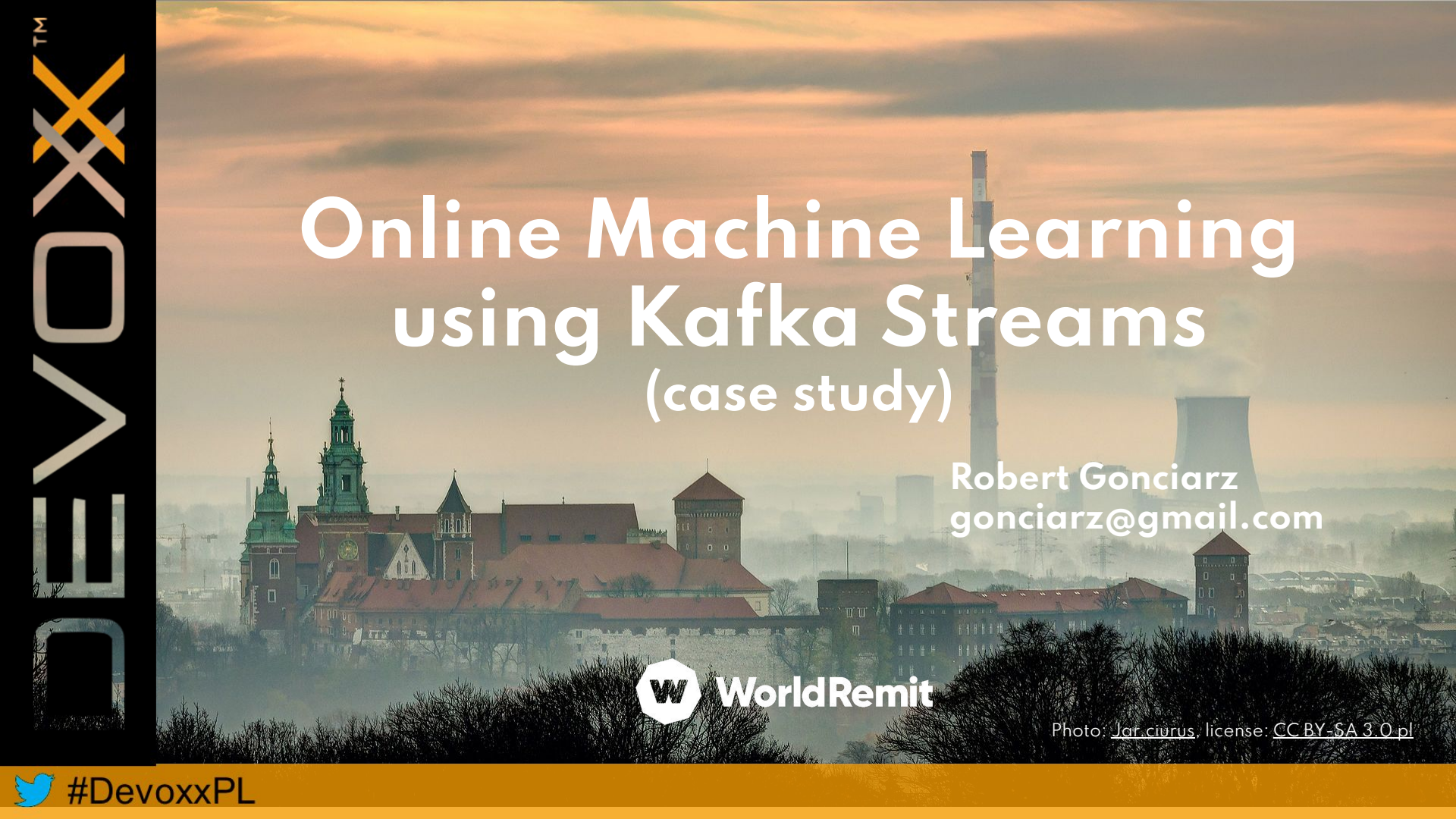
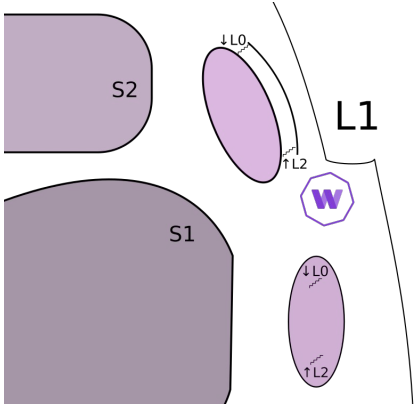

DEVOXX TM
Online Machine Learning using Kafka Streams
(case study)
Robert Gonciarz
gonciarz@gmail.com
WorldRemit
Photo: Jar.ciurus, license: CC BY-SA 3.0 pl
#DevoxxPL

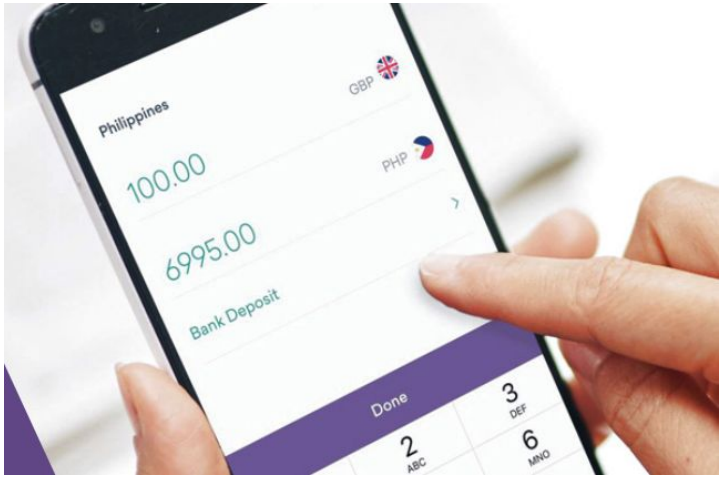
About me

- Husband, Father, Rock climber, Physicist, Software Engineer
- 15 years experience, mostly in distributed systems
- “If you’re good at something, never do it for free” (Joker)
- Email: gonciarz@gmail.com, L: linkedin.com/in/gonciarz

About Worldremit

- Founded 2010, 1500+ employees (300+ engineers/product)
- 18 offices, tech mostly in London and Kraków
- We transfer money to 130 countries (70 currencies, 5000+ agents)
- Kraków's office: al 29 listopada (near to main railway station)
- **We are hiring!** (Java/Kotlin, DevOps, Cloud, React, QA, etc.)





https://www.officefinder.pl/office-cracow-voffices.html

https://tinyurl.com/**x783p4**

Pollution Predictor

- Predict pollution level based on historical data and current weather conditions
- **Goal**:
  - Present that JVM based services and Kafka Streams may be useful in ML problems
  - Publish end-to-end implementation example
- **Non-Goal**:
  - Find a perfect ML model that would maximize prediction's success rate
- Why such a domain?

