Learning Cheap and Novel Flight Itineraries

Dima Karamshuk
Skyscanner

AI UKRAINE
V INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE AND DATA SCIENCE APPLICATIONS
18

Planning a Trip

How much time you spend to choose a flight?

Planning a Trip

- 3.5h European travellers spend on average to find a perfect flight, often longer than the flight itself https://goo.gl/74CivT

Planning a Trip

- 3.5h European travellers spend on average to find a perfect flight, often longer than the flight itself https://goo.gl/74CivT

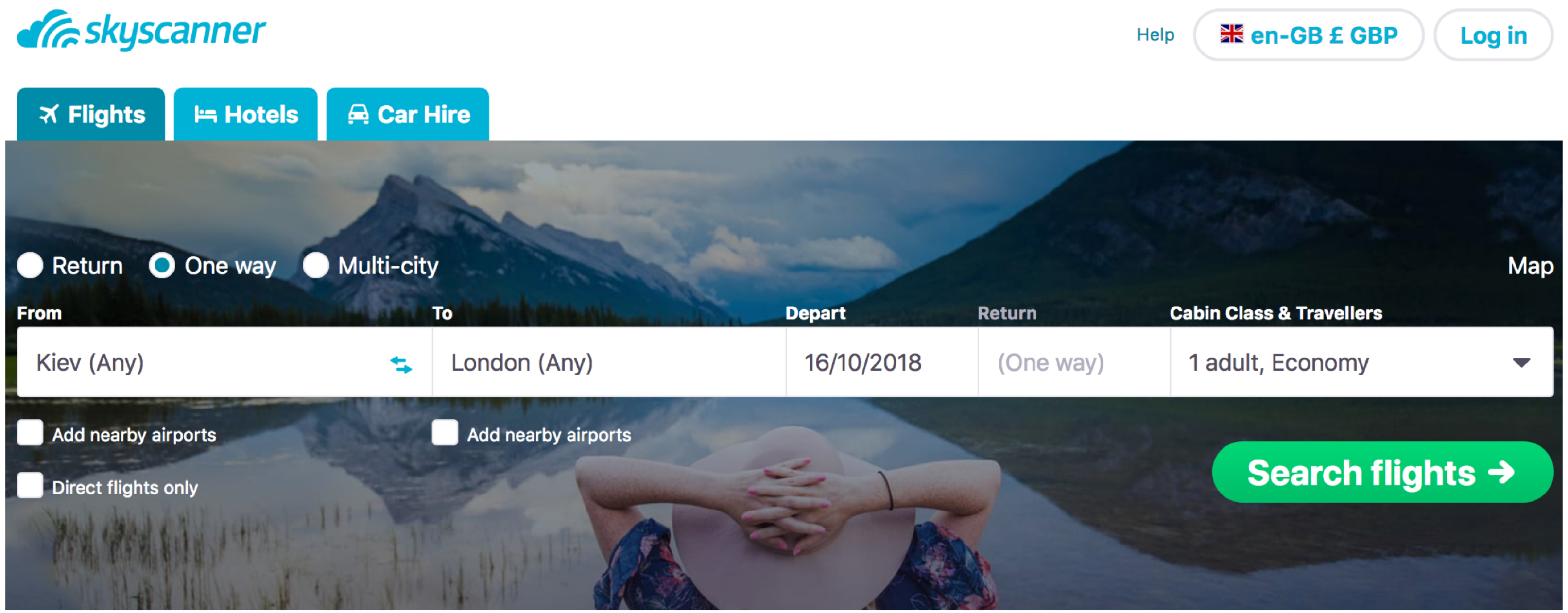
How much of you choose airline by price?

Planning a Trip

- 3.5h European travellers spend on average to find a perfect flight, often longer than the flight itself https://goo.gl/74CivT

- 37% of users choose airlines by competitive price, more want to see cheapest price for comparison https://goo.gl/8UX3vx

Skyscanner


skyscanner
Help
en-GB £ GBP
Log in
Flights
Hotels
Car Hire
Return
One way
Multi-city
Map
From
To
Depart
Return
Cabin Class & Travellers
Kiev (Any)
London (Any)
16/10/2018
(One way)
1 adult, Economy
Add nearby airports
Add nearby airports
Direct flights only
Search flights

Skyscanner in a Nutshell

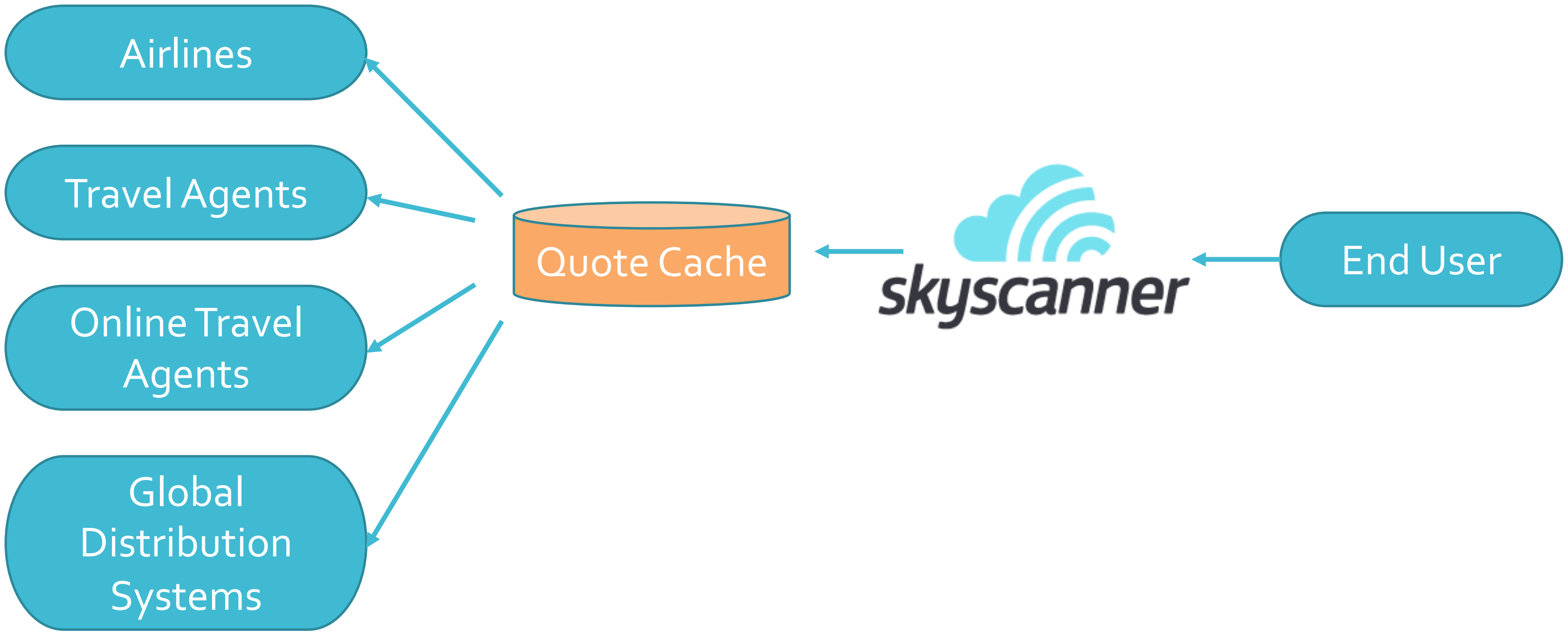
- Each user search triggers dozens/hundreds requests to partners resulting in a total of **7B/day quotes**

Skyscanner in a Nutshell

- Each user search triggers dozens/hundreds requests to partners resulting in a total of **7B/day quotes**

- Repeated requests with **85% probability** return same price

Caching Quotes

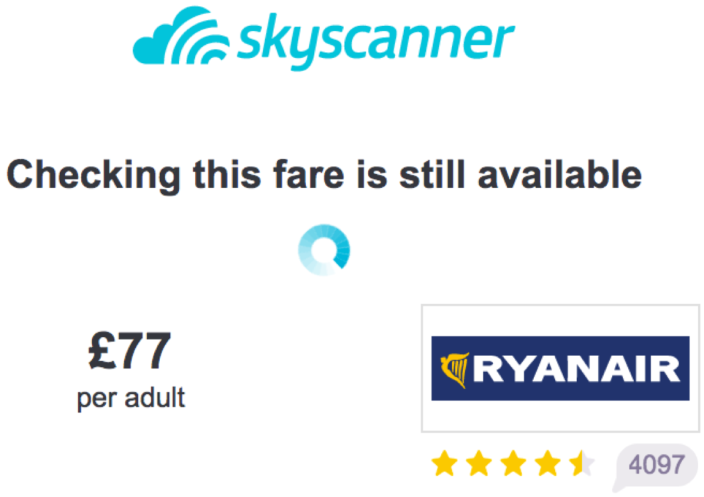


Strong case for **caching quotes**:

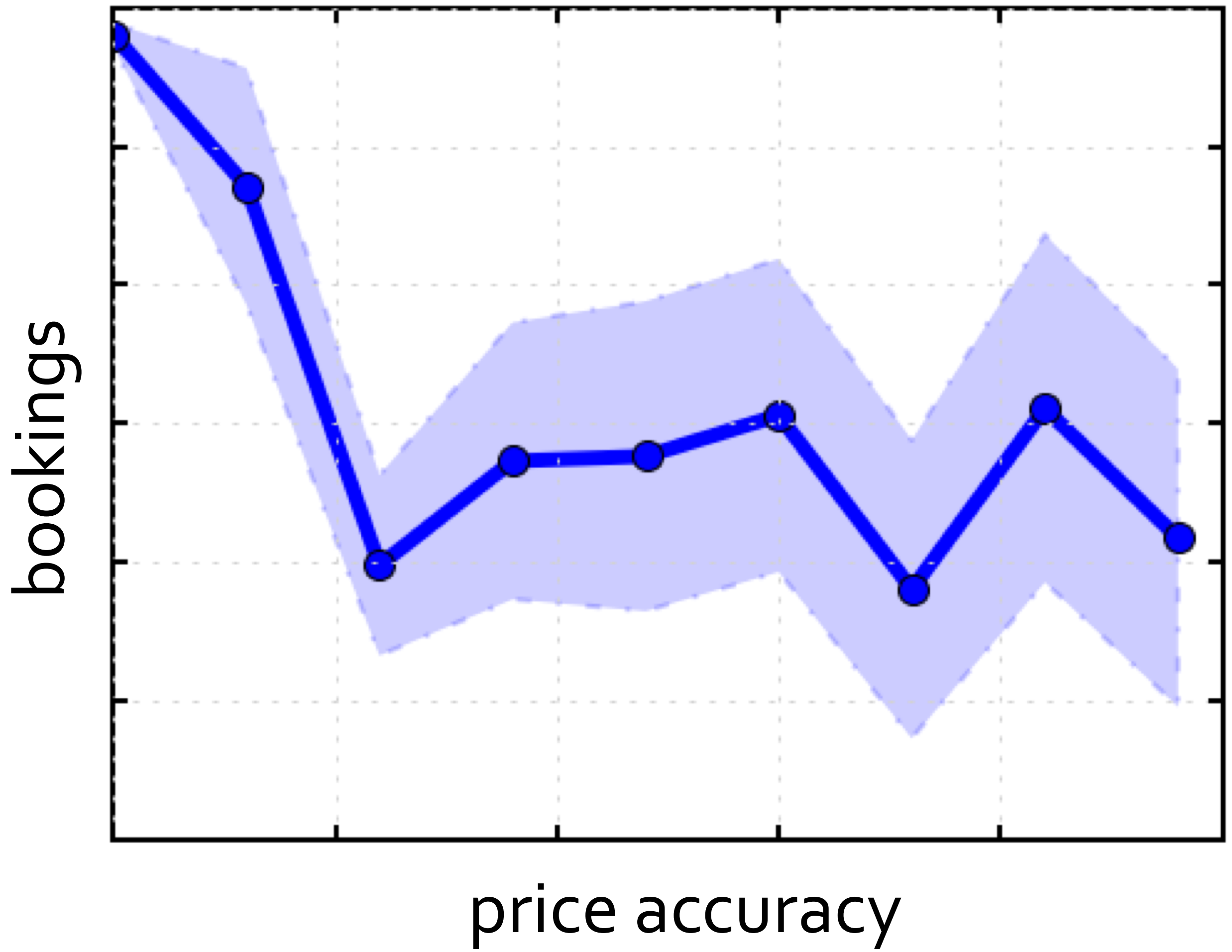
- reduced load on partners
- faster results to end users

