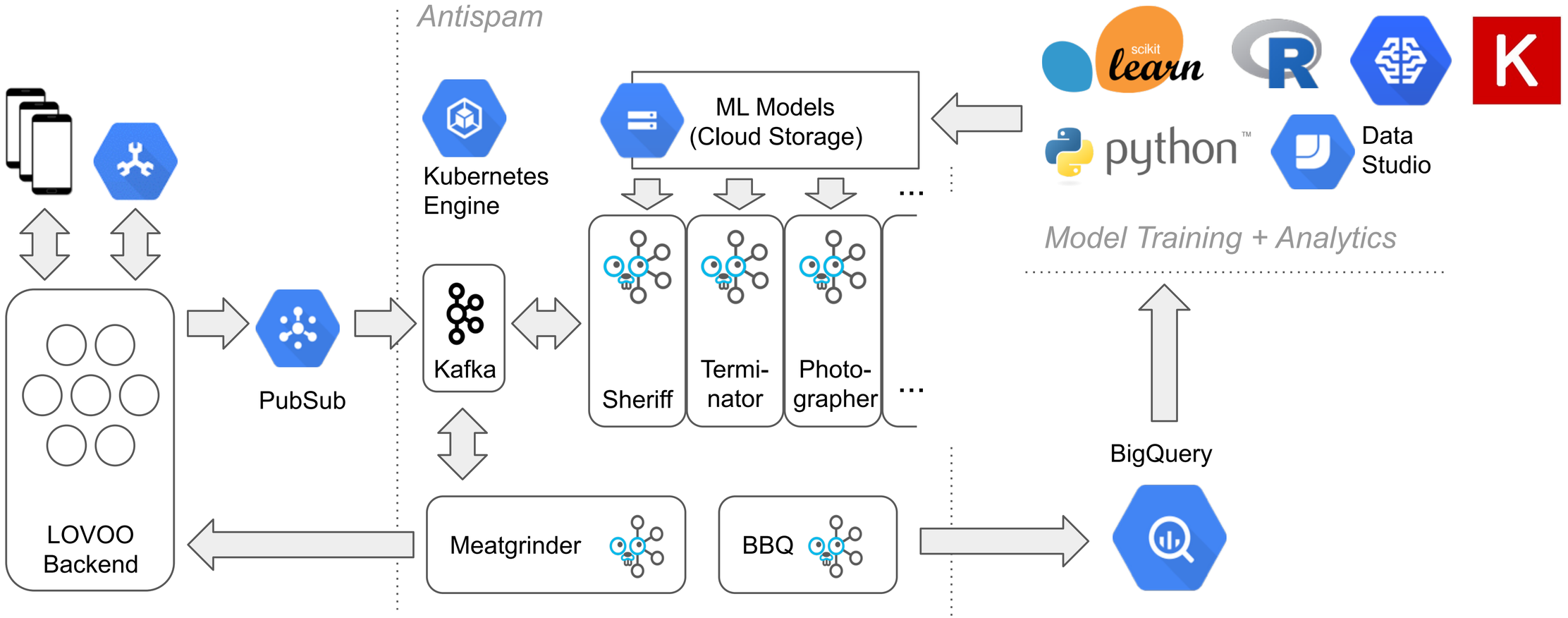
Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO

ANTISPAM ARCHITECTURE

- Antispam is a platform made of over 15 components.
- At its center is Kafka, which is used as the mean of communication between them.
- We interface Kafka using Goka, a Go open-source library written in-house
- In Kafka, we keep our live data stored in table-like structures.
- It makes use of many products from Google Cloud Platform
- It runs on Kubernetes Engine