Photo: Grzegorz Łach, Krzysztof Kardaś, Aneta Zawadzka, Gabriela Lisowska, Daria Góźdź - Thank you :)

Air Quality standards (EPA, GIOS)

- AQI:

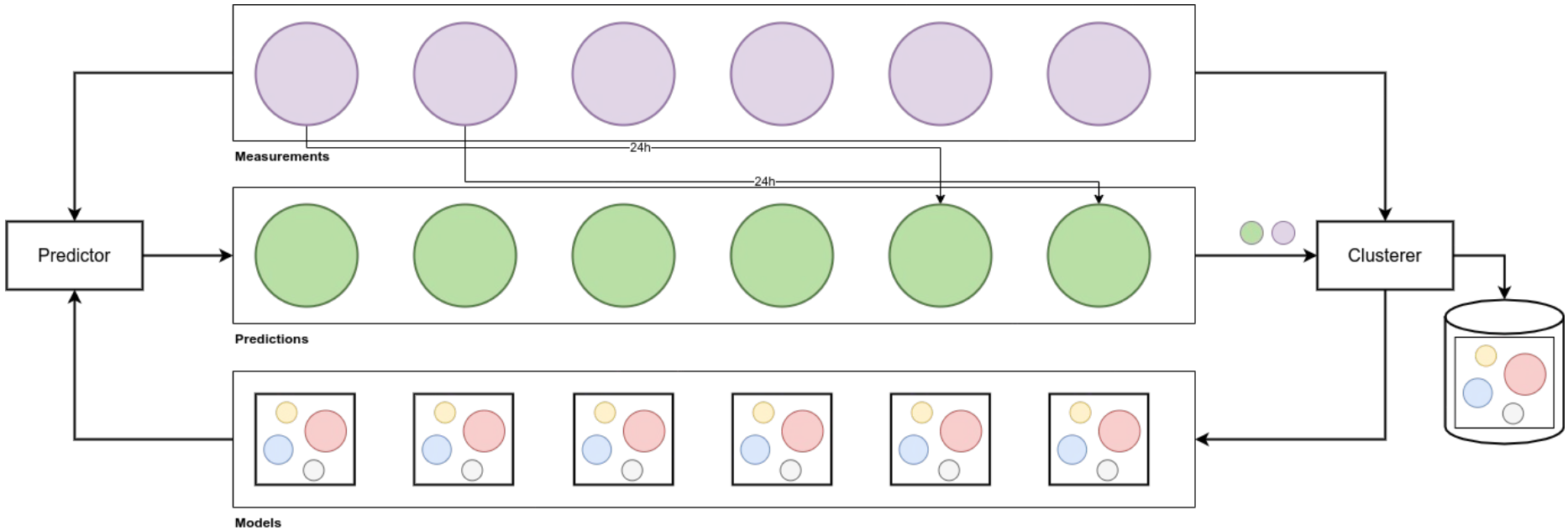
- Labels:

Feature	Unit	Good	Moderate	Unhealthy
PM25	ppb/24h	0-12	12-13.5	35.5+
PM10	ppb/24h	0-55	55-155	155+
NO2	ppb/1h	0-53	53-100	100+
CO	ppb/8h	0-4.5k	4.5k-9.5k	9.5+
SO2	ppb/1h	0-35	35-75	75+
O3	ppb/8h	0-55	55-70	70+

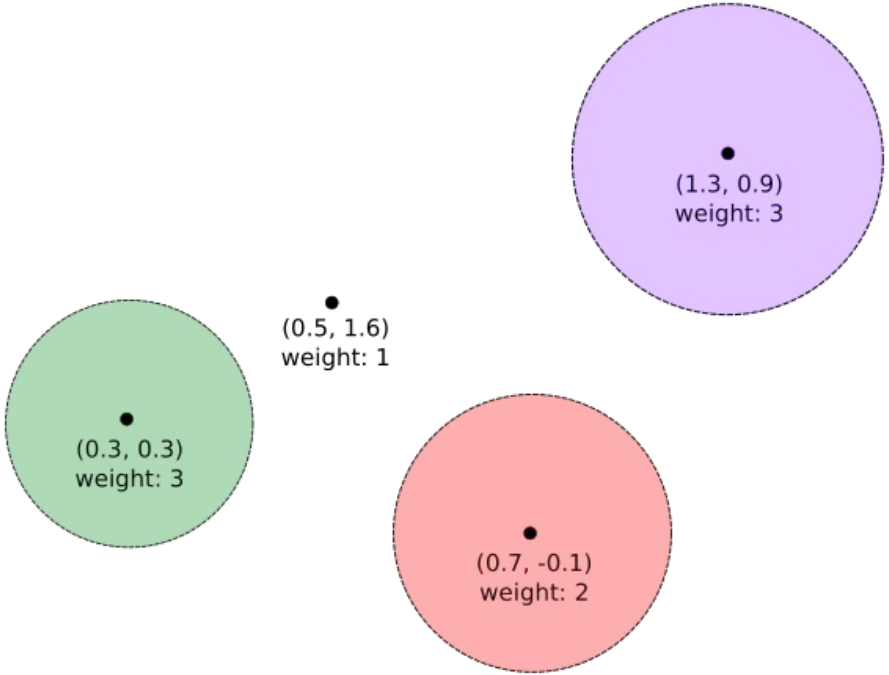
https://www.forbes.com/sites/emanuelabarbiroglio/2020/02/29/cities-in-poland-and-italy-among-europes-100-most-polluted

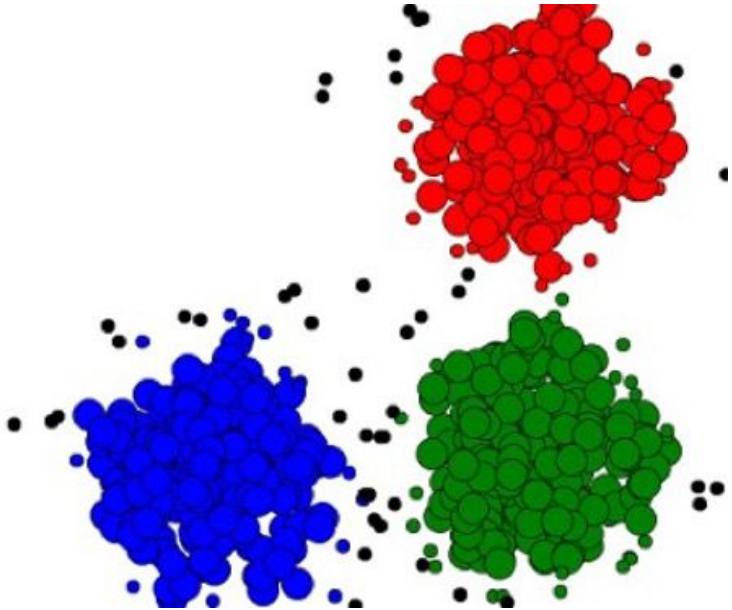
A view of Krakow's Wawel Royal Castle, during a smog alert on January 14, Nurphoto via Getty Images