Problem with Caching

Prices are changing dynamically, so, caching may **introduce inaccuracies**



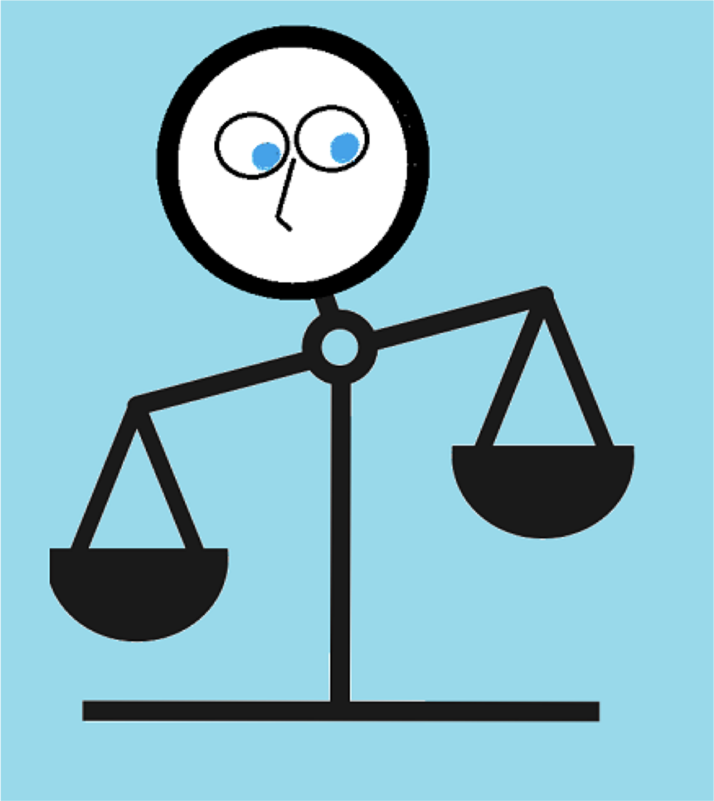
Analysis



<u>Bookings drop</u> significantly even if the prices are slightly inaccurate

Caching Trade-off

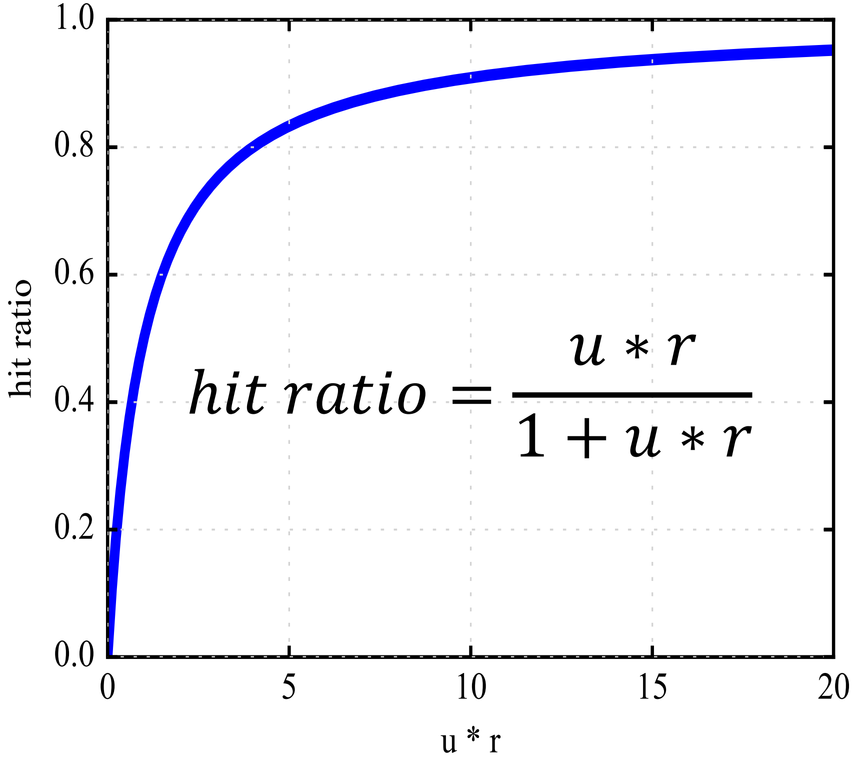
Accuracy of Quotes



Load on Partners
and Response Time

Optimal trade-off: Update prices **<u>only/always</u>** when they change

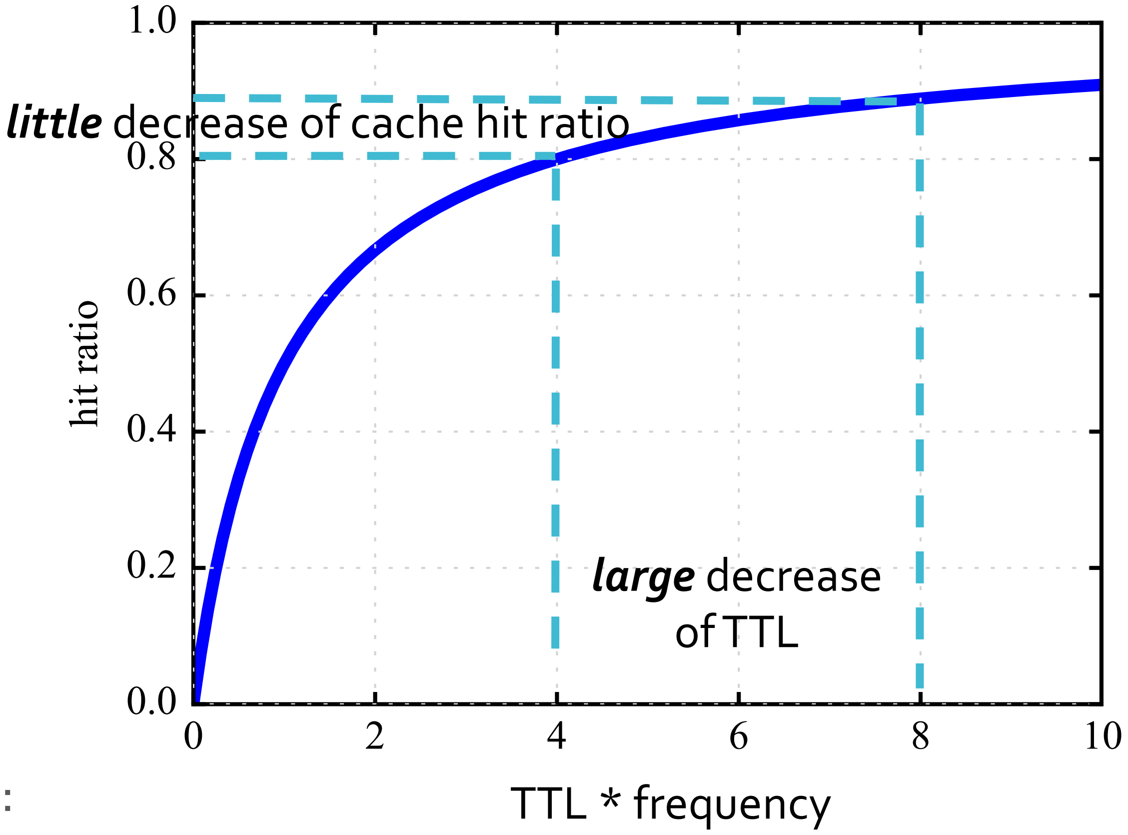
Erlang’s Loss Model



- *u* – TTL of the cached quote
- *r* – frequency of requests

assuming “memoryless” Poission arrivals

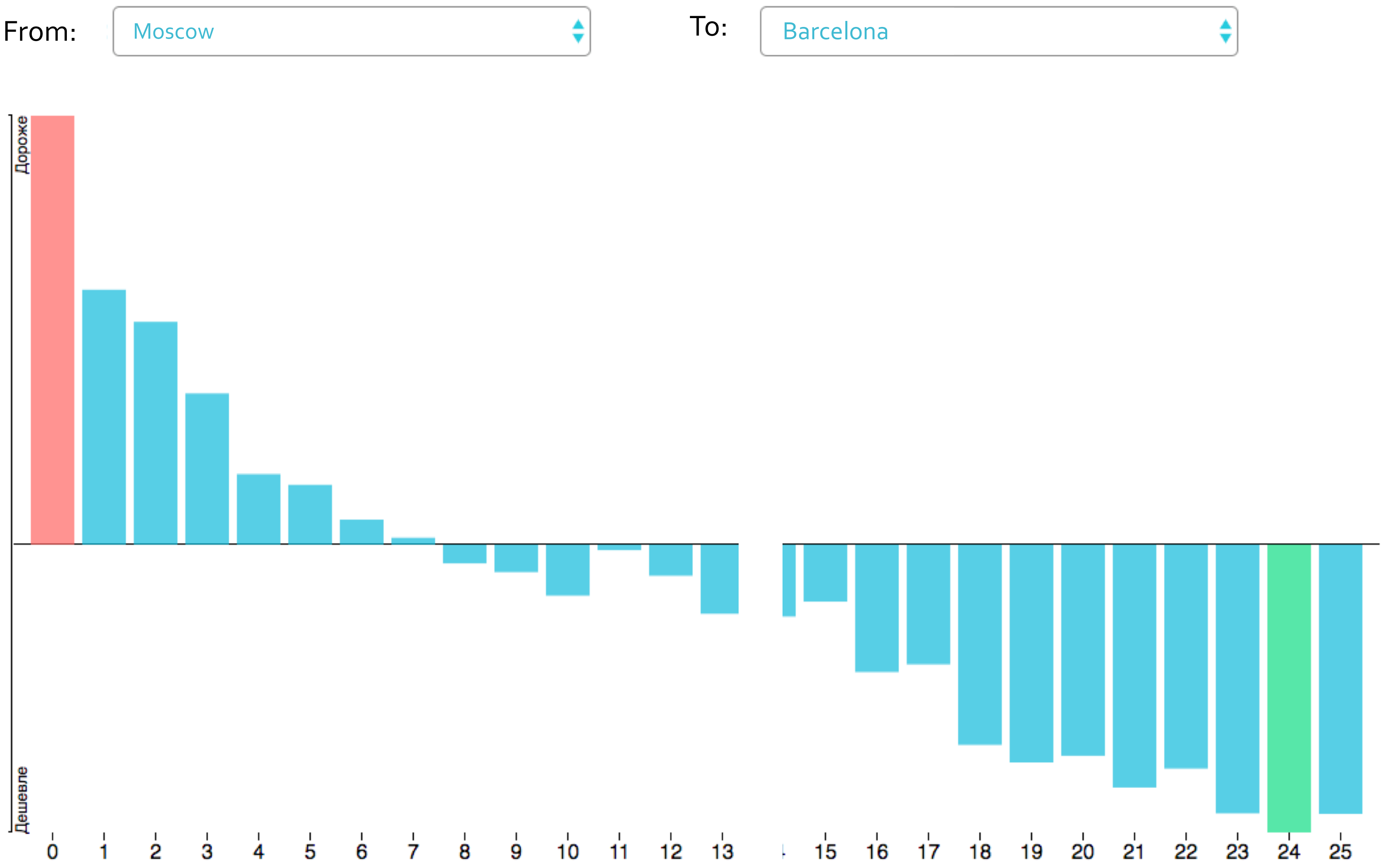
Simple Strategy



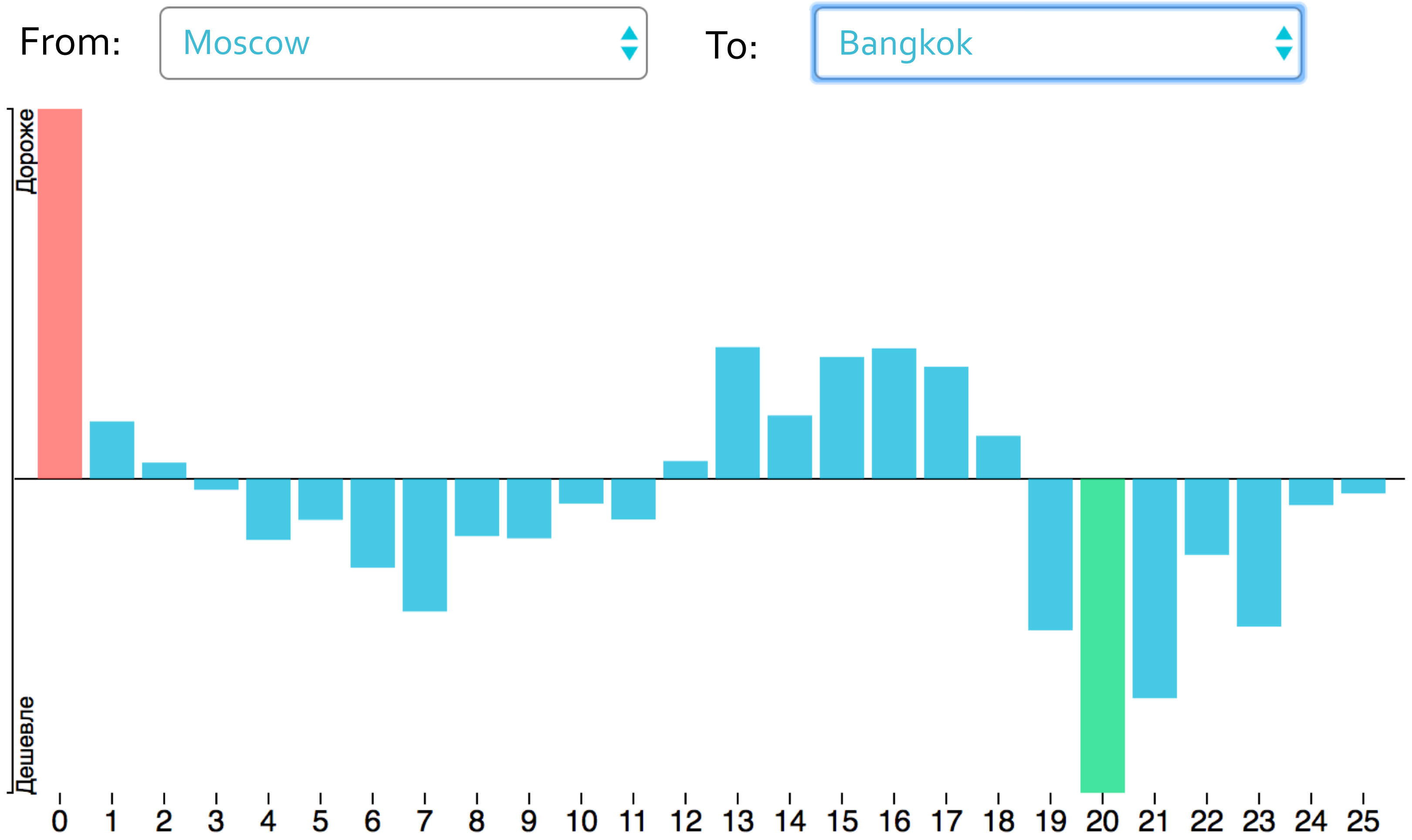
Example:

- u = 8h, r = 1/h, *hit ratio* = 88%
- if we decrease *TTL* by half (u = 4h) => *hit ratio* will decrease by only 8%
- at the same time we will <u>decrease</u> (*by half ?)* the <u>average age</u> of cached quotes served to users