ANTISPAM ARCHITECTURE



DATA + ALGORITHMS + MACHINE LEARNING

Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO

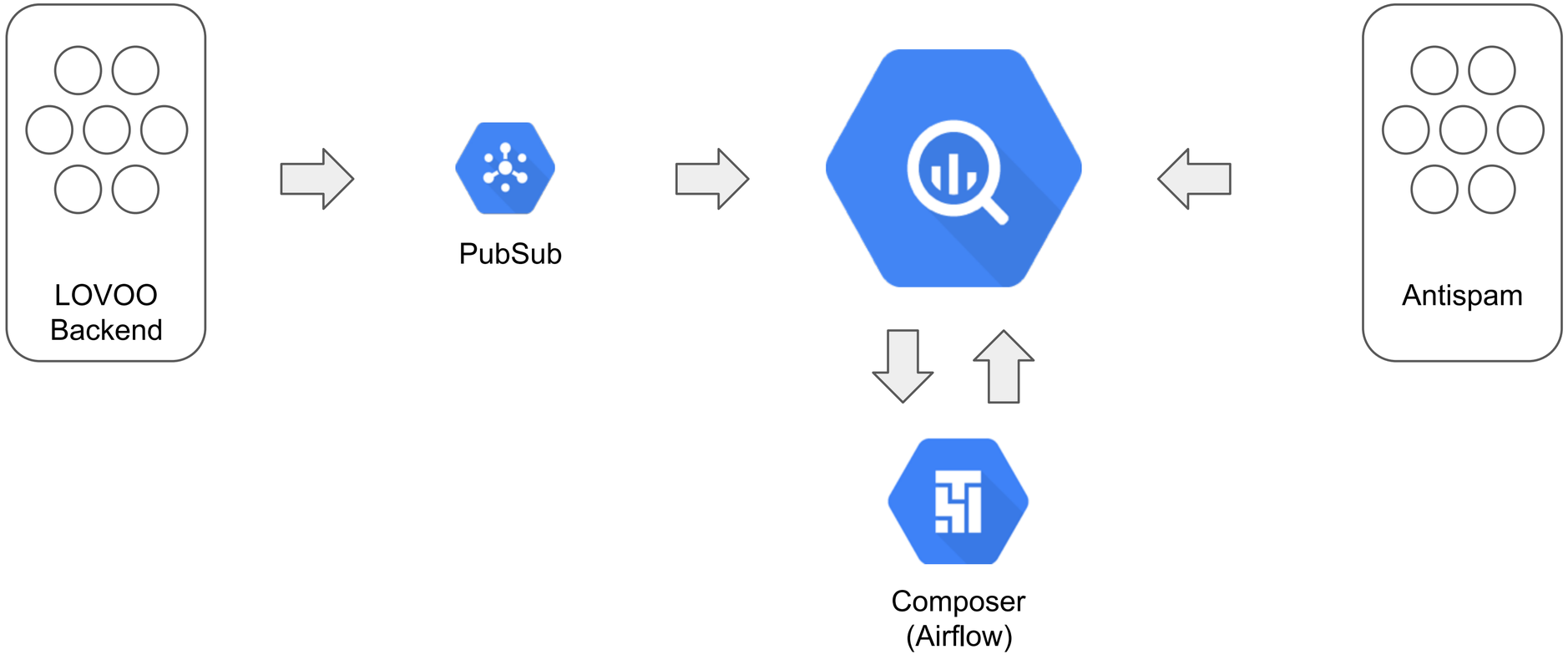
ALGORITHMS WE EMPLOY

- Antispam uses a mix of machine learning algorithms and heuristics to detect and fight the spammers
- The data these algorithms use is made of text, counts, images and sequences (sequential data)
- Some of the algorithms are developed from scratch by us, or trained using Keras or scikit-learn