Main concept



Online vs batch clustering

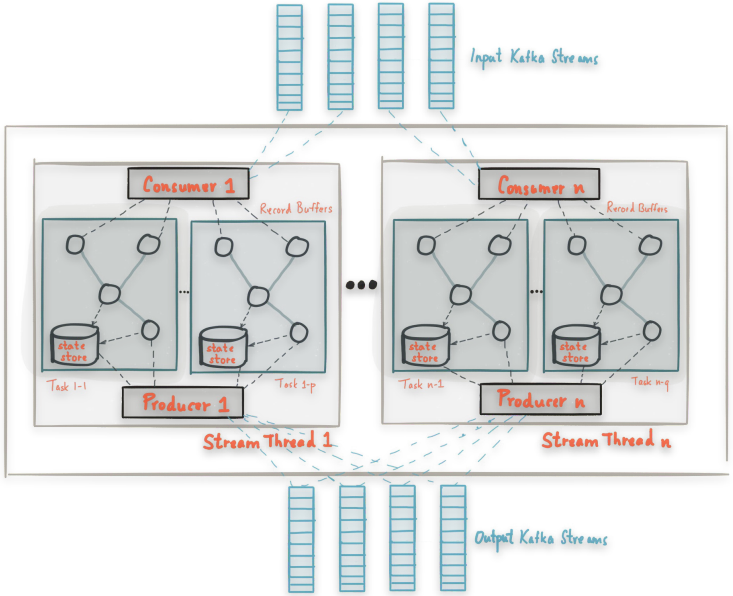


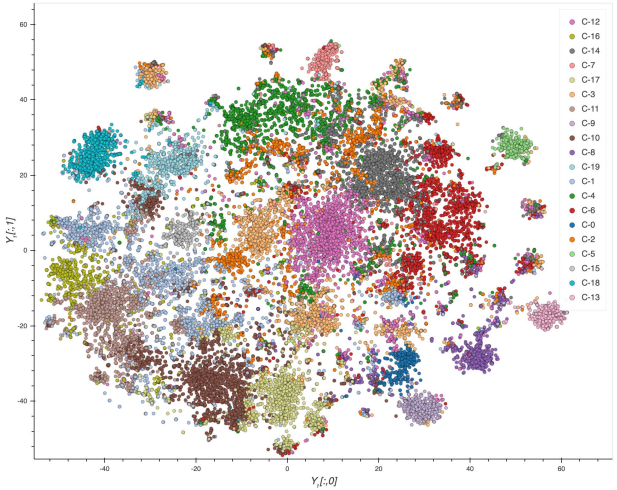


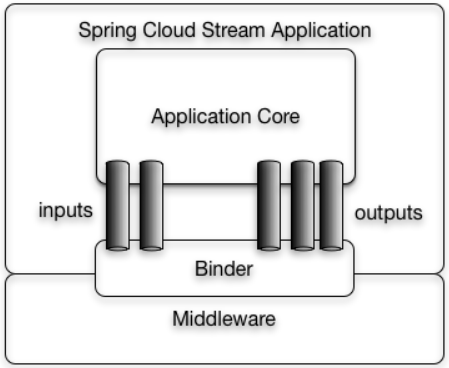
by Matthew Mayo, Machine Learning Key Terms, Explained

Technology

- Kafka Streams
- Spring Cloud Stream
- Kotlin
- Avro
- Mahout
- Bokeh



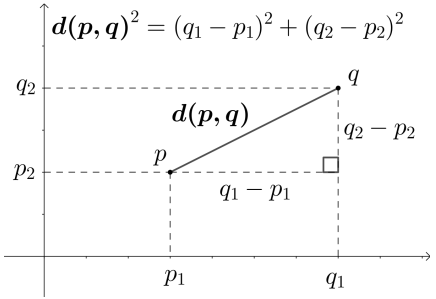


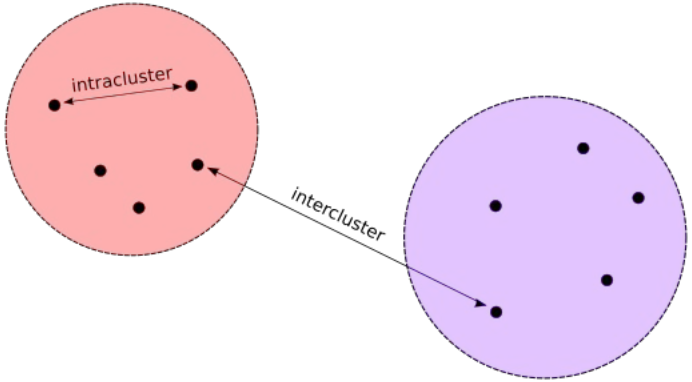


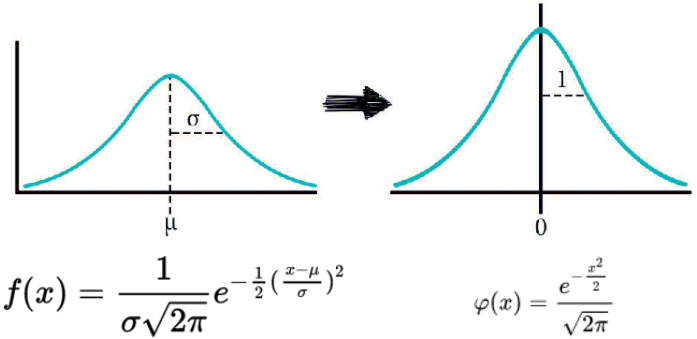
https://www.kaggle.com/maksimeren/covid-19-literature-clustering

K-Means clustering

1. Finding similarities
2. Distance metric
3. Common scale







S. Lakshmanan, https://towardsai.net/p/data-science/how-when-and-why-should-you-normalize-standardize-rescale-your-data-3f083def38ff
https://en.wikipedia.org/wiki/Euclidean_distance

Why streaming clustering?

Streaming k-means approximation

Nir Ailon
Google Research
nailon@google.com

Ragesh Jaiswal*
Columbia University
rjaiswal@gmail.com

Claire Monteleoni†
Columbia University
cmontel@ccls.columbia.edu

Abstract

We provide a clustering algorithm that approximately o
jective, in the one-pass streaming setting. We make n
data, and our algorithm is very light-weight in terms o
tion. This setting is applicable to unsupervised learning
resource-constrained devices. The two main ingredient
are: a derivation of an extremely simple pseudo-appro
for k-means (based on the recent k-means++), in which
to output more than k centers, and a streaming clustering
clustering algorithms are performed on small inputs (fitt
bined in a hierarchical manner. Empirical evaluations o
reveal the practical utility of our method.