Price Volatility Not Easy



Price Volatility Not Easy



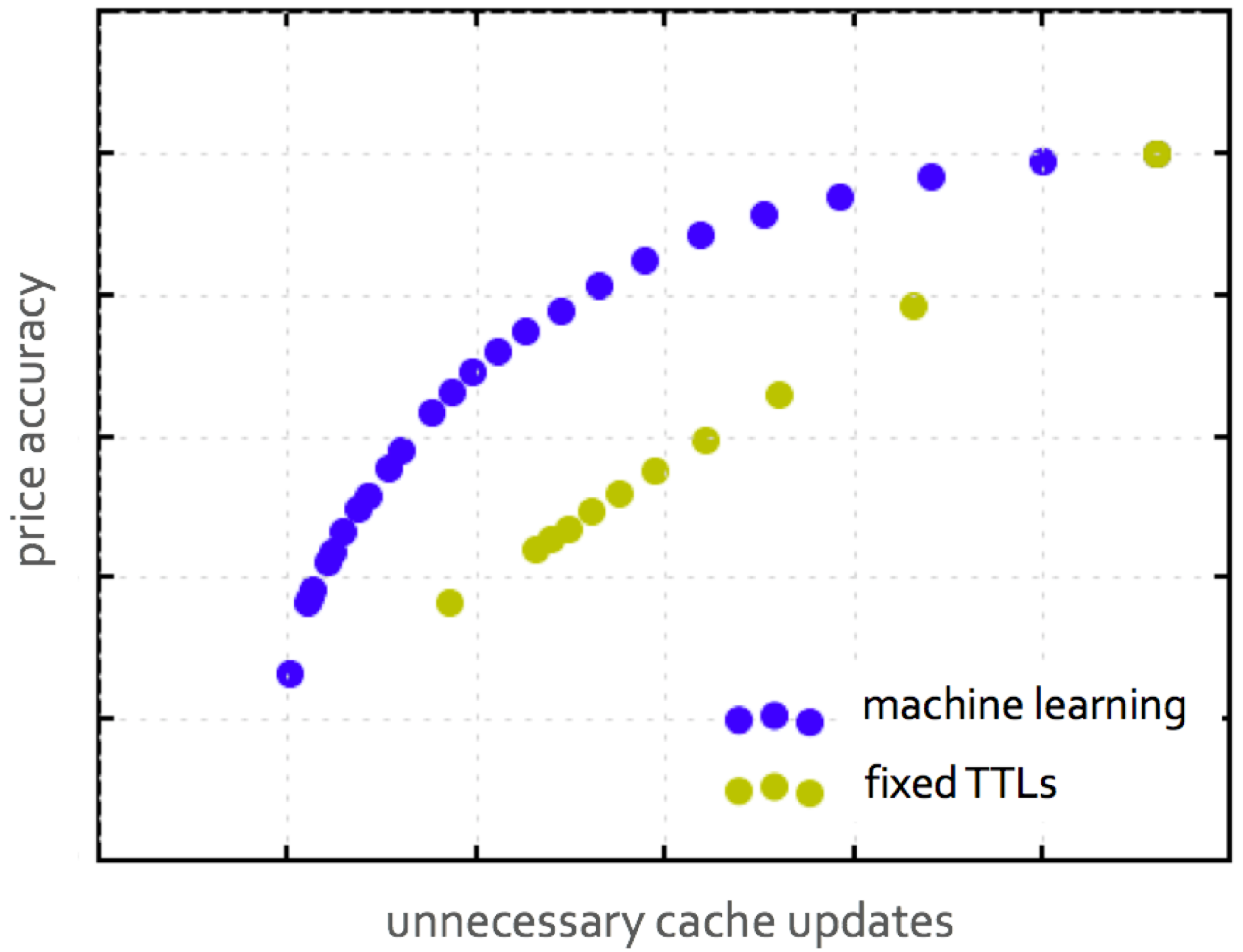
Predicting Price Volatility

1. Approach N1: **constant** cache expiry times
   - simple to implement
   - does not accurately model price volatility

2. Approach N2: **emulate pricing models** of each individual partner
   - pricing models of some airlines are incredibly complex

3. Approach N3: **machine learning** approach
   - best trade-off between simplicity and accuracy

Model Performance



Data Science

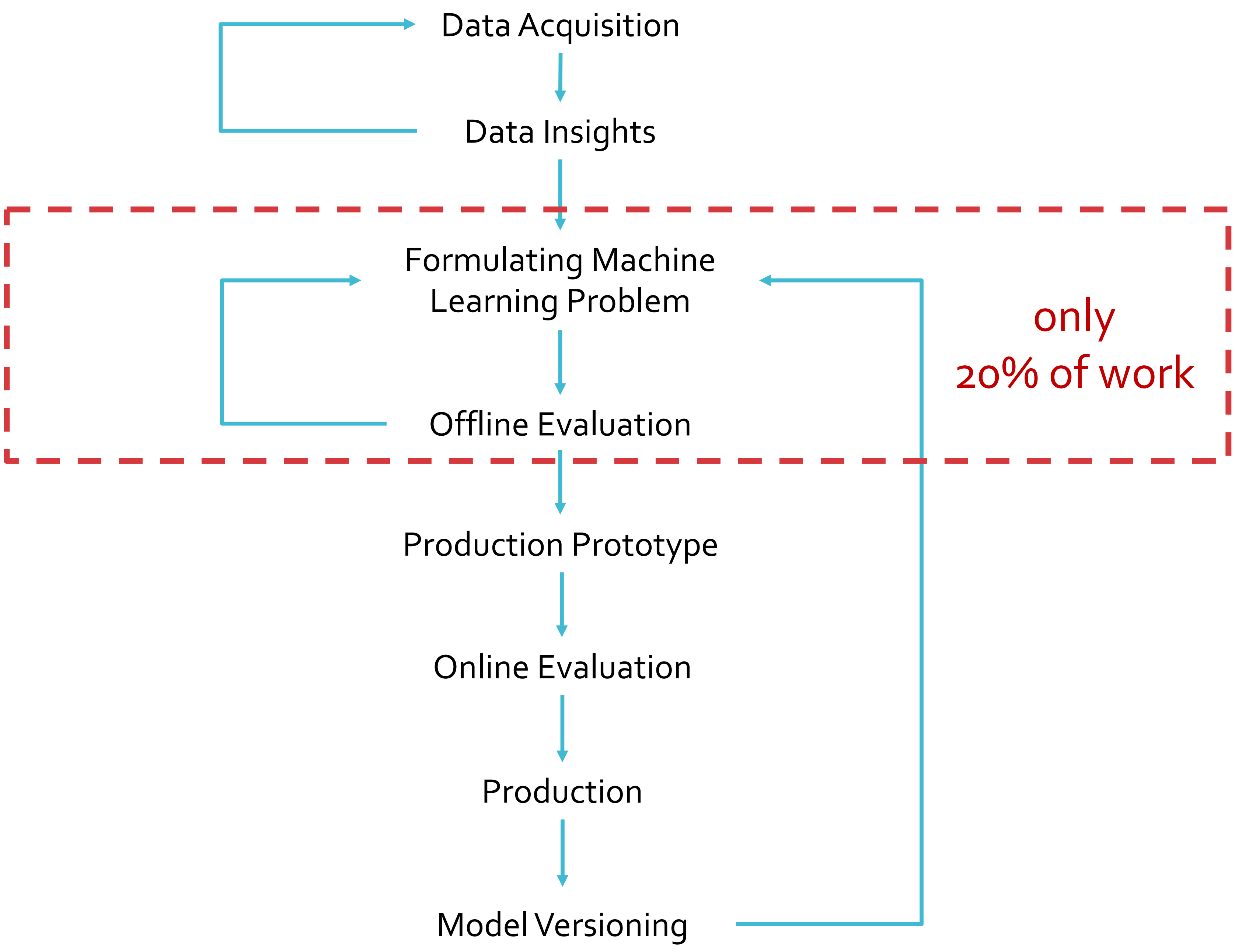
Product Cycle

Formulating Machine Learning Problem

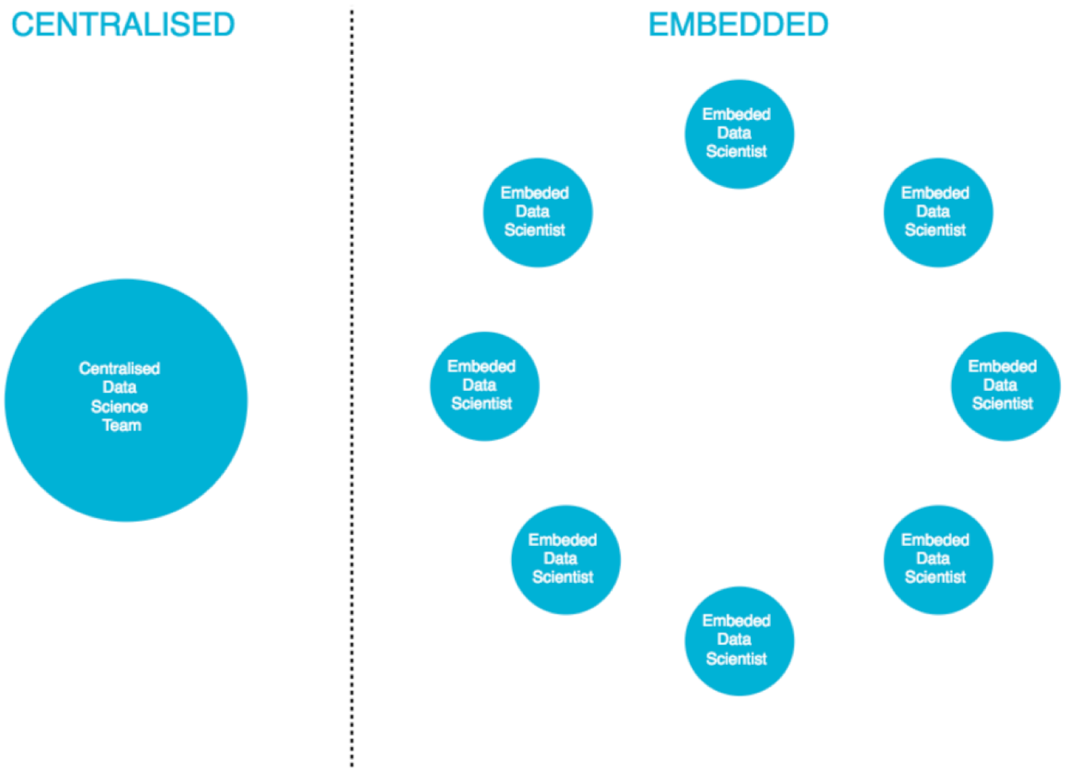
↓

Offline Evaluation

Product Cycle



Data Science Structure



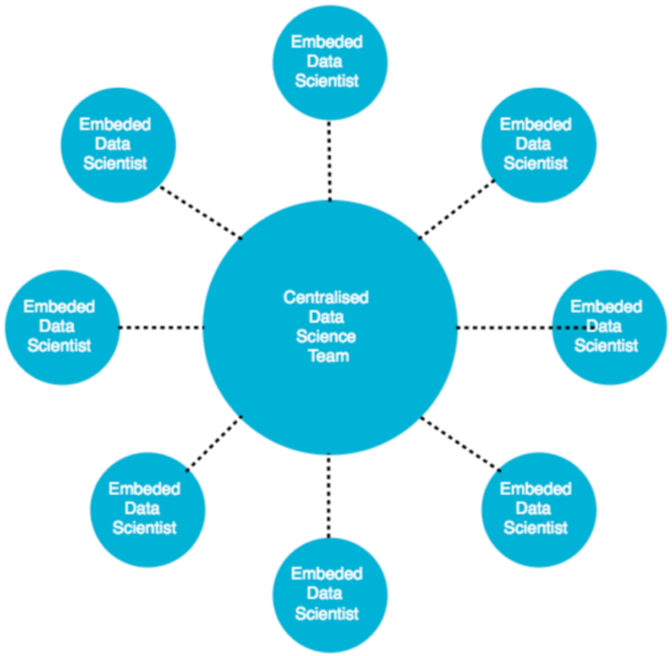
+ Great autonomy
- Risk of marginalization