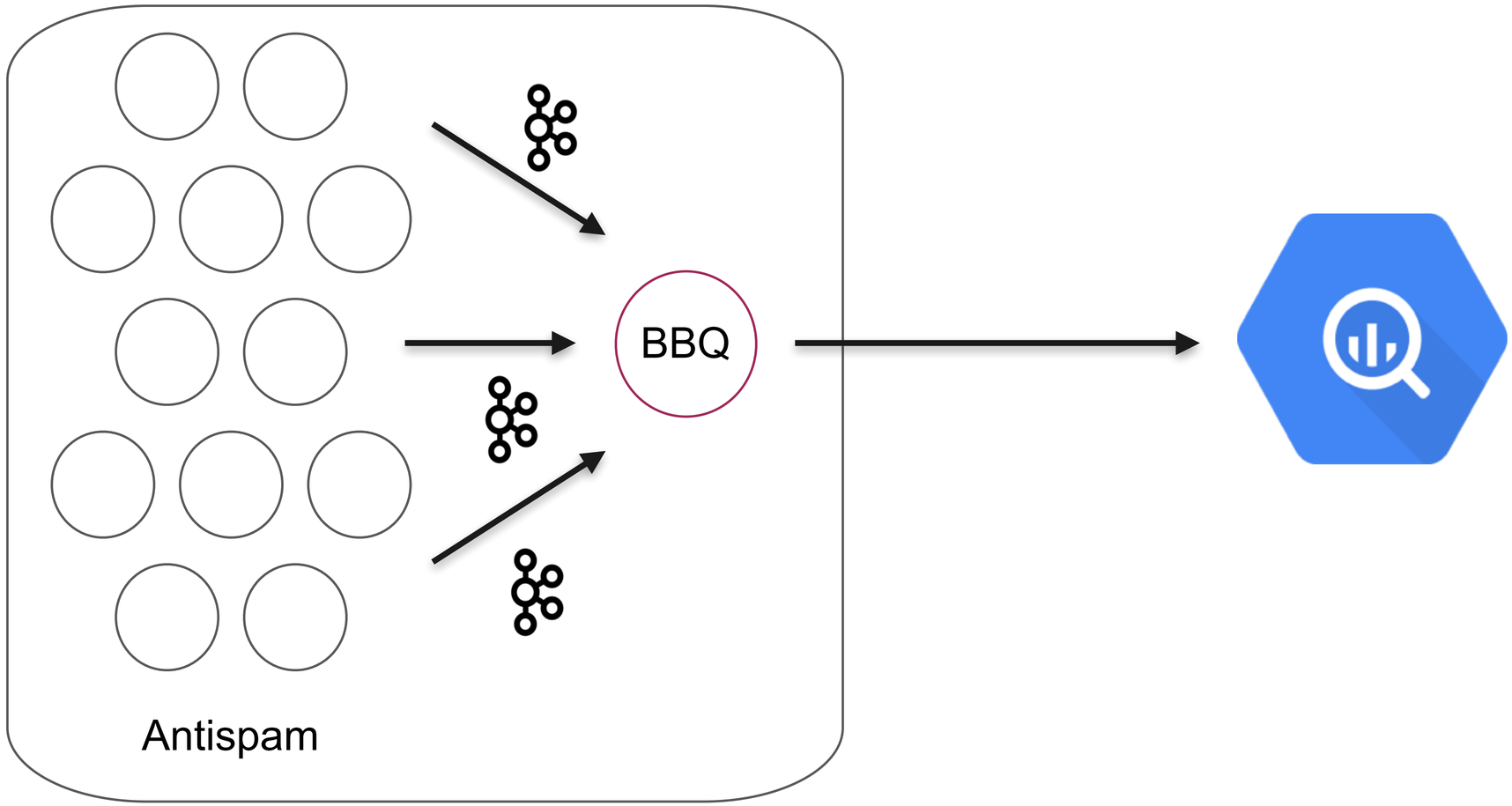
DATA

Everything starts with **data**, right?

DATA



BBQ



DATA

Now that we have the data, we are able to learn from it :)

HEURISTICS BASED COMPONENTS

Terminator

Behaviour Rules

Community

MACHINE LEARNING BASED

Now the **cool** ones :)

Spammers and **non-spammers** behave in different ways. For example, a typical female user, might do a couple of likes, send 1-2 messages and reject a billion boys (sorry guys), but a spammer user usually tries to reach a broad audience and thus use the app in uncommon ways.