Algorithm 1 Fast streaming k-means (data stream, k, κ, β)

Initialize $f = 1/(k(1+\log n))$ and an empty set K
while some portion of the stream remains unread **do**
  while $|K| \leq \kappa = \Theta(k \log n)$ and some portion of the stream is unread **do**
    Read the next point x from the stream
    Measure $\delta = \min_{y \in K} d(x,y)^2$
    if probability δ/f event occurs **then**
      set $K \leftarrow K \bigcup \{x\}$
    else
      assign x to its closest facility in K
  if stream not exhausted **then**
    while $|K| > \kappa$ **do**
      Set $f \leftarrow \beta f$
      Move each $x \in K$ to the center-of-mass of its points
      Let w_x be the number of points assigned to $x \in K$
      Initialize $\hat{K}$ containing the first facility from K
      for each $x \in K$ **do**
        Measure $\delta = \min_{y \in \hat{K}} d(x,y)^2$
        if probability $w_x \delta/f$ event occurs **then**
          set $\hat{K} \leftarrow \hat{K} \bigcup \{x\}$
        else
          assign x to its closest facility in $\hat{K}$
      Set $K \leftarrow \hat{K}$
  else
    Run batch k-means algorithm on weighted points K
    Perform ball k-means (as per [9]) on the resulting set of clusters

Ref 1: NIPS'09

Fast and Accurate k-means For Large Datasets

Michael Shindler
School of EECS
Oregon State University
shindler@eecs.oregonstate.edu

Alex Wong
Department of Computer Science
UC Los Angeles
alexw@seas.ucla.edu

Adam Meyerson
Google, Inc.
Mountain View, CA
awmeyerson@google.com

Abstract

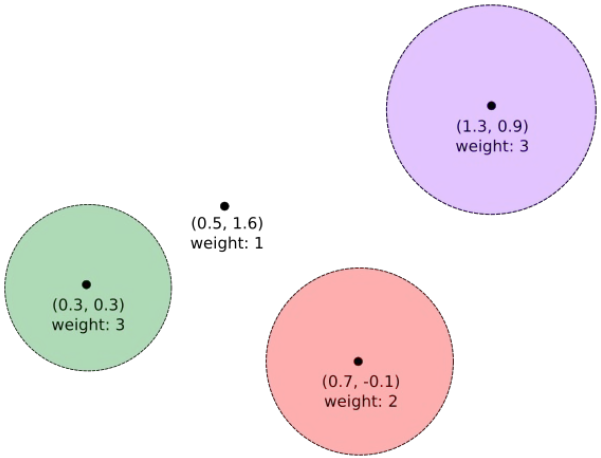
Clustering is a popular problem with many applications. We consider the k-means
lem in the situation where the data is too large to be stored in main memory
must be accessed sequentially, such as from a disk, and where we must use
ttle memory as possible. Our algorithm is based on recent theoretical results,
significant improvements to make it practical. Our approach greatly simpli-
a recently developed algorithm, both in design and in analysis, and eliminates
constant factors in the approximation guarantee, the memory requirements,
he running time. We then incorporate approximate nearest neighbor search to
pute k-means in $o(nk)$ (where n is the number of data points; note that com-
g the cost, given a solution, takes $\Theta(nk)$ time). We show that our algorithm
pares favorably to existing algorithms - both theoretically and experimentally,
providing state-of-the-art performance in both theory and practice.

Ref 2: NIPS'11

NEURAL INFORMATION PROCESSING SYSTEMS

Streaming K-means clustering with Kafka

1. Mahout's implementation of "Fast and accurate k-means"
2. **Custom implementation** of the serializing/deserializing cluster




```
val closestPoint = searcher.searchFirst(centroidToCluster, false)

if (centroidToCluster.weight * closestPoint.weight / distanceCutoff > nextDouble()) {
    searcher.add(centroidToCluster) // add new center to cluster
} else {
    closestPoint.update(centroidToCluster) // updated existing center
}

if (clusterOvershoot * clustersEstimated < searcher.size()) {
    // estimate number of clusters, shuffle centers and re-cluster
    if (clustersEstimated < searcher.size()) {
        distanceCutoff *= beta
    }
}
```

Data source / Features

Pollution data:
http://monitoring.krakow.pios.gov.pl

Weather data:
https://darksky.net


```
{
  "weatherId": {
    "value": "8e9ed377-d753-4cbf-a9b1-d5fbe119df8b"
  },
  "locationTime": {
    "location": {
      "latitude": 50.057678,
      "longitude": 19.926189
    },
    "timestamp": 1426777200
  },
  "data": {
    "icon": "partly-cloudy-day",
    "temperature": 7.78,
    "temperatureApparent": 5.27,
    "dewPoint": -2.96,
    "humidity": -2.96,
    "pressure": 1028,
    "windSpeed": 14.39,
    "windGust": 19.33,
    "windDirection": 67,
    "cloudCover": 0.44,
    "precipitationIntensity": 0,
    "precipitationProbability": 0,
    "precipitationType": null,
    "uvIndex": 2,
    "visibility": 10.003,
    "solar": {
      "azimuth": 220,
      "altitude": 32,
      "dni": 644,
      "ghi": 376,
      "dhi": 34.2,
      "etr": 1373
    }
  }
}
```



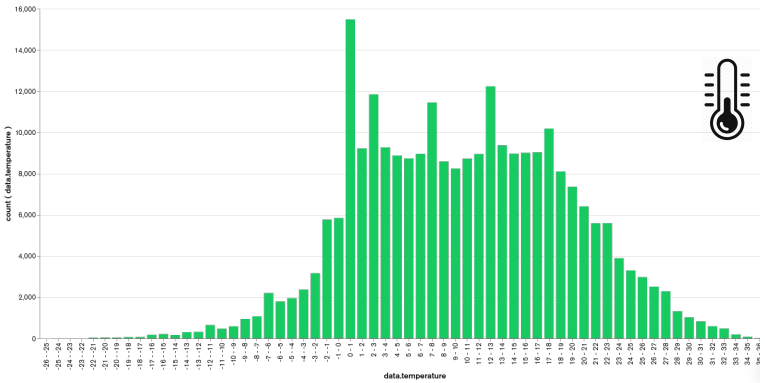
```
{
  "pollutionId": {
    "value": "9be4c513-2b74-4281-9d5a-29ffe8930709"
  },
  "locationTime": {
    "location": {
      "latitude": 50.057678,
      "longitude": 19.926189
    },
    "timestamp": 1426698000
  },
  "data": {
    "pm10": 62.5384,
    "pm25": 34.0352,
    "bzn": 0,
    "no2": 72.4939,
    "nox": 166.2,
    "no": 61.3244,
    "co": 587.435,
    "so2": 0
  }
}
```