+ Ensured utilization
- Lesser autonomy, focus on second-class tasks

https://goo.gl/5cdPjP

Hybrid Structures

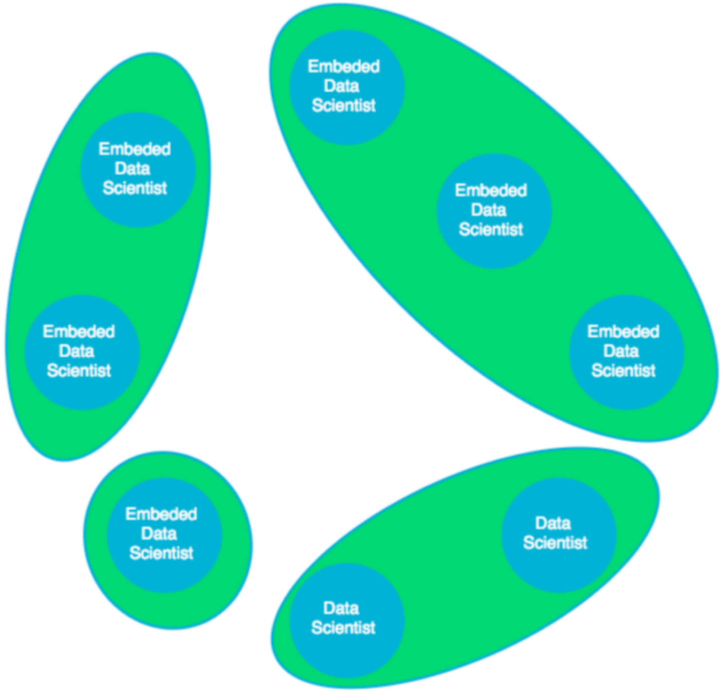
HUB & SPOKE



- part-time embedded, part-time autonomous

https://goo.gl/WJv8TR

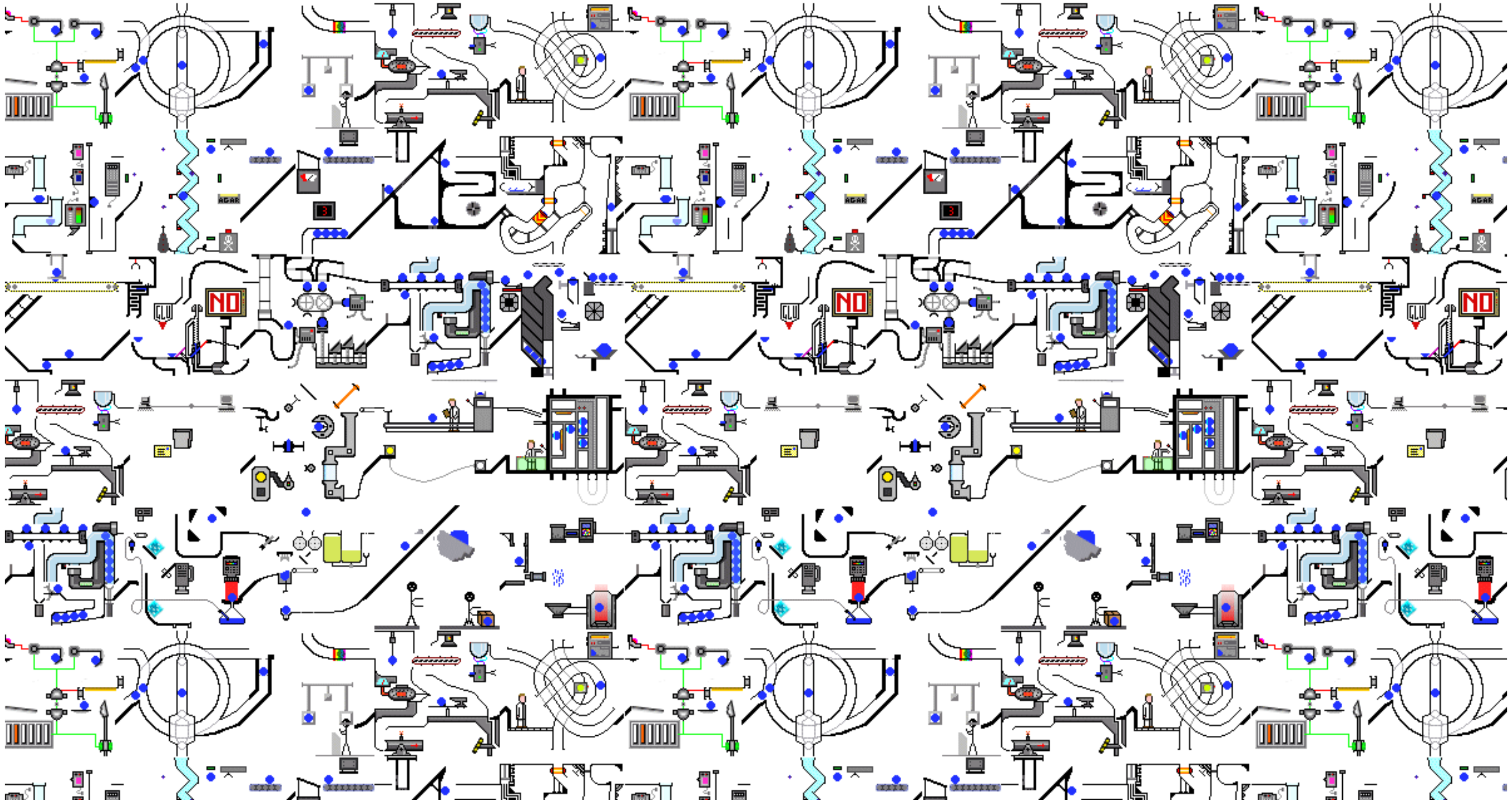
DATA SCIENCE CLUSTERS



- clusters of embedded data scientists focused on the same goal

https://goo.gl/mtQvyn

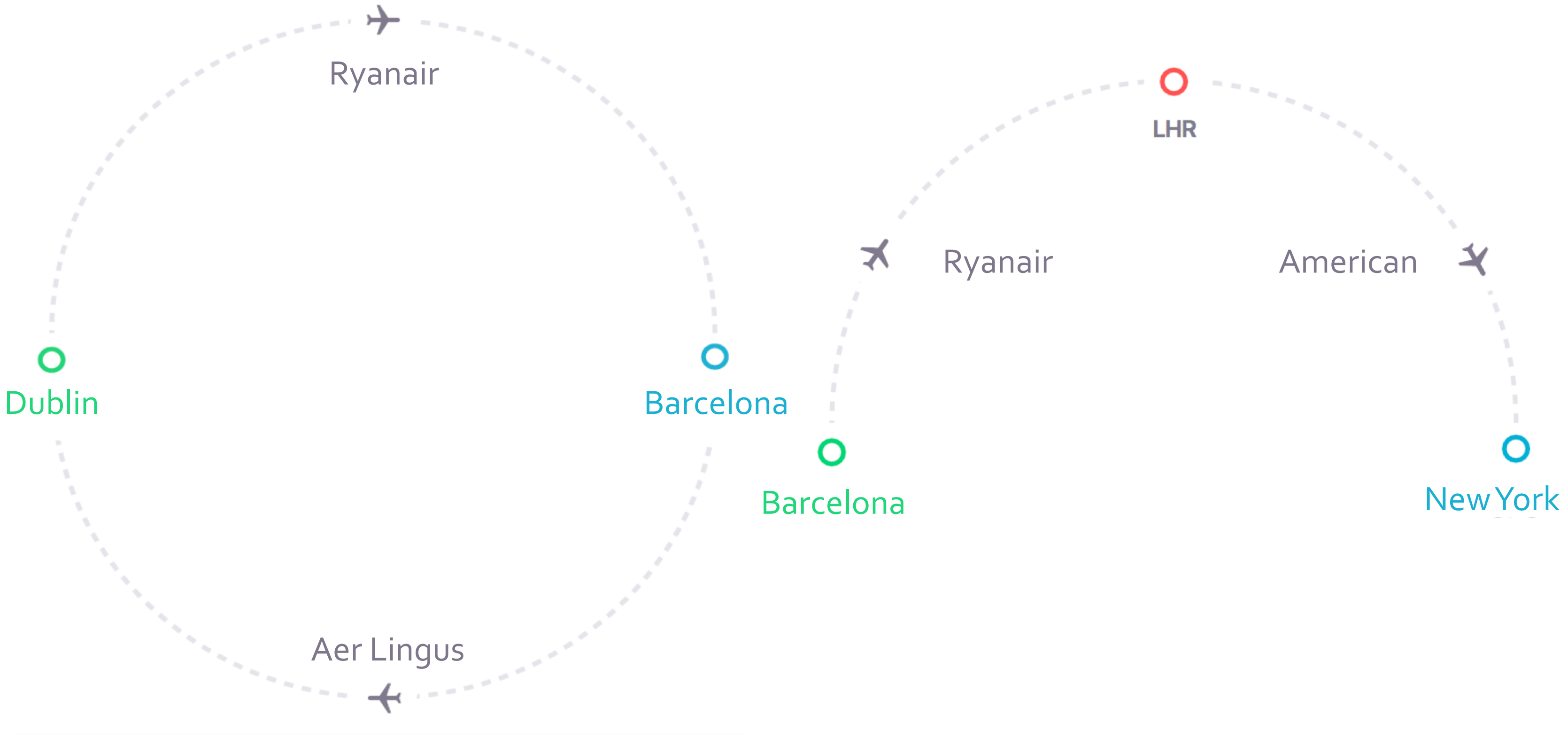
New vs. Optimizing Old Features



- it's easier to build new ML feature than optimizing what works OK already

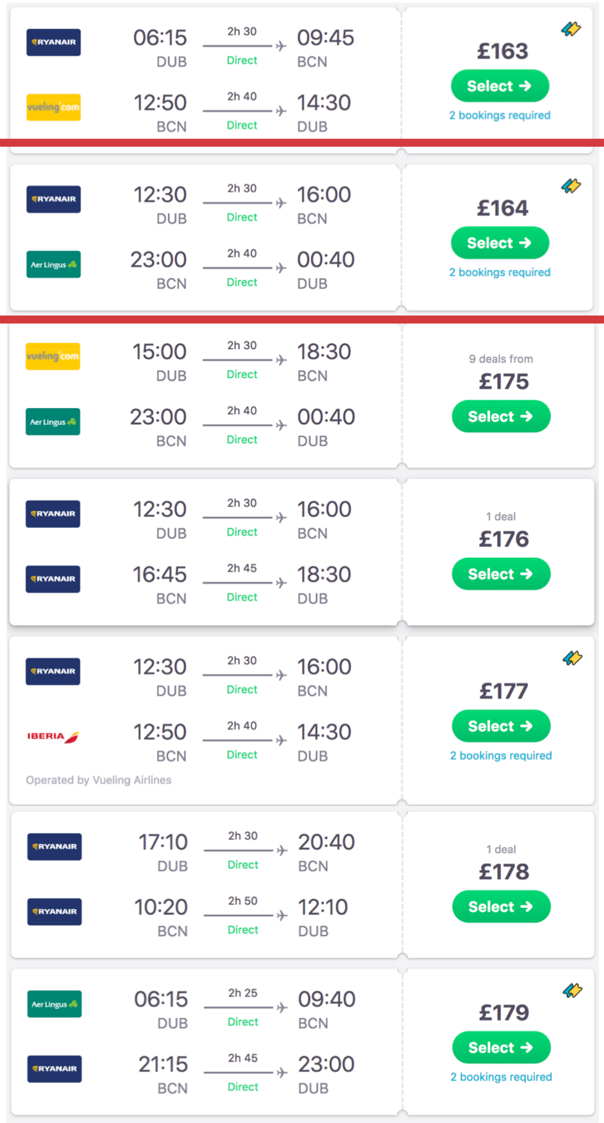
Second Try: Constructed Itineraries

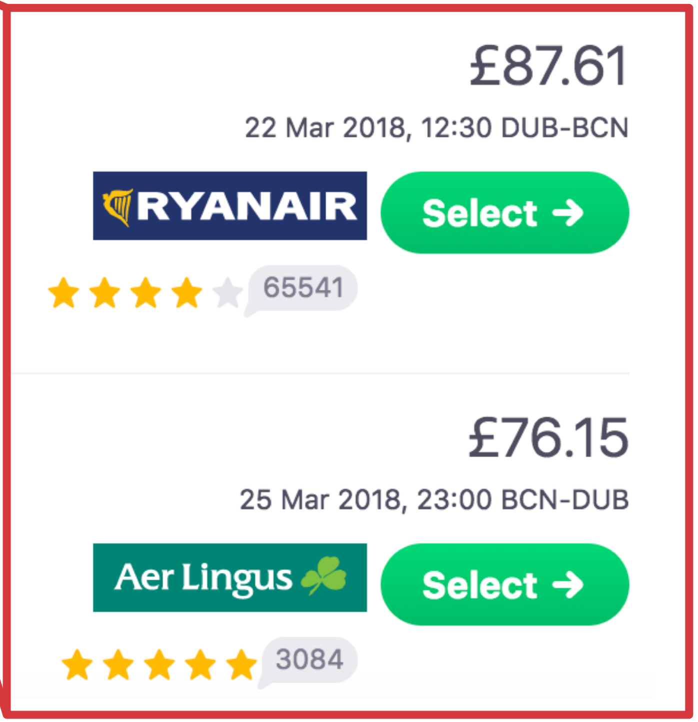
Constructing mixed-carrier itineraries



Competitive Itineraries

Competitive Itineraries are the ones in the **Top-10 cheapest** search results

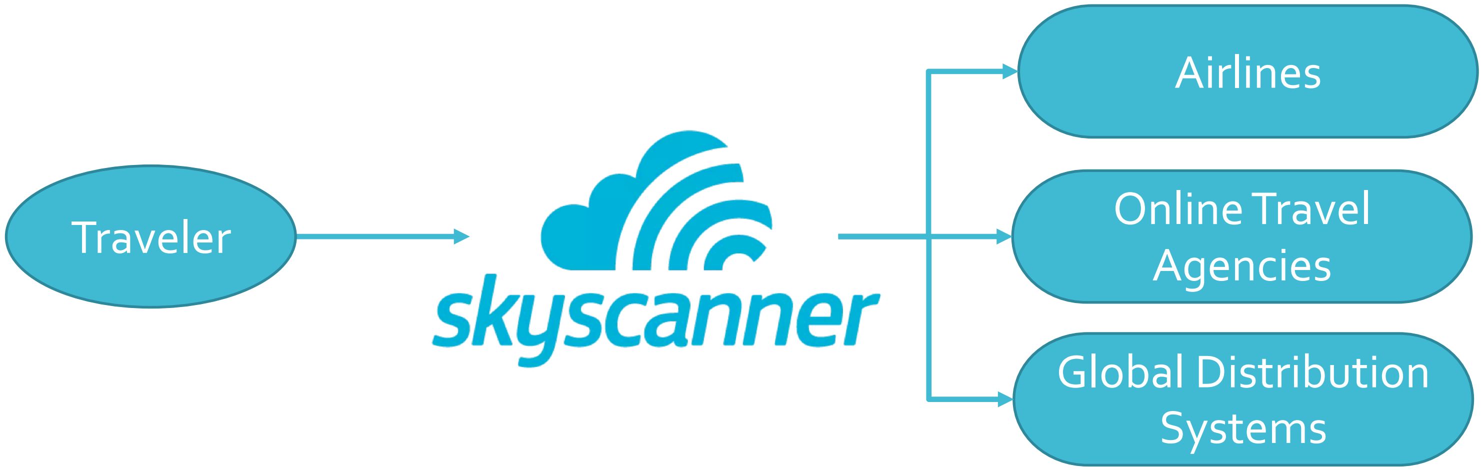




Potentially cheaper itineraries in more than half of all search results

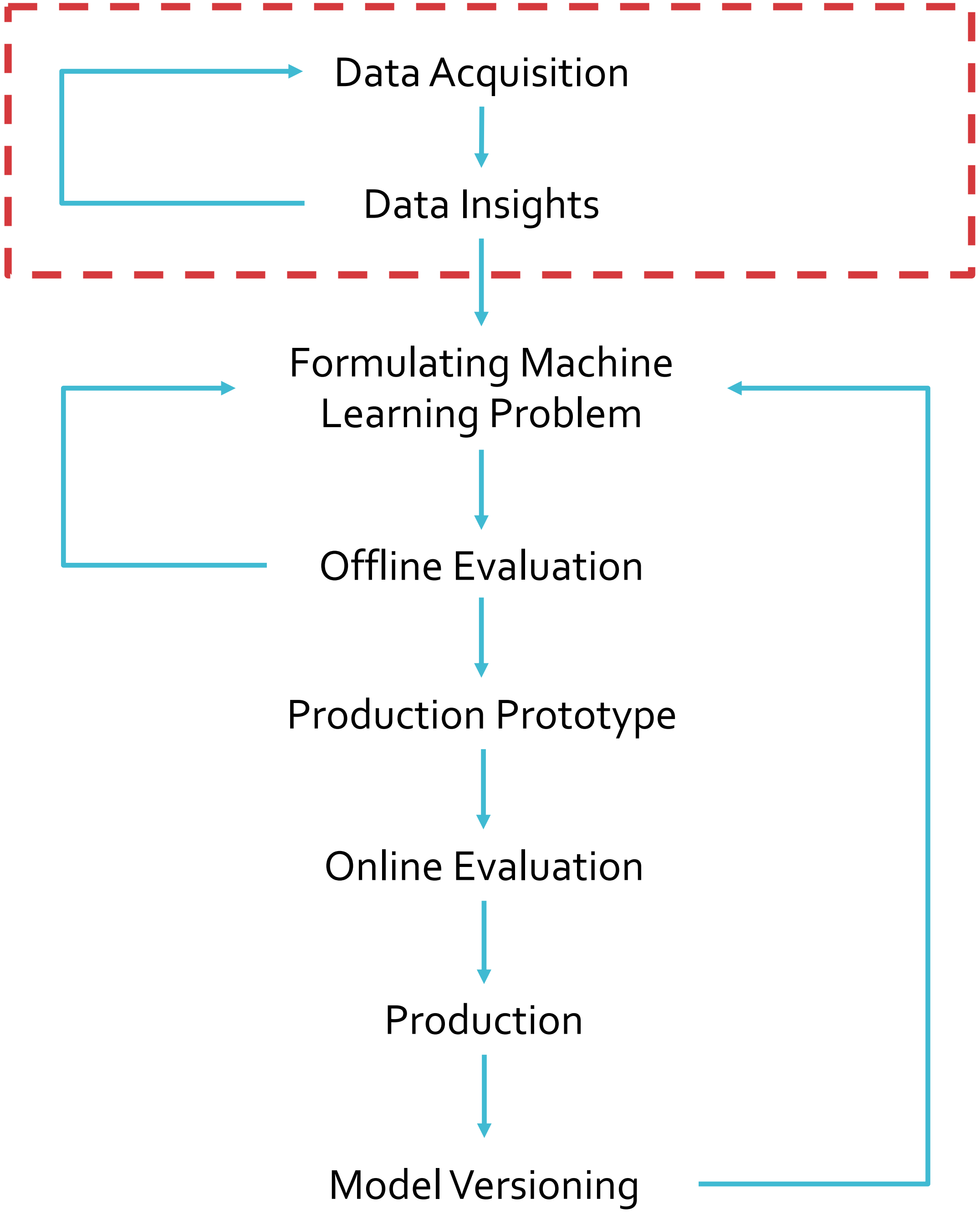
Problem

- Combinations require more queries to ticket providers
- Most of variants are not competitive