SHERIFF

- *User Behaviour Model*
- This system classifies users (spam and no spam) based on their behaviour
- Sheriff's features describe a user's execution ratio of events
- **Logistic Regression** trained using **scikit-learn** with a dataset of over 1m observations and ~30 features

SHERIFF

message_sent	like	event_3	…	*spam*
0.33	0.33	0.33		**0**
0.25	0.25	0.50		**0**
0	0	1		**1**
0.17	0.58	0.25		**0**

In the LOVOO app, there are two fields in which a user can write any text they want: the **username** and “**about me**” section.

Of course, spammers use them 😡

Username example: “**sex me call 123456789**”

“about me” example: “**interested in hot steamy sex? send me an email to xxxhhhdwdwdf@hotsteamysex.com**”

ROSETTA

- Component made of several text-related models
- The input of these models is a numerical vector representation of the text
- **Logistic regression** trained in **scikit-learn**

ROSETTA

“about me” models

- One model per country
- Training set varies; ~40000 to ~100k observations
- Data pre-processing step: lower case everything, remove stop words, filter out rows where `len(text)<=X` and vectorise text

usernames models

- One general model
- Training set is made of ~15000 names
- Data pre-processing step: lower case everything, convert name to character-level bigram, filter out rows where `len(text)<=X` and vectorise text

Let's talk about **images**.

Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO


interested in hot steamy sex? send
me an email to
xxxhhhdwdwdf@hotsteamysex.com

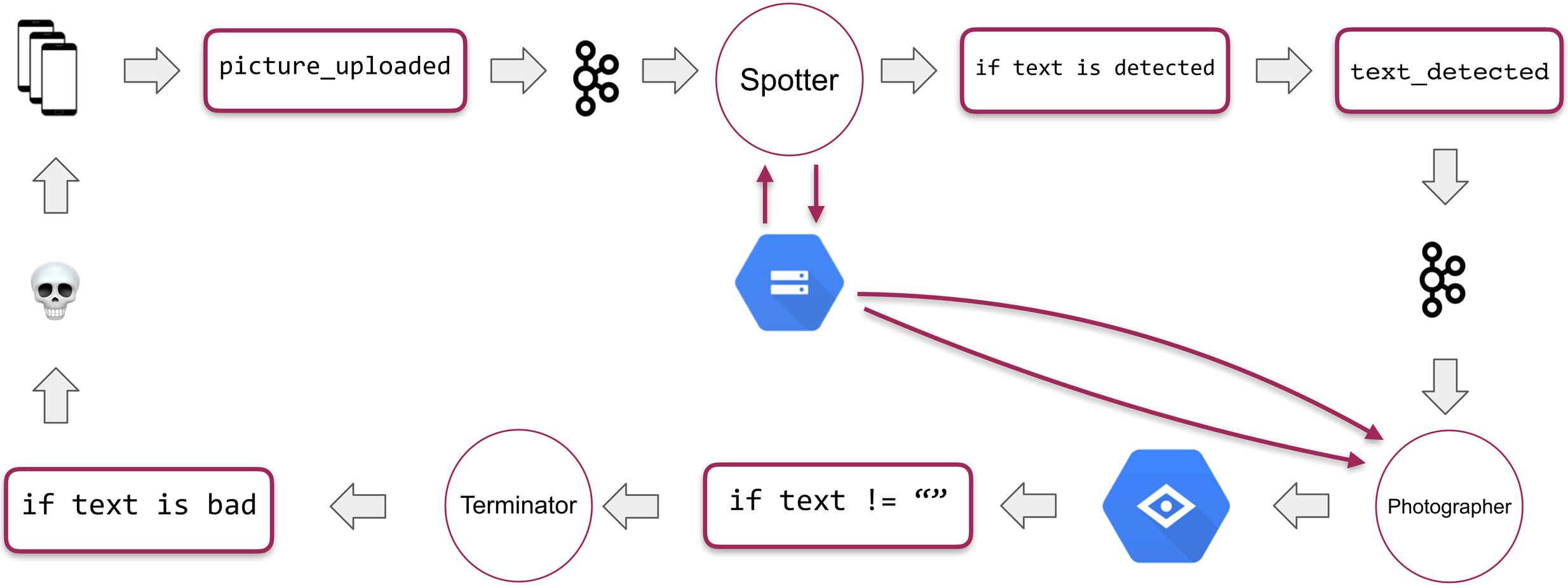
Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO