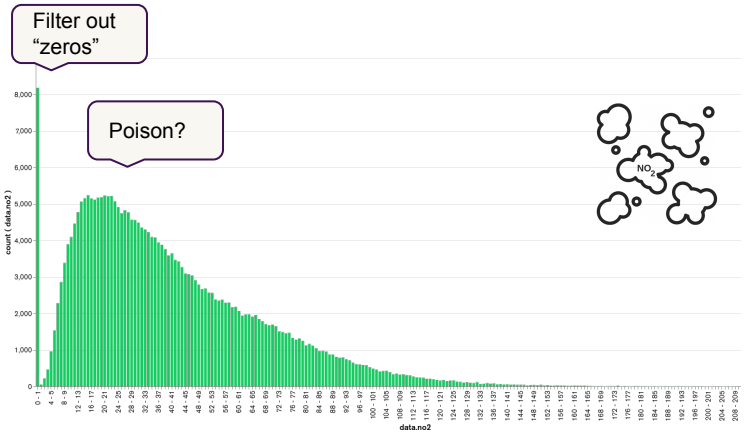

CET

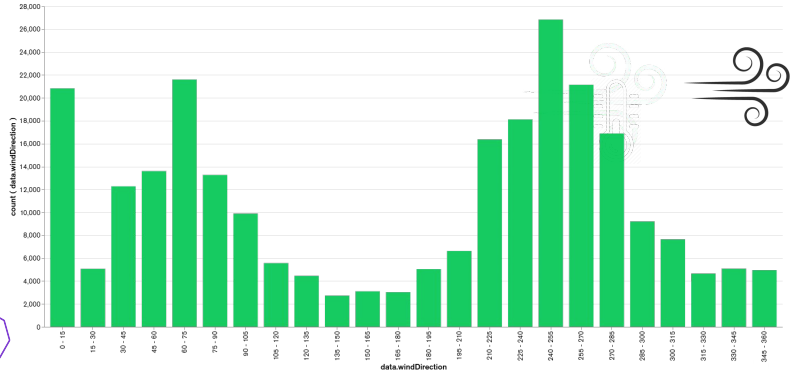
NO

Photo: J.Ociepa, Stacja monitoringu zanieczyszczeń powietrza w Krakowie,
https://krakow.wyborcza.pl/krakow/7,44425,24344071,osma-stacja-monitorowania-jakosci-powietrza-stanela-w-krakowie.html

Probability mass function







One-Hot

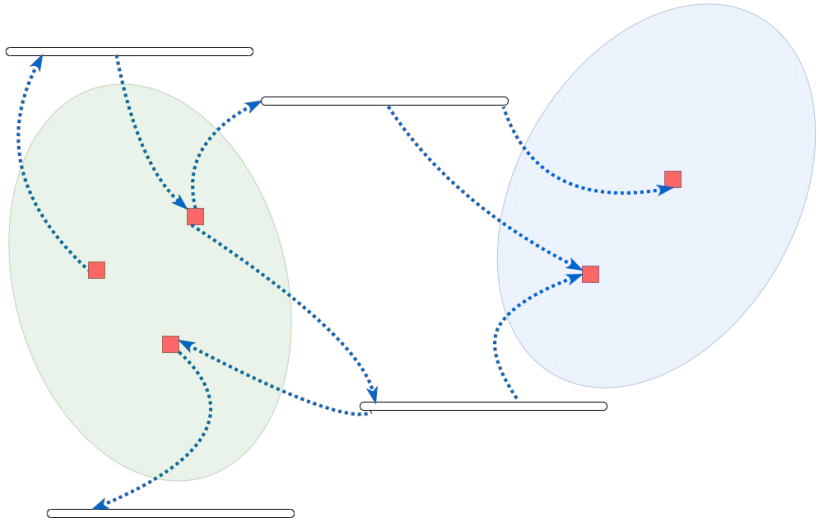
```
enum Season {
  WINTER, // XII-II
  SPRING, // III-V
  SUMMER, // VI-VIII
  AUTUMN // IX-XI
}
```

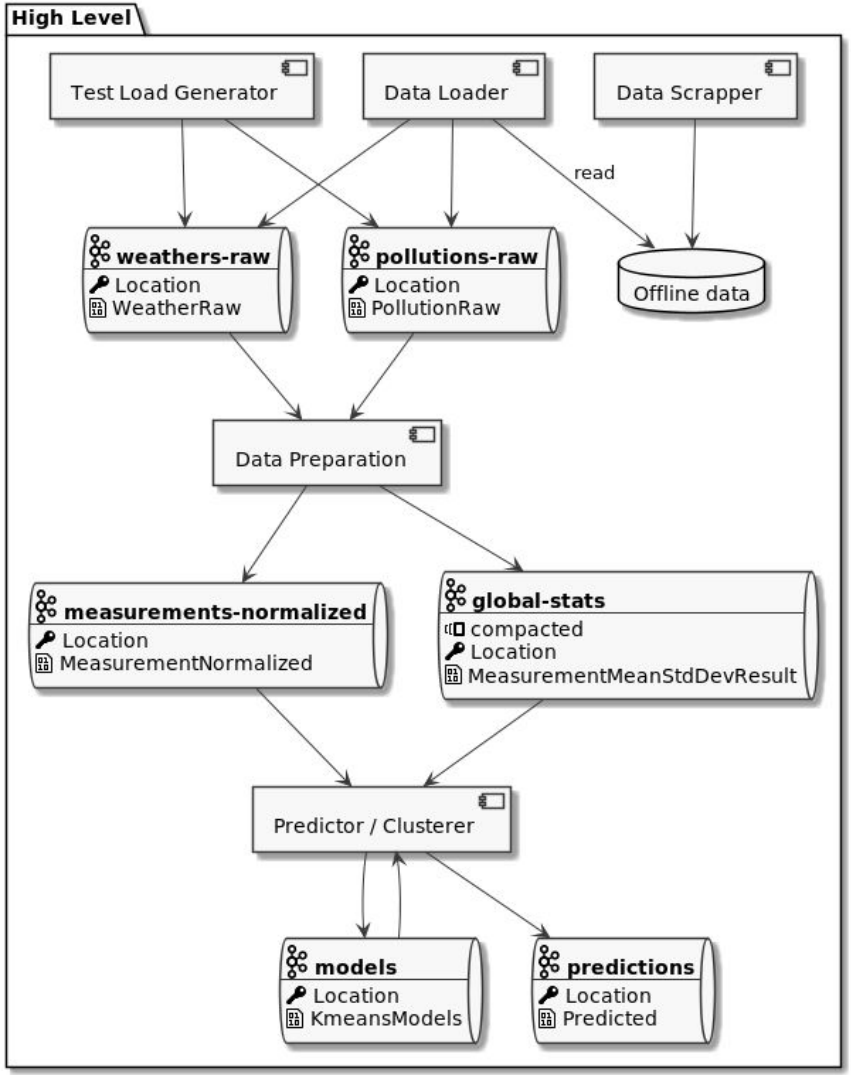
→

```
SEASON_WINTER: 0.0,
SEASON_SPRING: 1.0,
SEASON_SUMMER: 0.0,
SEASON_AUTUMN: 0.0,
```