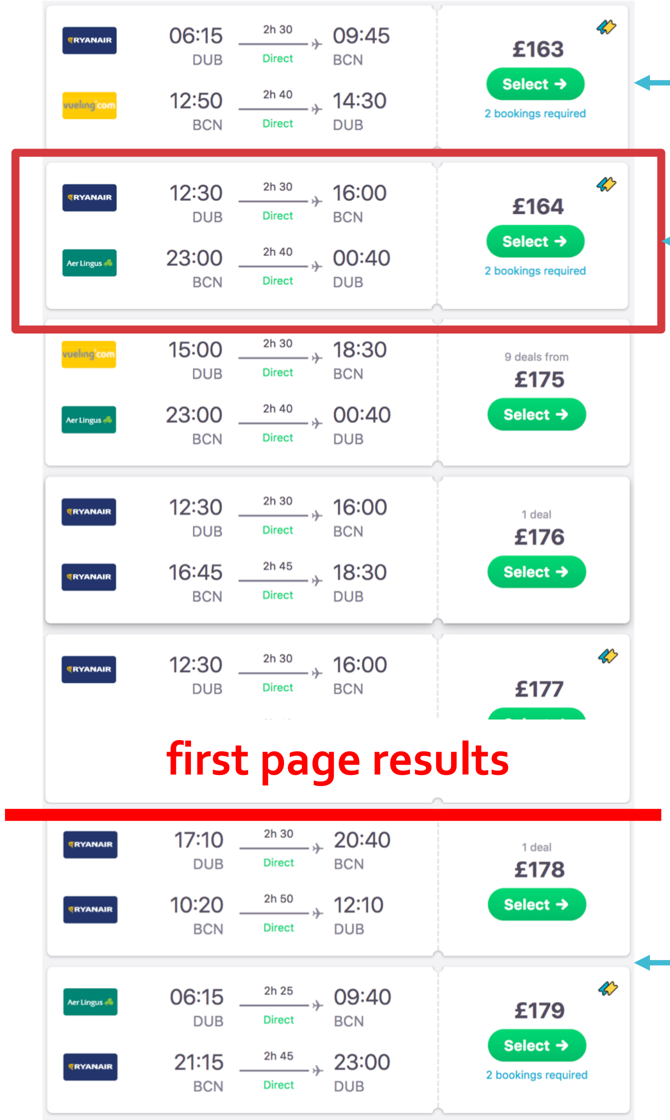


Solution: Only choose combinations which are likely to be competitive

Product Cycle



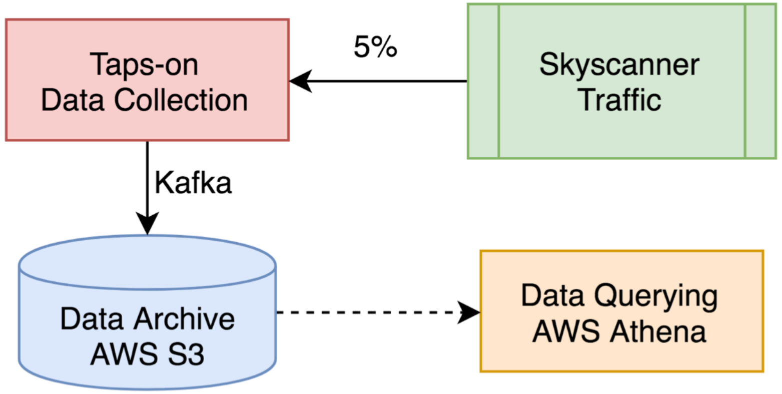
Logging



what users click on

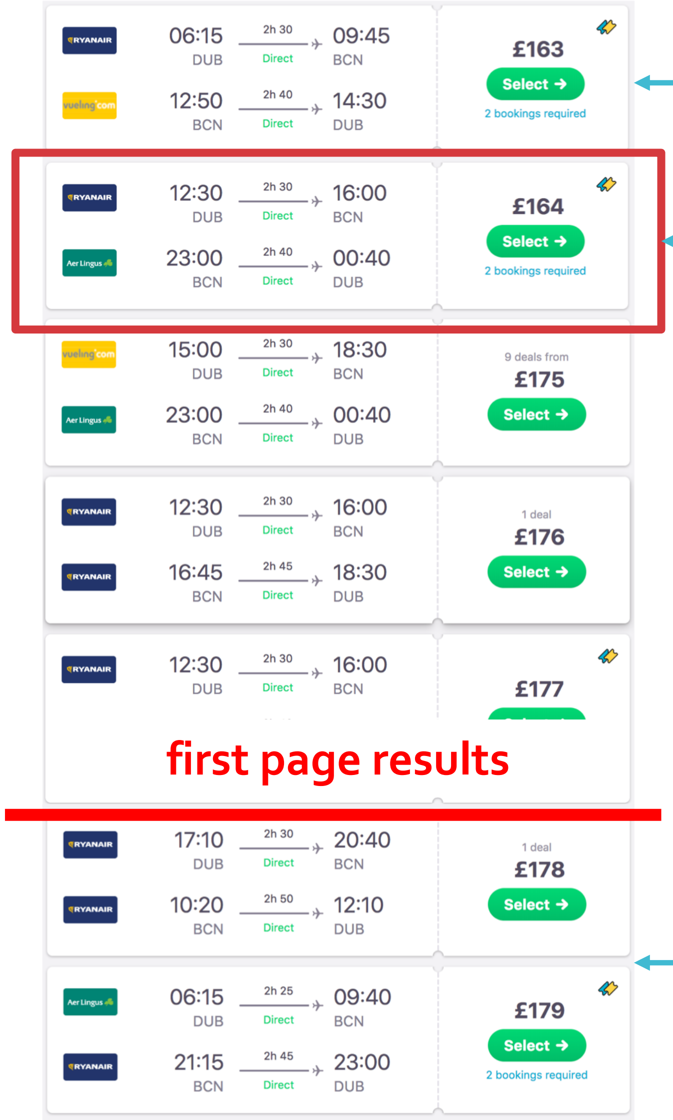
what users see but do not click on

what users do not see



- it is important to log negative samples
- it is important to allow exploration

Logging



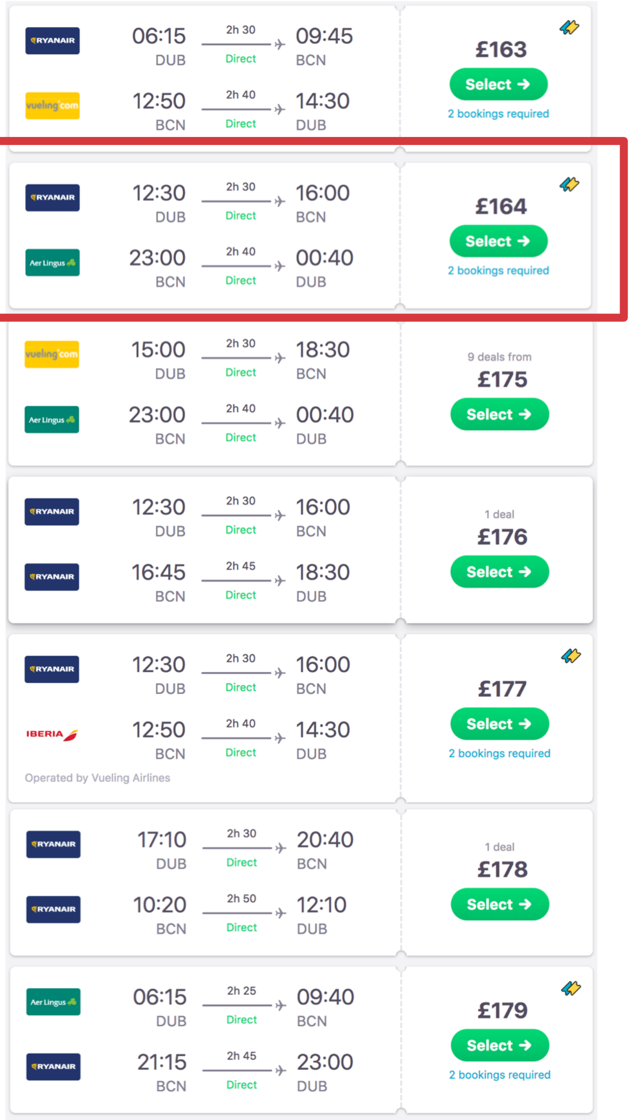
what users click on

what users see but do not click on

what users do not see

- it is important to log negative samples
- it is important to allow exploration
- non-trivial data pre-processing ETL jobs are needed
- along with robust querying interfaces (e.g., Athena)

Competitive Combinations



Tips for booking your next flight

- good for last minute booking
- average savings of 9% on return ticket
- 90% of competitive combinations are from top-30% airlines
- good deals when flying from US, UK, Spain, Germany, Italy and other origins

Supervised Learning

Metrics

Coverage: How many of all possible cheap itineraries we recall

Cost: How much queries for flight quotes are required

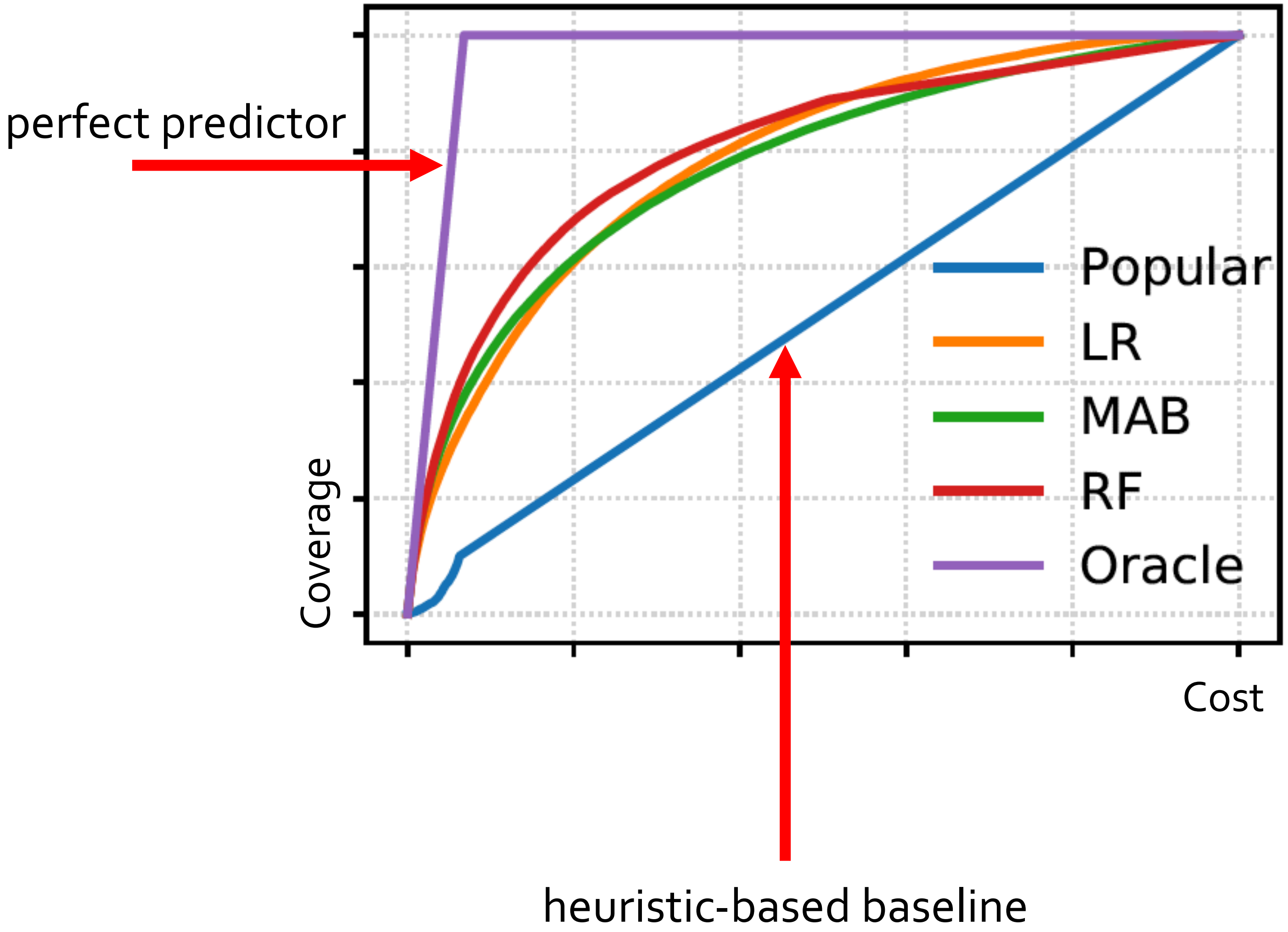
Classify whether for a query **Q** a combination of partners **(X and Y)** is going to be in **Top-10 search results**

Dataset

- sample all possible combinations for a share of searches
- collect examples of competitive and non-competitive combinations