SPOTTER AND PHOTOGRAPHER

- Components that deal with images
- The goal of Spotter is to **detect** text on an image, and the Photographer **extracts** the text
- The extracted text is sent to another Antispam component - Terminator - to determine if it good or bad
- Spotter is written in **Python** and used **OpenCV** for the detection, and the text extraction part is done by **Google Cloud Vision API**

SPOTTER AND PHOTOGRAPHER



The **new** stuff.

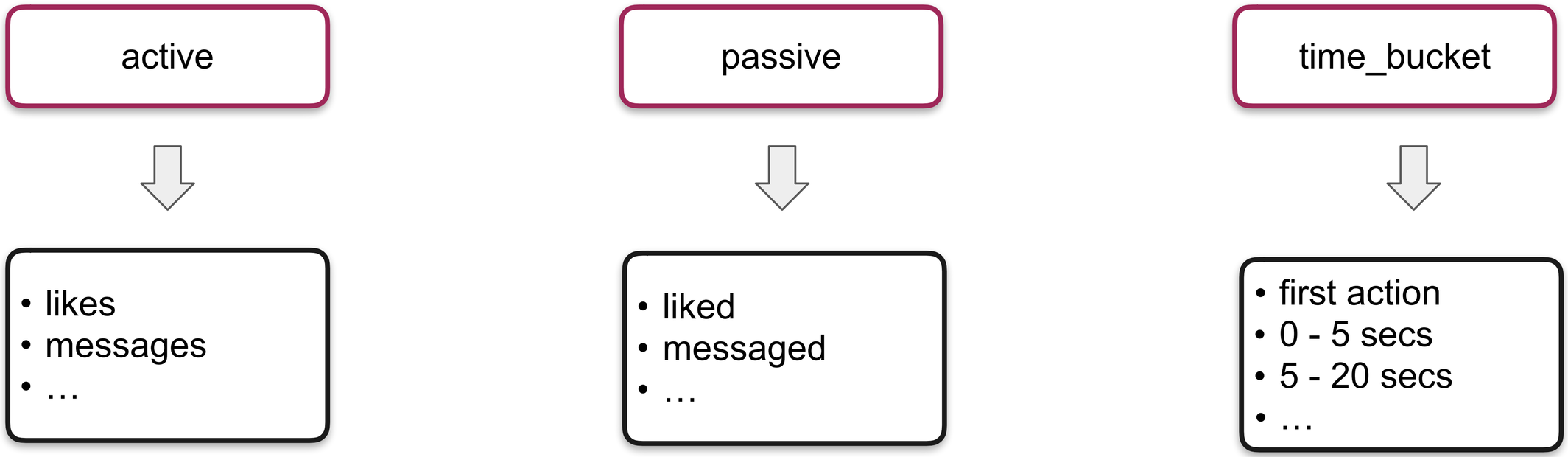
Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO

TRACER

- Component that deals with **sequences**.
- It tries to answer the question: *"how is the user using the app?"* and *"what's the order of the events executed by the user?"*
- Tracer's goal is to detect spammers by using their sequence of events, which we encode into something named *actions*.
- Model is a **Recurrent Neural Network** with **Long Short-Term Memory** (LSTM) units, trained using **Keras**.
- Training is performed in **Google Cloud's ML Engine** and also served there.