System design

1. Kafka Key
2. Services
3. Topologies





Back in time

Data Expiration problem

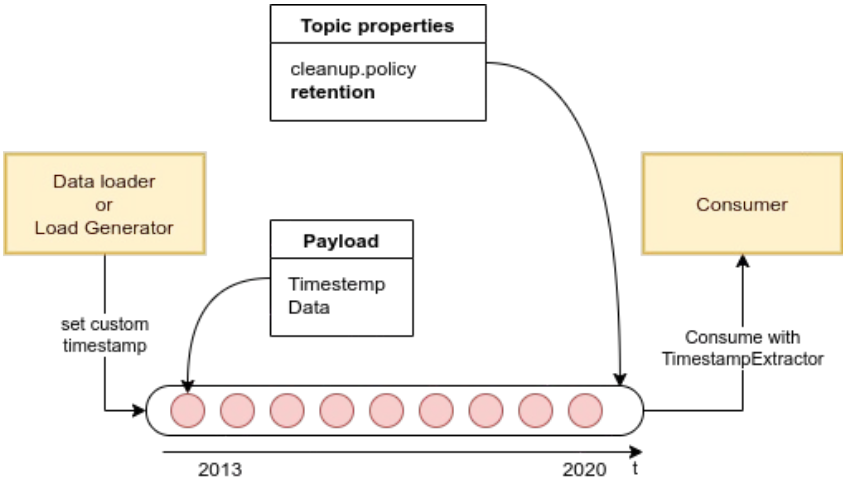


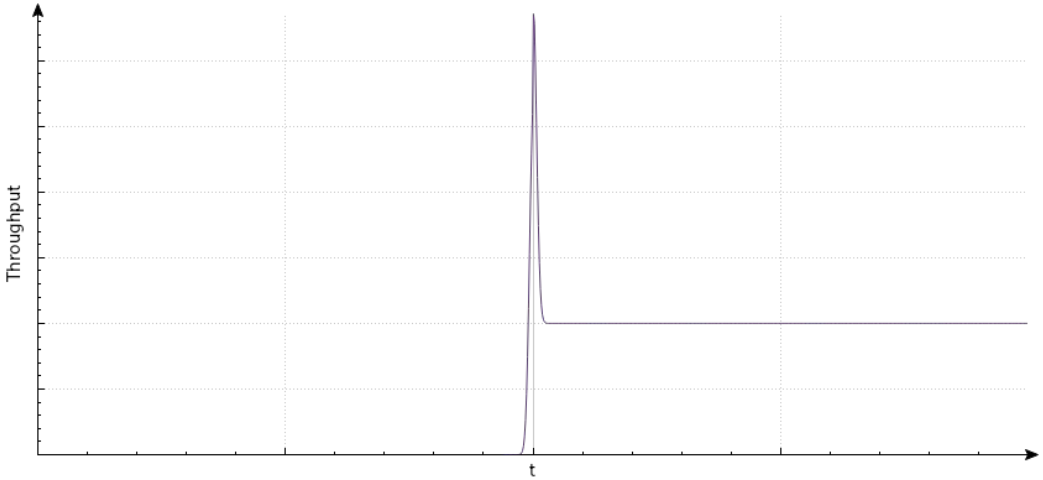
Photo: https://puckbuddys.com/timemachine-2/

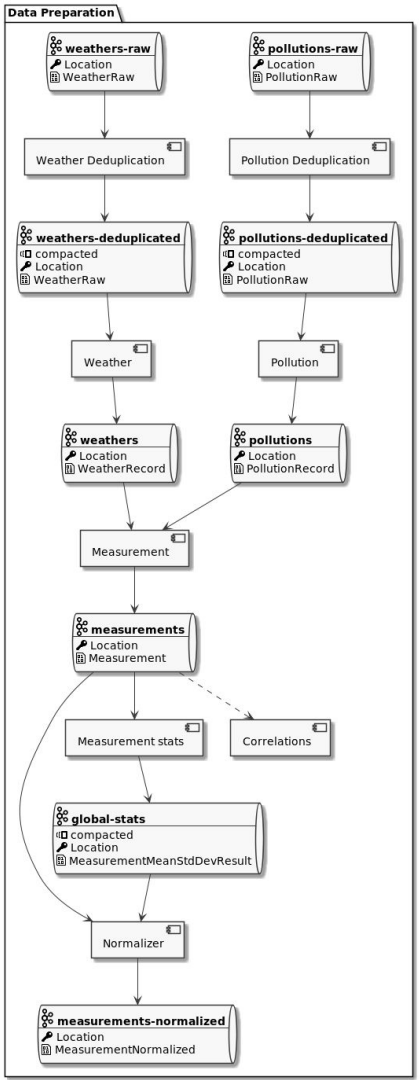
Data preparation

1. Removing zeros & non-realistic data
2. Processing missing data [removed]
3. Online one-pass aggregations
4. Identify non/over-correlated features
5. Discretization / Ranges transformation
6. Normalize & removing outsiders

Dynamic features problem

Global normalization problem





One-pass variance

calculate the standard deviation of some observations by using the textbook formula

$$\sigma = \sqrt{\left(n \sum_{1\le k\le n} x_k^2 - \left(\sum_{1\le k\le n} x_k\right)^2\right) \Big/ n(n-1)} \qquad (14)$$

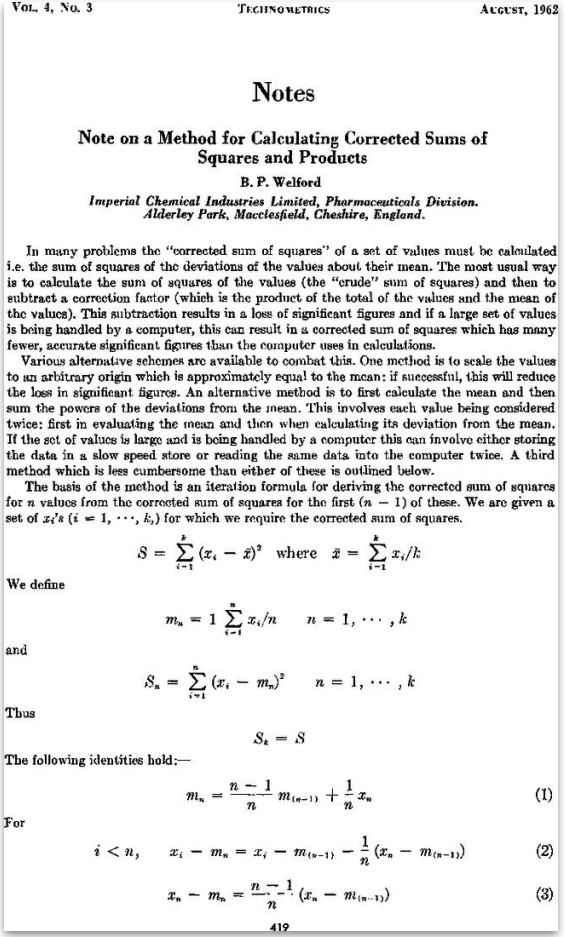
often find themselves taking the square root of a negative number! A much better way to calculate means and standard deviations with floating point arithmetic is to use the recurrence formulas

$$M_1 = x_1, \qquad M_k = M_{k-1} \oplus (x_k \ominus M_{k-1}) \oslash k, \qquad (15)$$

$$S_1 = 0, \qquad S_k = S_{k-1} \oplus (x_k \ominus M_{k-1}) \otimes (x_k \ominus M_k), \qquad (16)$$

for $2 \le k \le n$, where $\sigma = \sqrt{S_n/(n-1)}$. [See B. P. Welford, *Technometrics* **4** (1962), 419–420.] With this method S_n can never be negative, and we avoid other serious problems encountered by the naïve method of accumulating sums, as shown in exercise 16. (See exercise 19 for a summation technique that provides an even better guarantee on the accuracy.)