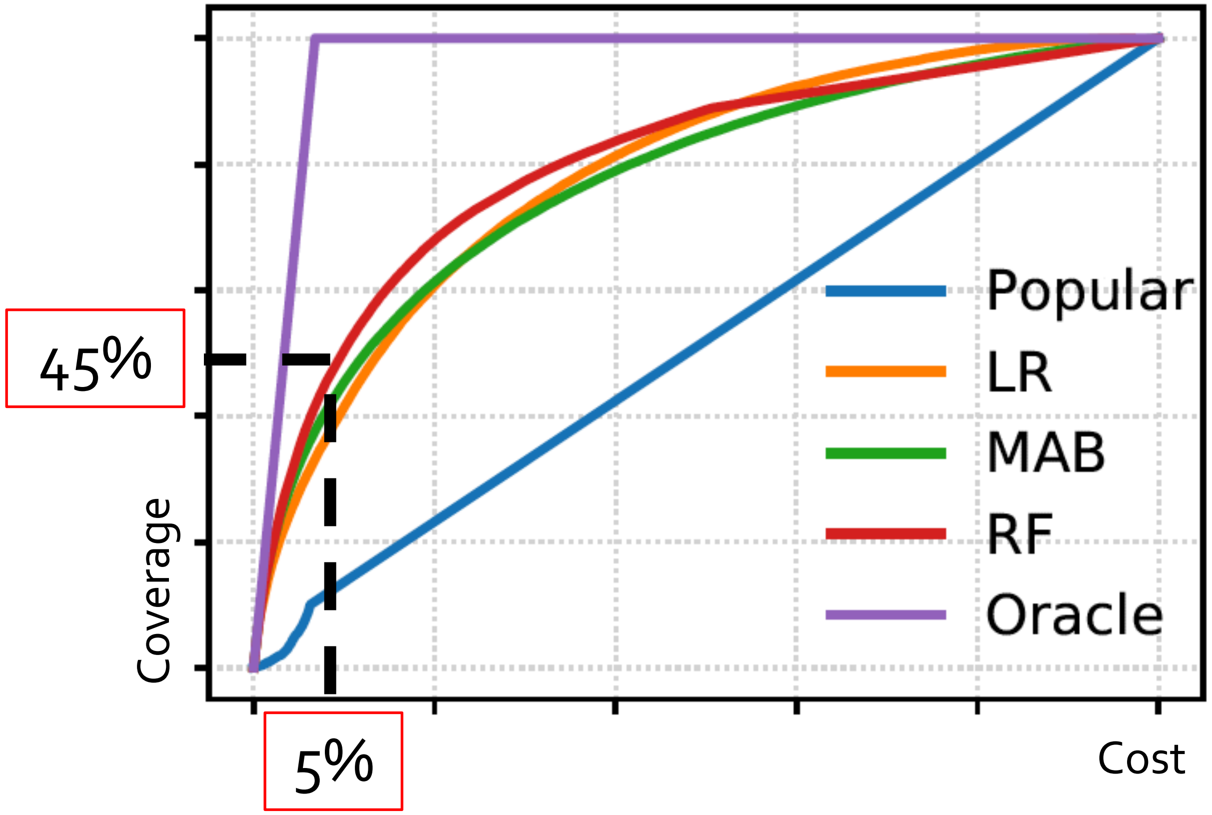
Supervised Learning

Use your favorite classifier



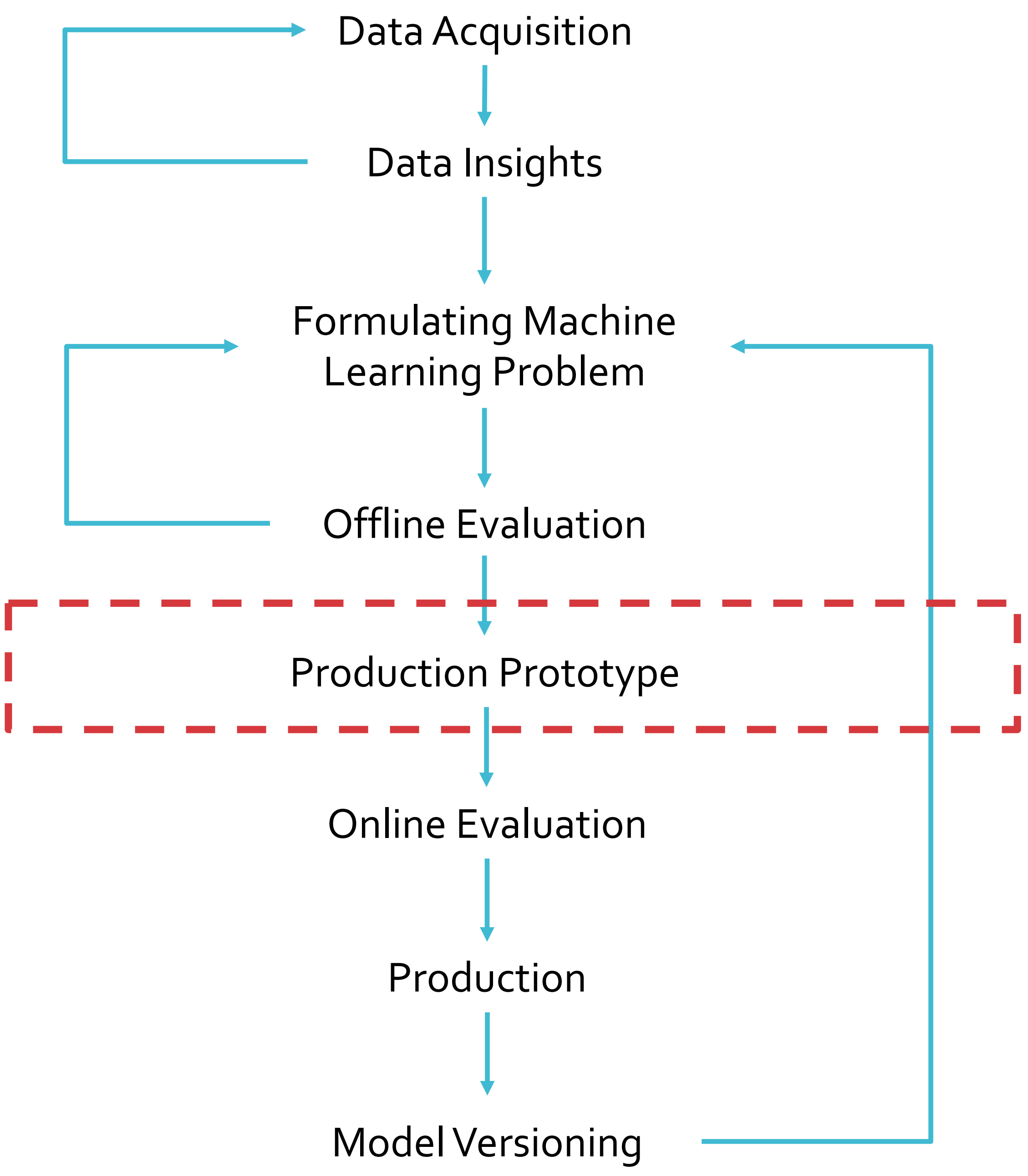
Supervised Learning

In practice

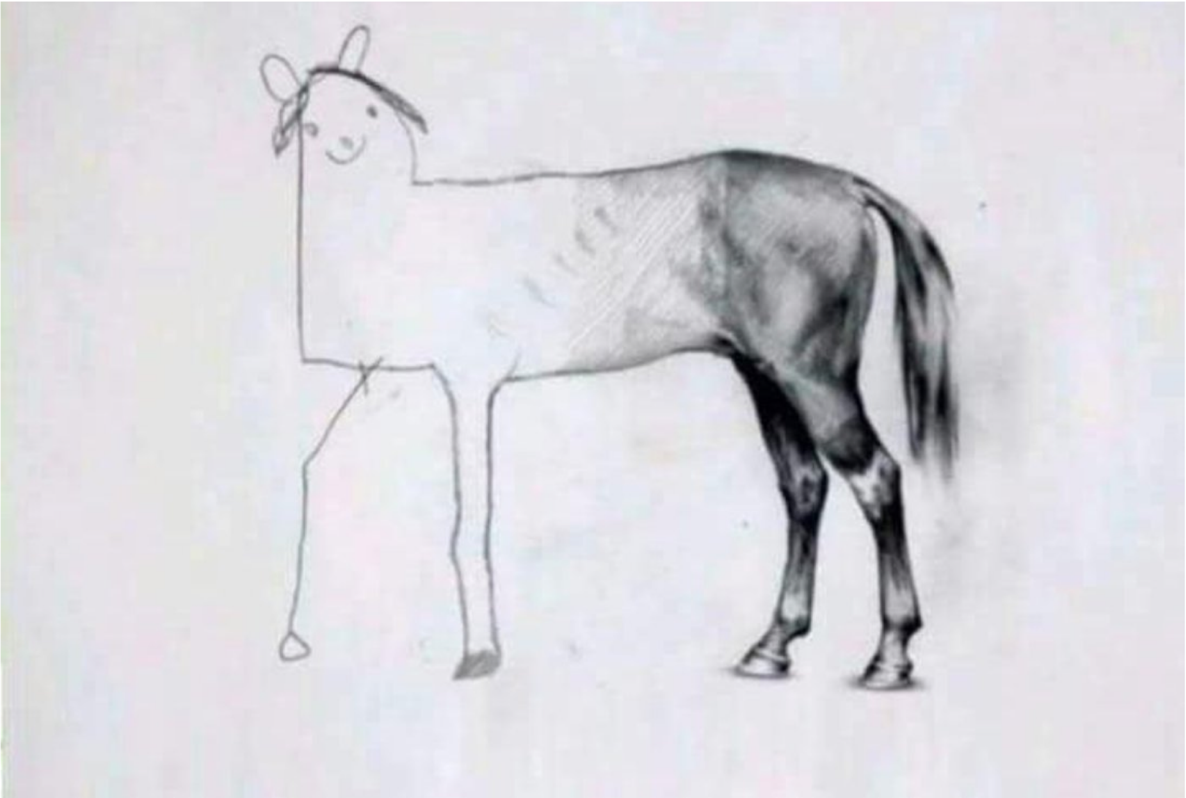


Tree ensembles (Random Forest) achieve good performs

Product Cycle

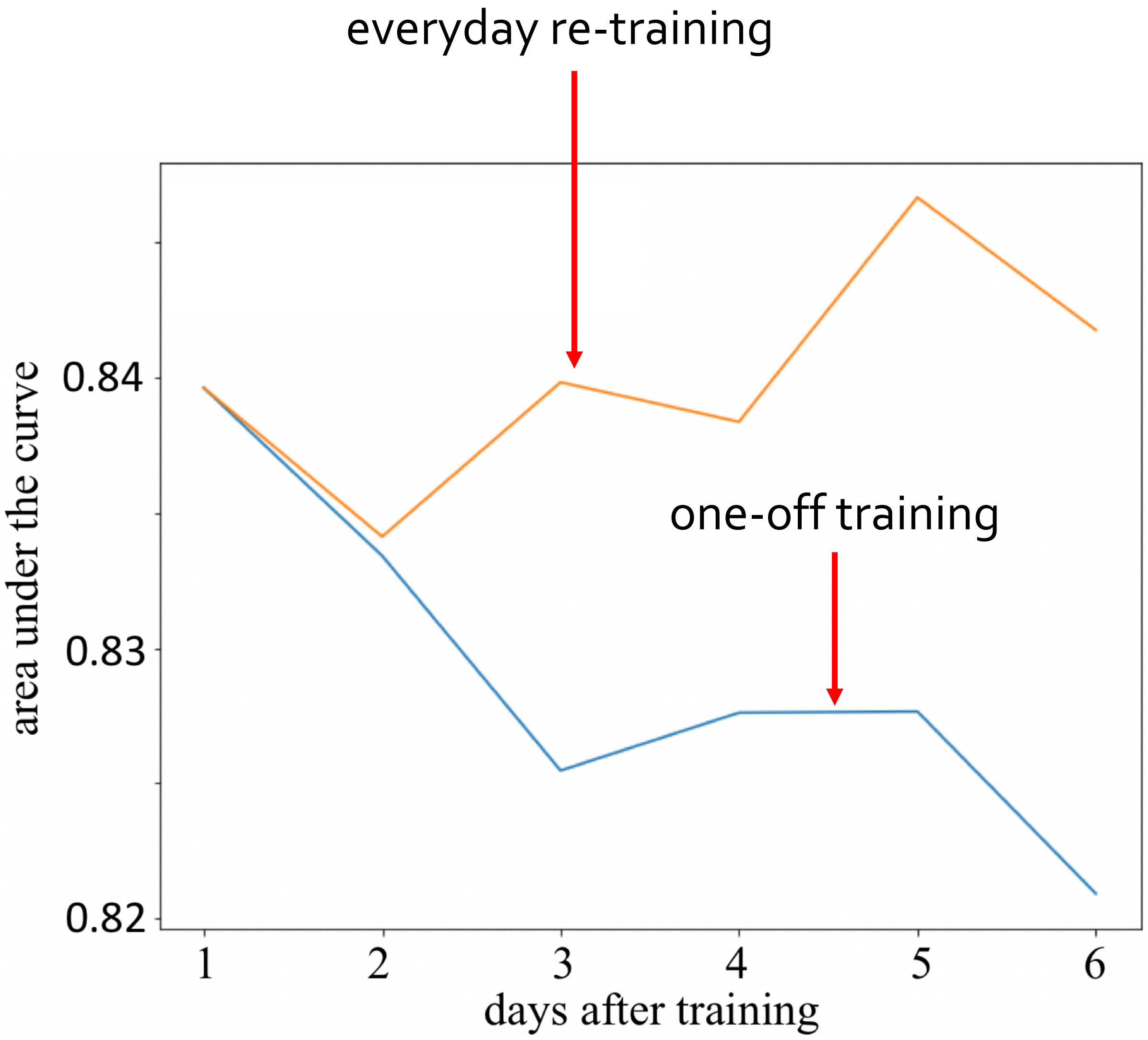


Lean Prototyping



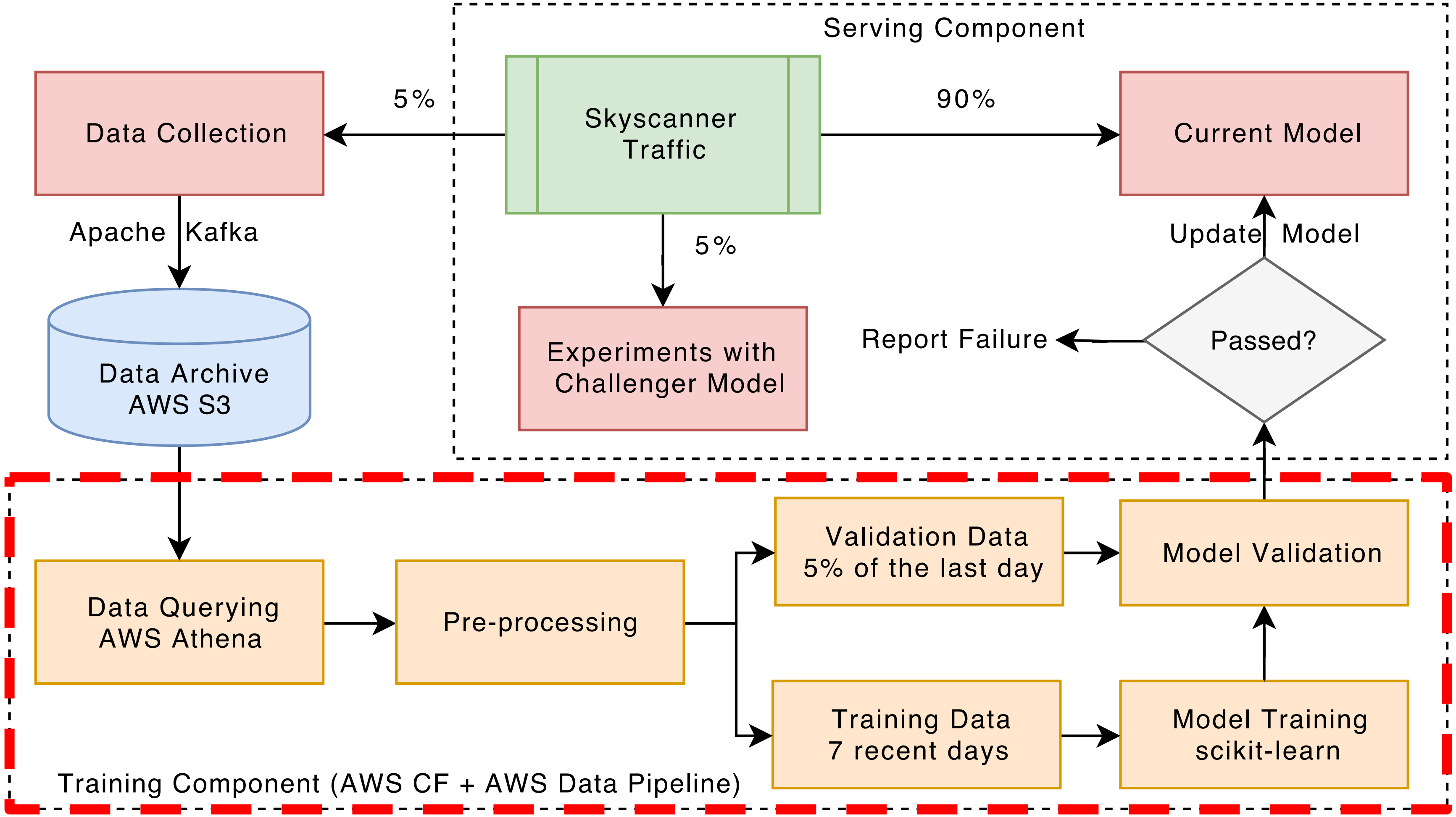
- simple model trained in a Jupyter Notebook
- very hacky setup in production on a tiny share of traffic
- proved the value of ML optimization

Model Staleness