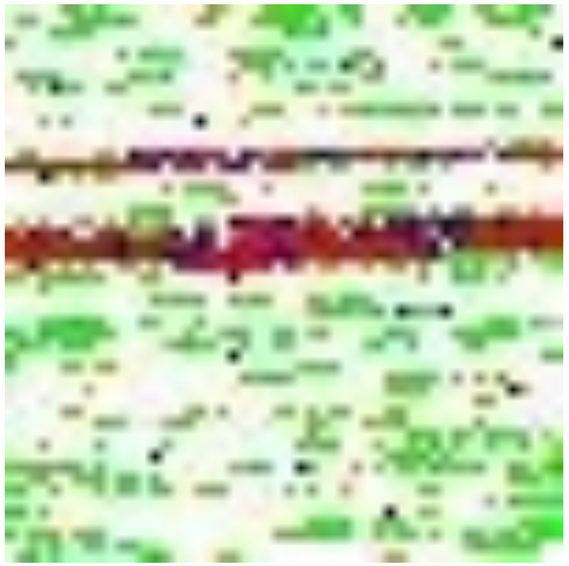
TRACER'S ACTIONS



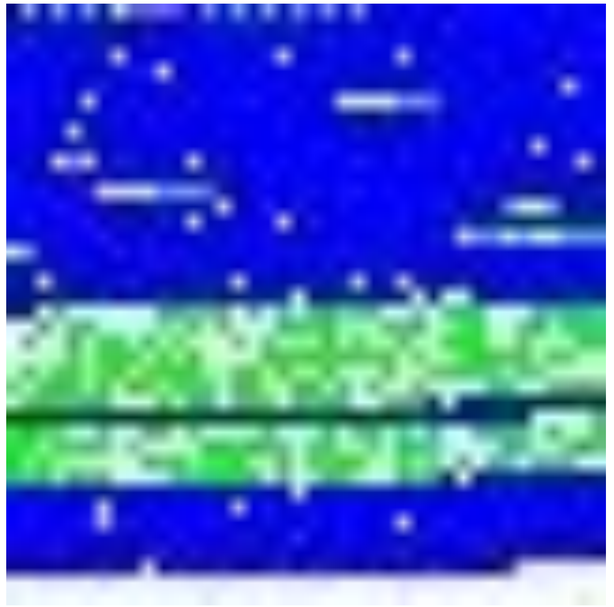
TRACER'S ACTIONS

```
1l2l2l4D...
```

TRACER'S ACTIONS



Non-spammer 👼



Spammer 😡

Lastly, our **newest** experiment...

CONVICTION

- “Meta”-model that uses the scores of other components
- Its features are the confidences values of Sheriff, Rosetta, Tracer and so on
- Our theory is that combinations of these scores might lead us to produce more punishments
- In development...
- ...but it will probably be another logistic regression, random forest or just a bunch of heuristics rules produced by hand