[Donald Knuth, Art of Computer Programming, 1962, Vol 2, p 232, 3rd edition]
[B.P. Welford, Technometrics 4, 1962, p 419-420]



VOL. 4, NO. 3 TECHNOMETRICS AUGUST, 1962

Notes

Note on a Method for Calculating Corrected Sums of Squares and Products

B. P. Welford

Imperial Chemical Industries Limited, Pharmaceuticals Division. Alderley Park, Macclesfield, Cheshire, England.

In many problems the "corrected sum of squares" of a set of values must be calculated i.e. the sum of squares of the deviations of the values about their mean. The most usual way is to calculate the sum of squares of the values (the "crude" sum of squares) and then to subtract a correction factor (which is the product of the total of the values and the mean of the values). This subtraction results in a loss of significant figures and if a large set of values is being handled by a computer, this can result in a corrected sum of squares which has many fewer, accurate significant figures than the computer uses in calculations.

Various alternative schemes are available to combat this. One method is to scale the values to an arbitrary origin which is approximately equal to the mean: if successful, this will reduce the loss in significant figures. An alternative method is to first calculate the mean and then sum the powers of the deviations from the mean. This involves each value being considered twice: first in evaluating the mean and then when calculating its deviation from the mean. If the set of values is large and is being handled by a computer this can involve either storing the data in a slow speed store or reading the same data into the computer twice. A third method which is less cumbersome than either of these is outlined below.

The basis of the method is an iteration formula for deriving the corrected sum of squares for n values from the corrected sum of squares for the first $(n-1)$ of these. We are given a set of x_i's $(i = 1, \cdots, k,)$ for which we require the corrected sum of squares.

$$S = \sum_{i=1}^{k} (x_i - \bar{x})^2 \quad \text{where} \quad \bar{x} = \sum_{i=1}^{k} x_i/k$$

We define

$$m_n = 1 \sum_{i=1}^{n} x_i/n \qquad n = 1, \cdots, k$$

and

$$S_n = \sum_{i=1}^{n} (x_i - m_n)^2 \qquad n = 1, \cdots, k$$

Thus

$$S_k = S$$

The following identities hold:—

$$m_n = \frac{n-1}{n} m_{(n-1)} + \frac{1}{n} x_n \qquad (1)$$

For

$$i < n, \qquad x_i - m_n = x_i - m_{(n-1)} - \frac{1}{n}(x_n - m_{(n-1)}) \qquad (2)$$

$$x_n - m_n = \frac{n-1}{n}(x_n - m_{(n-1)}) \qquad (3)$$

419

Over and non-correlated features

- Pearson Correlation (one-pass co-variance algorithm)
- Limits: e.g. <20%-80%>

	PM25	PM10	BZN	NO2	NO	CO	NOX	SO2
temp	-60%	-53%	-42%	-18%	-29%	-40%	-29%	-51%
tempApp	-54%	-48%	-38%	-14%	-23%	-34%	-23%	-51%
dewPoint	-57%	-53%	-35%	-23%	-25%	-36%	-26%	-59%
humidity	27%	17%	24%	-6%	20%	24%	17%	-4%
pressure	10%	10%	-1%	2%	-3%	-5%	-2%	14%
windSpeed	-30%	-31%	-20%	-29%	-32%	-32%	-33%	2%
windGust	7%	-9%	12%	-19%	-4%	8%	-9%	12%
cloudCover	11%	4%	-6%	-11%	-2%	1%	-4%	6%
visibility	-60%	-56%	-42%	-18%	-42%	-53%	-41%	-28%

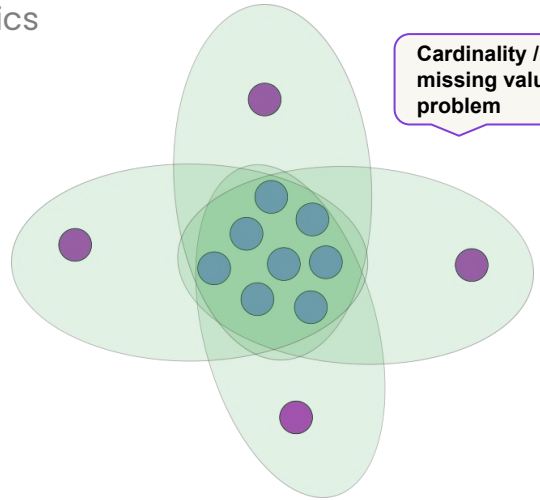
	temp	temp App	dew Point	*hum*	*press*	wind Speed	*wind Gust*	*cloud Cover*
temp								
tempApp	**99%**							
dewPoint	**93%**	**93%**						
humidity	-47%	-43%	-11%					
pressure	-16%	-16%	-21%	-6%				
windSpeed	-2%	-10%	-7%	-16%	-5%			
windGust	-4%	-8%	5%	14%	-16%	79%		
cloudCover	-37%	-36%	-22%	47%	-9%	11%	7%	
visibility	50%	45%	38%	-56%	-5%	32%	-11%	-28%

Live demo

https://funnyjunk.com/comment/anonymous/content/4519776/-999/1/parent_id/10/1/desc