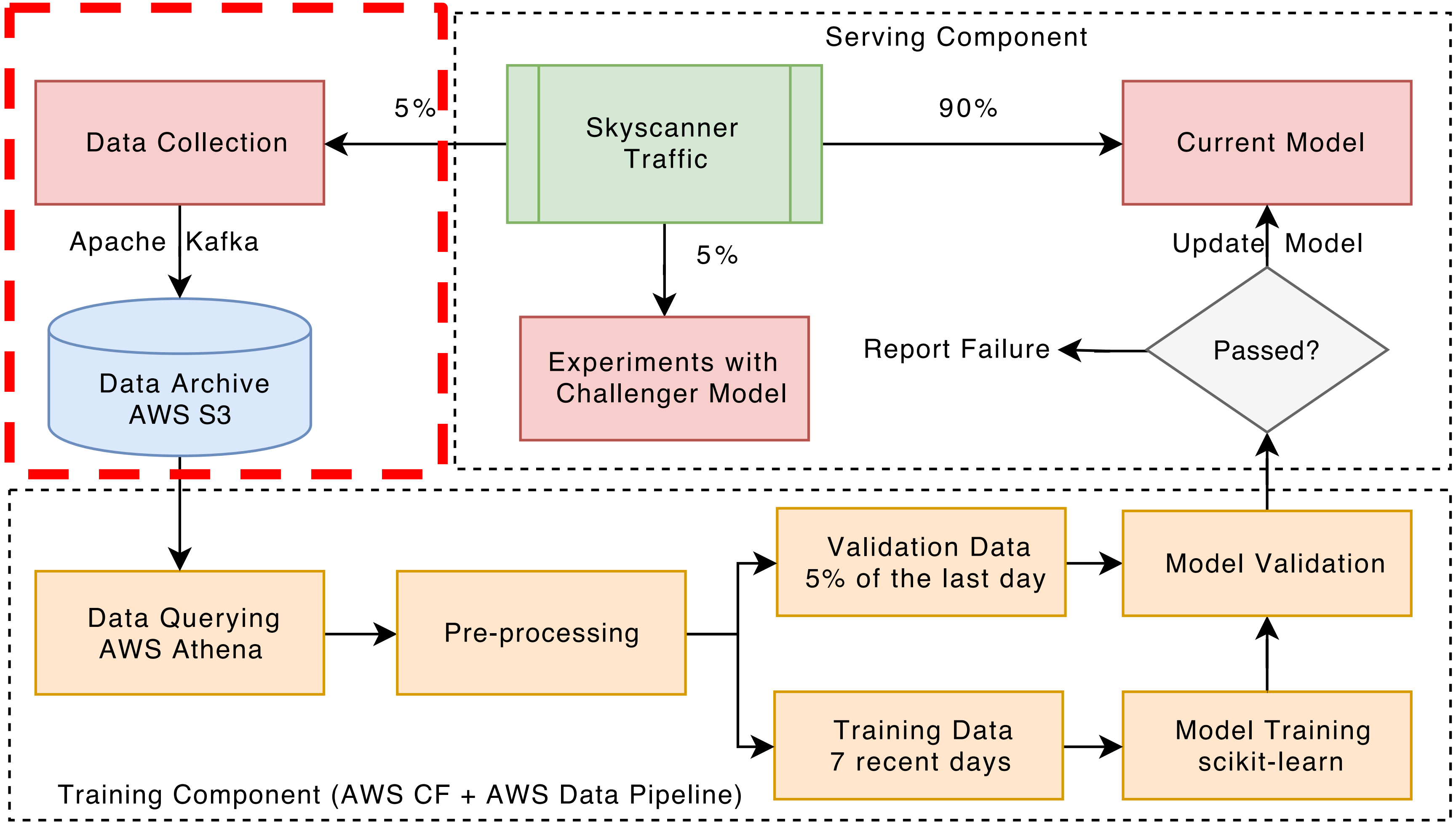
Performance of the model stales, hence needs to be updated regularly

Production Pipeline



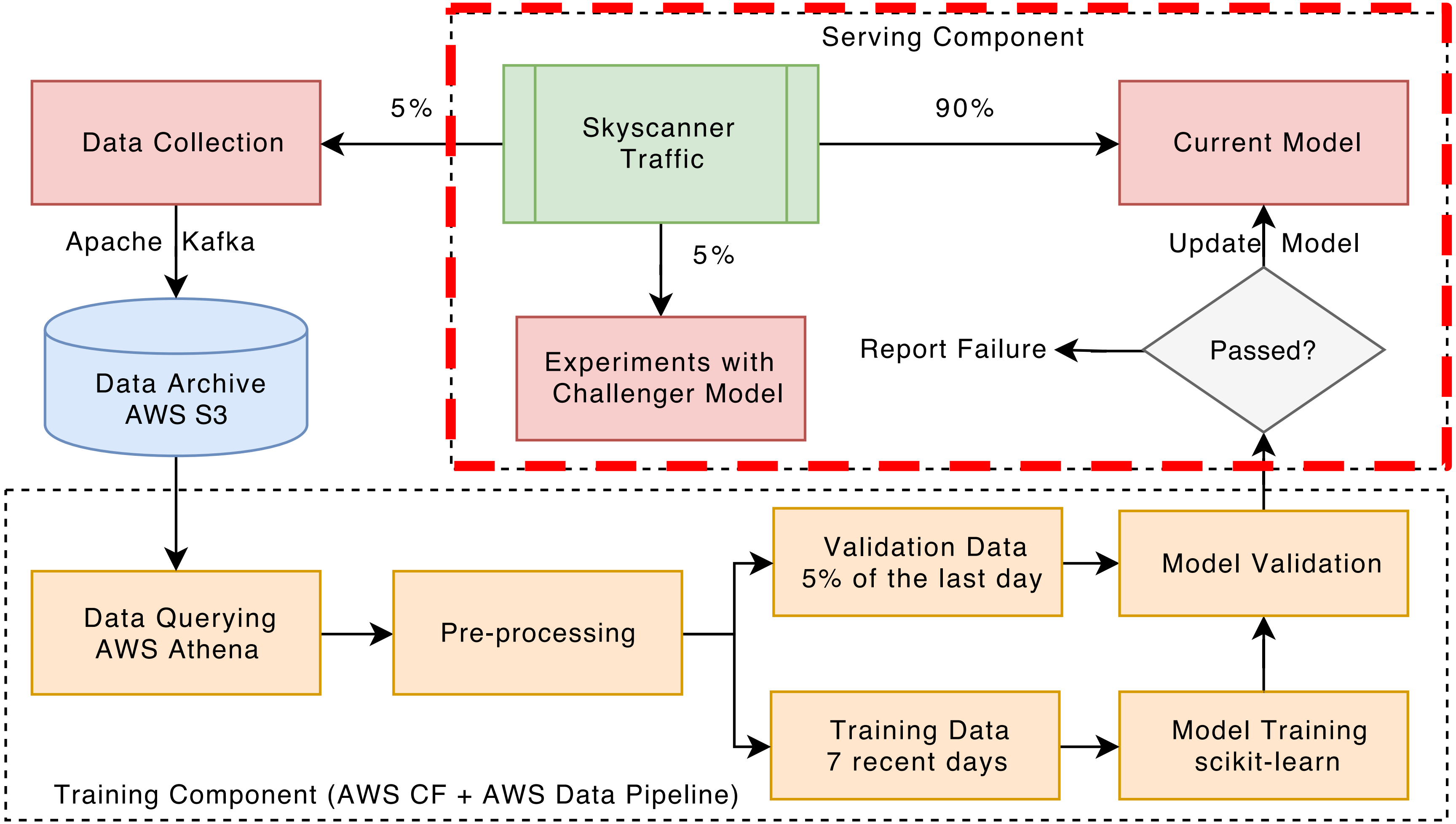
- re-train the model everyday against model drift
- run on a single large machine vs. distributed cluster

Production Pipeline



- sample all possible combinations on 5% of users' traffic

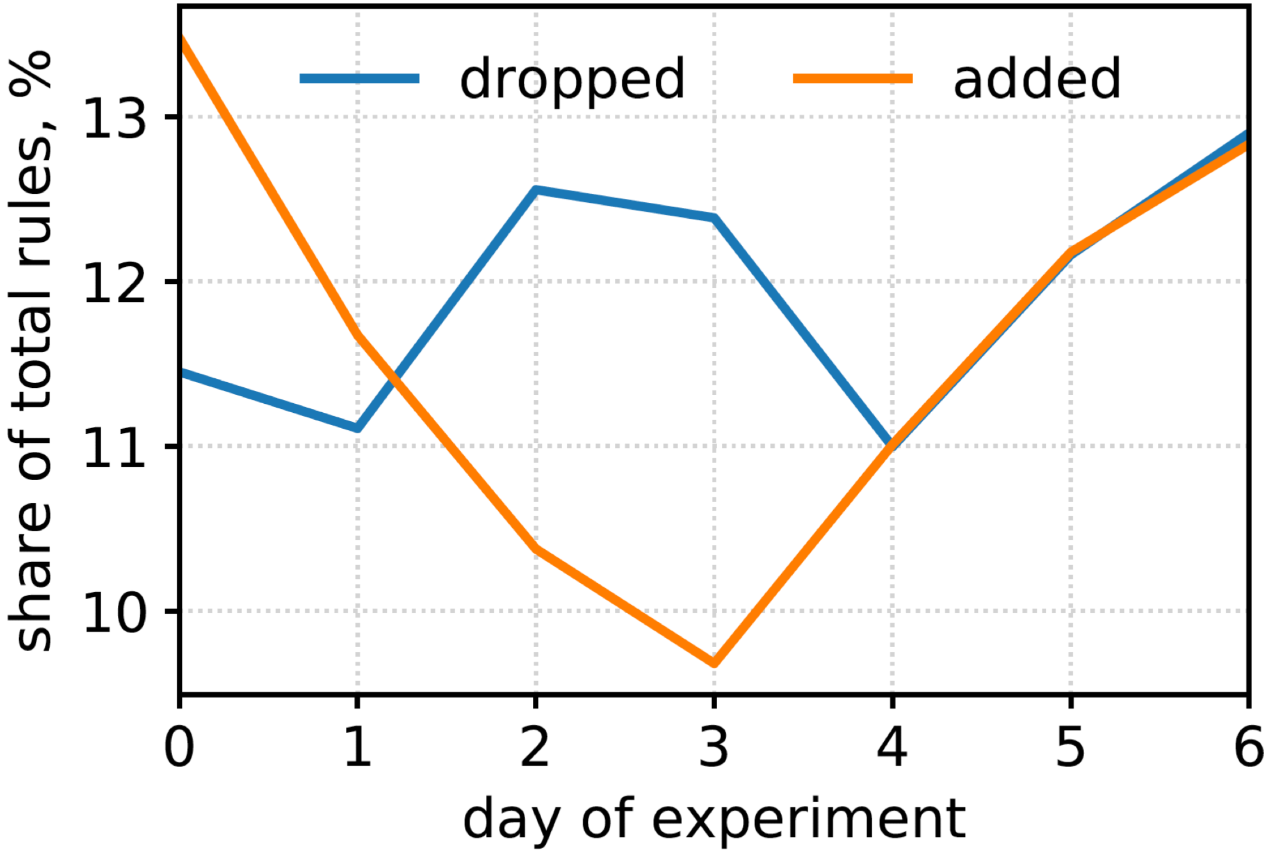
Production Pipeline



- update the model if it passes the tests and serve it to 90% of the users
  - leave 5% for A/B experiments with better models