CONVICTION

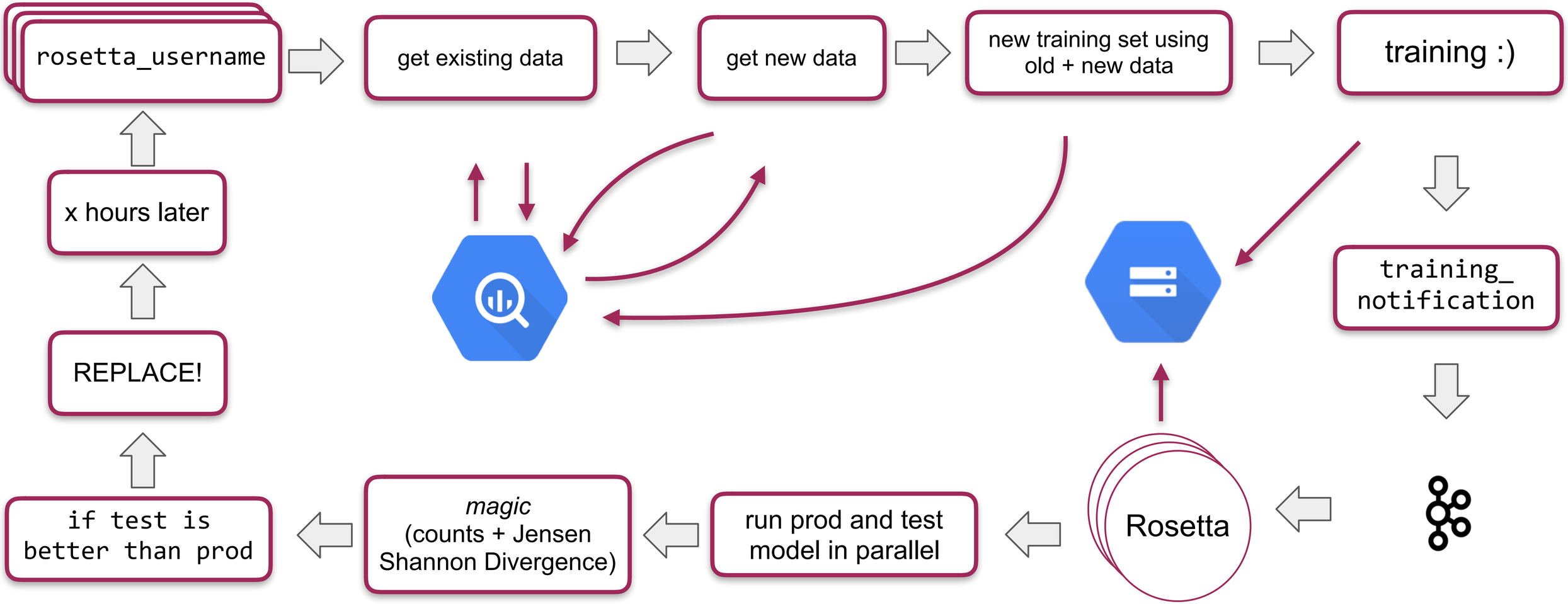
tracer_score	sheriff_score	rosetta_username_score	…	spam
0.38	0.18	0.28		0
0.60	0.75	0.40		1
0.99	0	0		1
0.17	0	0.80		1

No more algorithms.

AUTOMATIC TRAINING

- Pipeline we designed to automatically train and reload some of our models
- We only apply it to models that we fully understand and don’t represent a high risk

AUTOMATIC TRAINING



Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO

LOVOO

SOME NUMBERS

Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO

% SPAM USERS

Quarter	Spam users
Q3/2016	0.3 %
Q4/2016	0.2 %
Q1/2017	0.2 %
Q2/2017	0.3 %
Q3/2017	0.3 %
Q4/2017	0.3 %
Q1/2018	0.3 %

LOVOO's Antispam Transparency Report

https://tech.lovoo.com/2018/06/04/seventh-transparency-report/

TIME IT TAKES TO PUNISH SOMEONE

Quarter	Spam users
Q3/2016	2.2 h
Q4/2016	2.1 h
Q1/2017	1.1 h
Q2/2017	2.4 h
Q3/2017	2.7 h
Q4/2017	1.2 h
Q1/2018	0.7 h

LOVOO's Antispam Transparency Report

https://tech.lovoo.com/2018/06/04/seventh-transparency-report/

RECAP AND CONCLUSION

- Spammers are bad, data is good
- A combination of ML and heuristics help us in the fight
- Google Cloud services such as ML Engine, BigQuery assists us
- Future work: Conviction, more image processing and anti-antispam detector

THANKS

Say no to spam!

- **Twitter: @jdiossantos**
- **LOVOO Engineering @lovooeng**
- **Goka Repo: https://github.com/lovoo/goka**

Juan De Dios Santos Rivera (@jdiossantos)
Software Engineer Big Data @ LOVOO
November 19, 2018 PyData Warsaw 2018

LOVOO

TRACER'S ACTIONS

- Sequences have a max length of 500 actions
- Since we try to keep the Kafka messages below 500kb, storing these sequences is expensive.
- After each prediction the sequence array is emptied