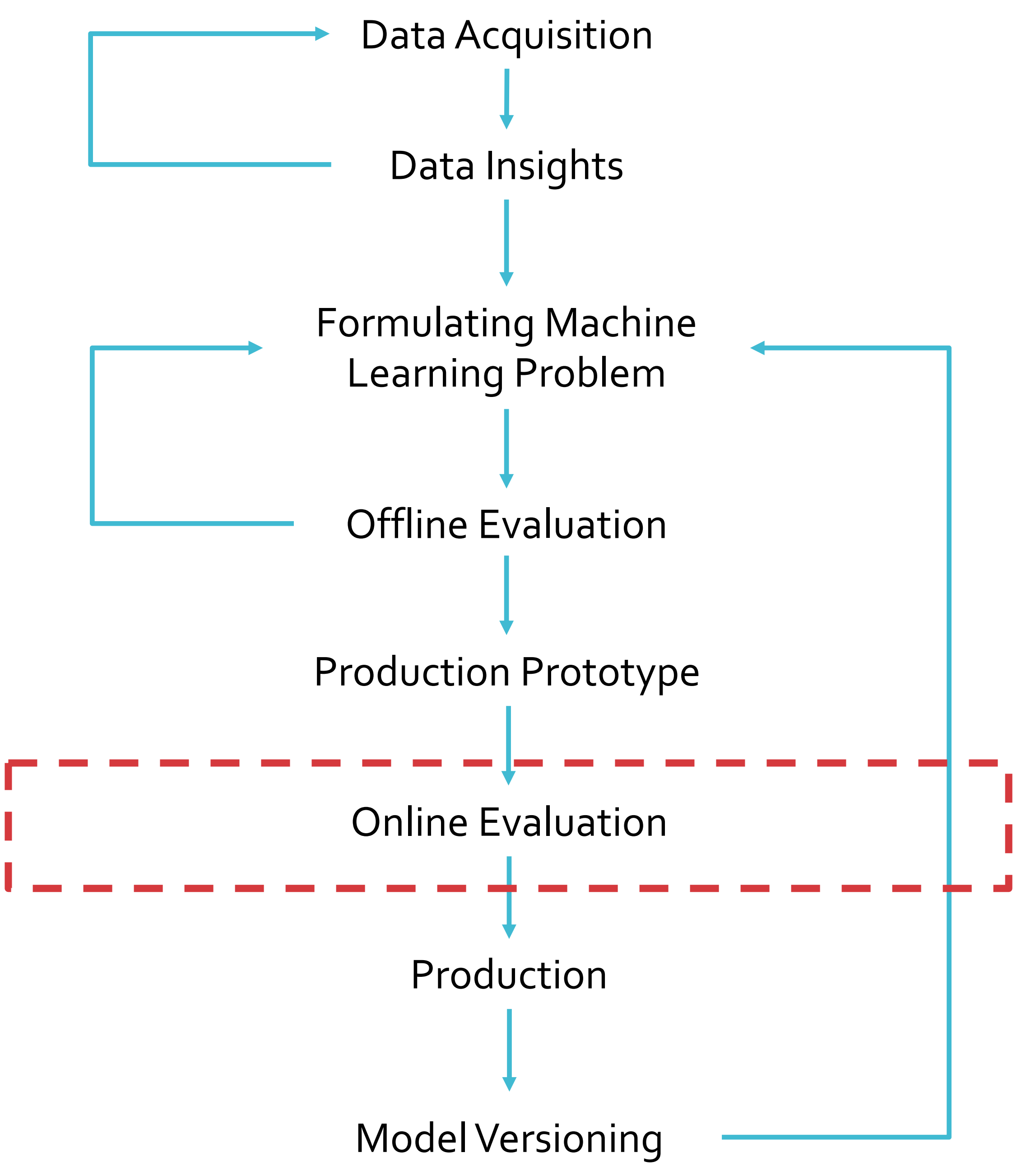
Temporal Stability

(Origin, Destination, Provider) rules

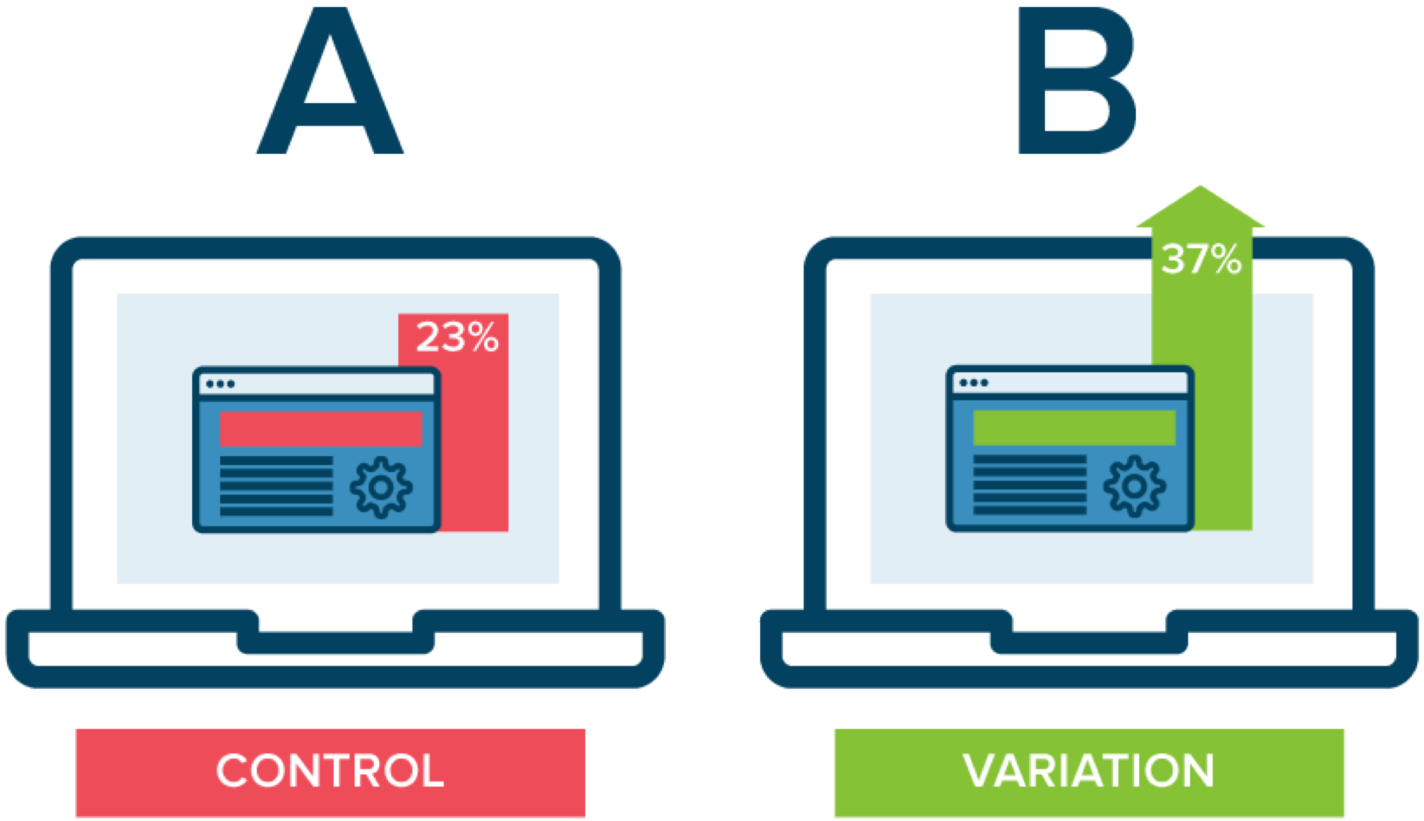


We need a mechanism to control temporal stability of the model

Product Cycle



Online Experiments



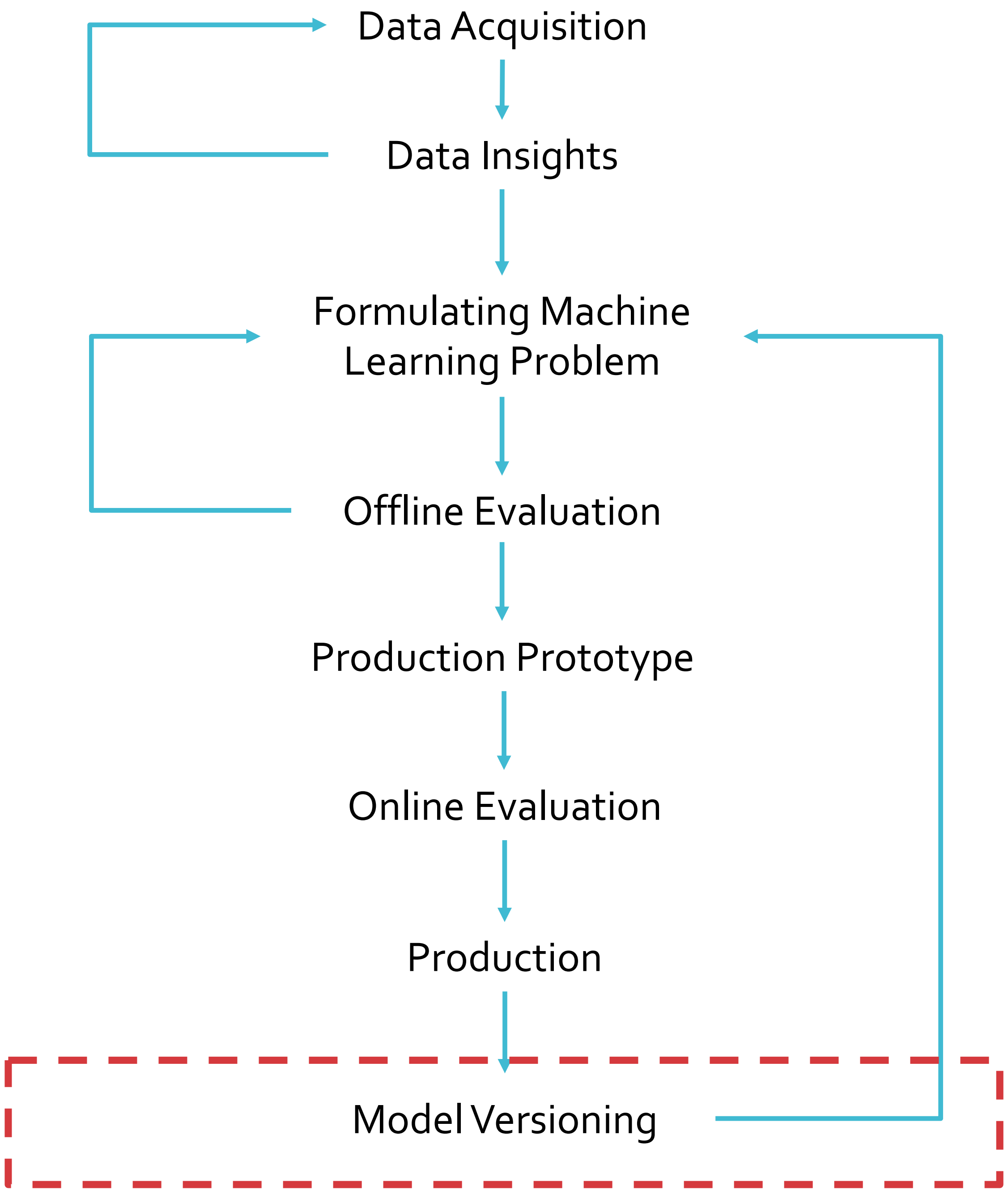
+ Test in the **real world**
+ **Benchmark** in equal conditions

- Some things are **difficult** to A/B test
- Online experiments might be **expensive**

Travelers First

- 45% of all competitive combinations for only 5% of the cost
- 22% of search results with cheaper itineraries
- 20% rel. increase in bookings on combination itineraries
- 0.74% rel. increase in user retention

Product Cycle



Feature Engineering

Can we improve performance with smart feature engineering?

One-hot encoding	
London Gatwick	[1 0 0 … 0]
London Stansted	[0 1 0 … 0]
Barcelona	[0 0 1 … 0]

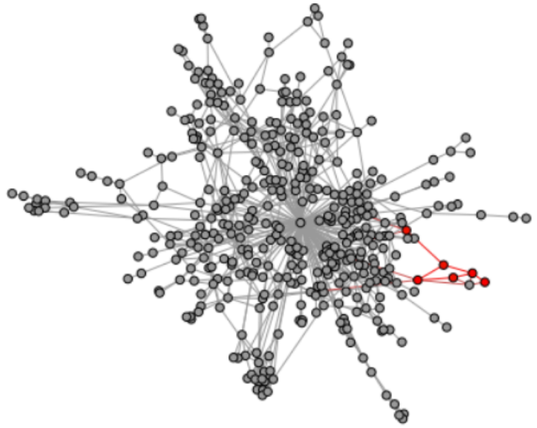
Better encoding	London European Trans-Atlantic
London Gatwick	[1.0 0.9 0.9 …]
London Stansted	[1.0 0.9 0.1 …]
Barcelona	[0.0 1.0 0.5 …]

Location Embeddings

word

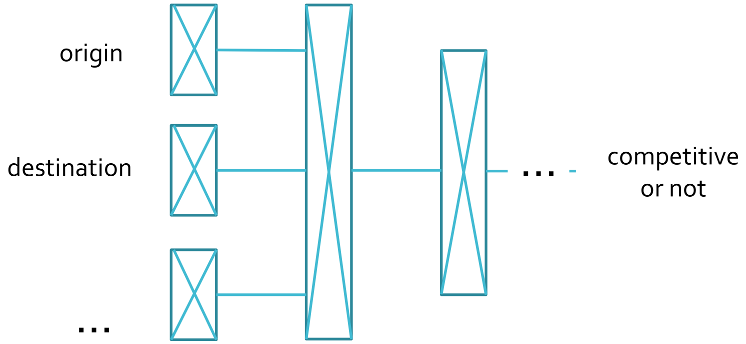
[London, Barcelona, Frankfurt am Main, New York, ...] — sentence

- **Option N1:** Every user's history is a sentence (think of Word2Vec)
- **Option N2:** Learn embeddings on graphs of locations



Perozzi et al., KDD, 2014.

- **Option N3:** Train embeddings for target problem



Location Embeddings

London Heathrow		Beijing Capital	
Airport	**Similarity**	**Airport**	**Similarity**
Frankfurt am Main	0.71	Chubu Centrair	0.91
Manchester	0.69	Taipei Taoyuan	0.90
Amsterdam Schipol	0.62	Seoul Incheon	0.90
Paris Charles de Gaulle	0.62	Miyazaki	0.88
London Gatwick	0.61	Shanghai Pudong	0.88

- Capture geographical proximity (Europe vs. Asia)
- Learn function of the airport (Heathrow and Gatwick vs. Stansted)
- Produce a slight improvement in prediction performance

Learnings

- **focus on right problems** which cannot be solved without ML or where ML gives 10x improvement
- **define the metrics and optimization objective** at the start of the project and stick to them thereafter
- **bootstrapping ML projects** requires 20% of modeling and 80% of engineering – in the long run should be vice versa
- **lean online experiments** are important on early stages to make sure users engage with the product
- **ML behavior in production** reveals interesting problems which are not visible during offline modeling (e.g., temporal stability)

Join our Team!

Dima.Karamshuk@skyscanner.net

on Twitter: **@karamshuk @SkyscannerEng**