Prediction / Clustering

1. Predictor / initial models
2. Matcher
3. Forecaster
4. Clusterer
5. Metrics

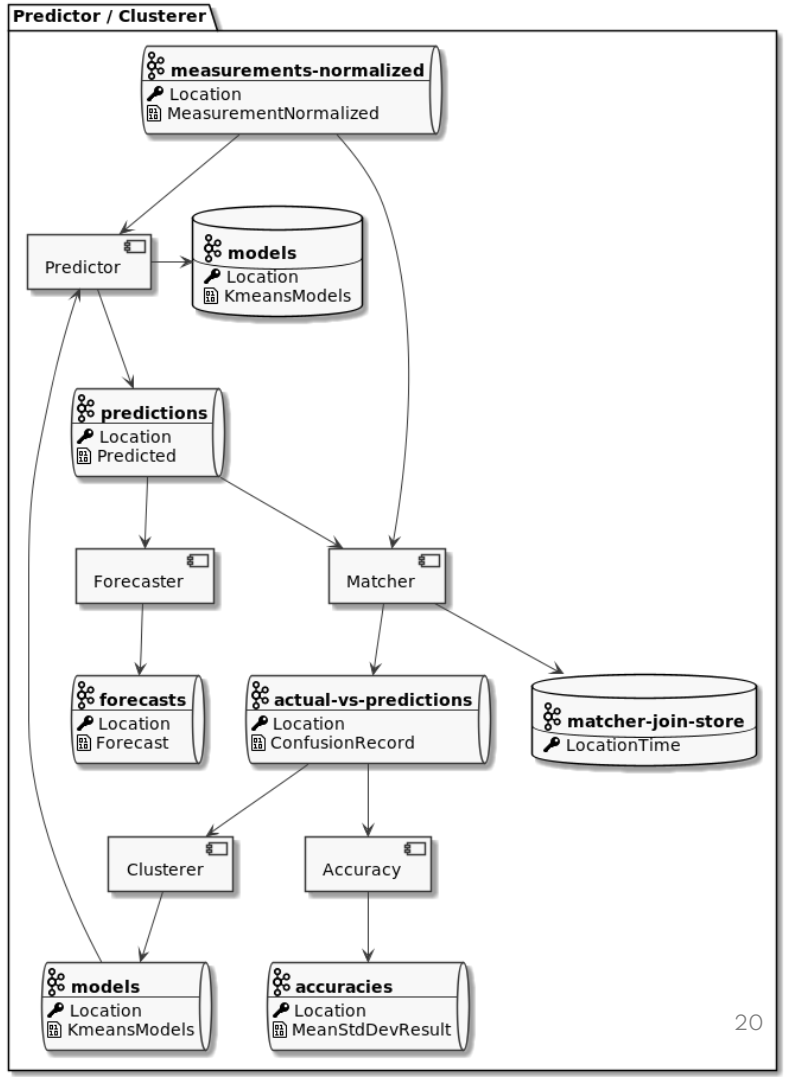


Cardinality / missing values problem

commit.interval

de/serialization models overhead

pollution feature (actual)
weather feature (past)



Model


```
{
  "locationTime": {...},

  "models": {

    "PM25": {

      "processedDataPoints": 5,
      "clustersEstimated": 10,
      "distanceCutoff": 0.01,

      "featuresDescription": [
        "SEASON_WINTER", "SEASON_SPRING", "SEASON_SUMMER", "SEASON_AUTUMN",
        "DAY_NIGHT", "TEMPERATURE", "WIND_SPEED", "VISIBILITY", "PM25"
      ],

      "clusterCenters": [
        {
          "weight": 1,
          "coordinates": [ 0, 1, 0, 0, 1, 0.88, -1.49, 1.95, 0.64 ]
        },
        {
          "weight": 2,
          "coordinates": [ 0, 1, 0, 0, 0, 0.28, -0.60, 1.79, 0.58 ]
        }
      ]

    },

    "NO2": {...}

  }
}
```


Prediction

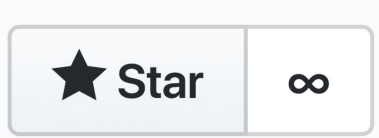

```
{
  "locationTime": {...},
  "initialMeasurement": {...},
  "modelMaturityHours": 30,

  "predictedPatterns": {
    "NO2": {
      "featuresDescription": [
        "SEASON_WINTER", "SEASON_SPRING", "SEASON_SUMMER", "SEASON_AUTUMN",
        "DAY_NIGHT", "TEMPERATURE", "WIND_SPEED", "VISIBILITY", "NO2"
      ],
      "coordinates": [ 0.77, 0.18, 0, 0.04, 0.12, 1.34, 0.39, 0.98, 0.71 ]
    },
    "PM25": {...}
  }
}
```


Matrix movie, Gloria Foster

Source code

- https://github.com/Worldremit/pollution-predictor-public
- License: Apache 2.0




```
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Kotlin                         180           1153            139           4653
Gradle                          10             94             10            462
YAML                             6            104             11            456
-------------------------------------------------------------------------------
SUM:                           196           1351            160           5571
-------------------------------------------------------------------------------
```

Lessons learned

What we achieved:

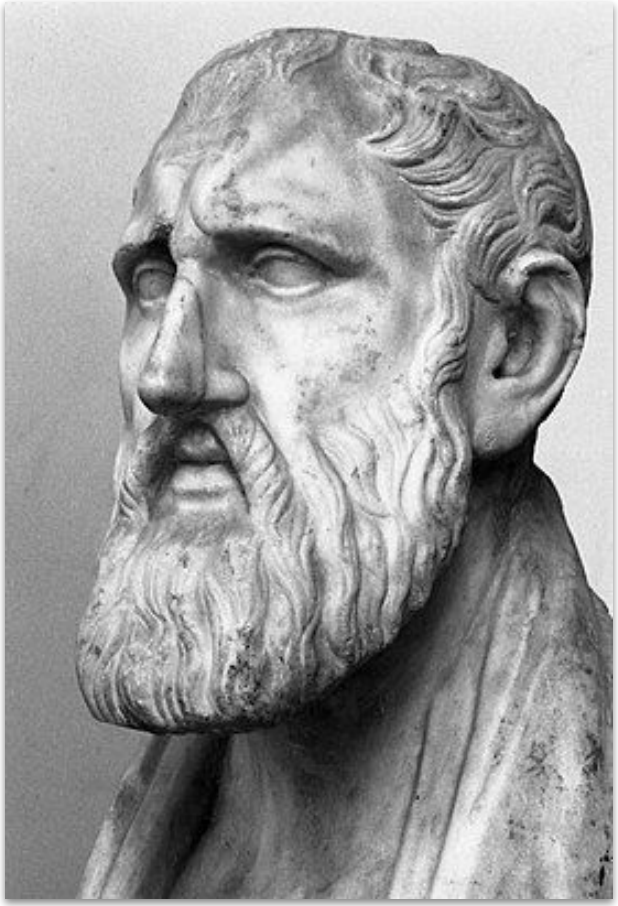
- Created a prototype able to ML problem
  In a streaming way based on Kafka Streams

What wasn't achieved:

- Visualization
- Dynamic features selection
- Missing values recreation
- Reinforcement learning

What to improve next:

- Use Python for ML operations
- Experiment with data as soon as possible



Zeno of Citium, in the Farnese collection, Naples, photo by Paolo Monti, 1969

Reference

1. Nir Ailon, Ragesh Jaiswal, Claire Monteleoni, “Streaming k-means approximation”, NIPS 2009, p.10-18
https://proceedings.neurips.cc/paper/2009/file/4f16c818875d9fcb6867c7bdc89be7eb-Paper.pdf
2. M. Schindler, A. Wong, A. Meyerson, “Fast and Accurate k-means for Large Datasets” NIPS 2011, p. 2375–2383
https://papers.nips.cc/paper/2011/file/52c670999cdef4b09eb656850da777c4-Paper.pdf
3. Donald Knuth, Art of Computer Programming, 1962, Vol 2, p 232, 3rd edition
https://seriouscomputerist.atariverse.com/media/pdf/book/Art%20of%20Computer%20Programming%20-%20Volume%202%20(Seminumerical%20Algorithms).pdf
4. B.P. Welford, Technometrics 4, 1962, p 419-420
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.302.7503&rep=rep1&type=pdf
5. K. Waehner, “How to Build and Deploy Scalable Machine Learning in Production with Apache Kafka”, 2017
https://www.confluent.io/blog/build-deploy-scalable-machine-learning-production-apache-kafka/
6. B. Bejeck, “Predicting Flight Arrivals with the Apache Kafka Streams API”, 2017
https://www.confluent.io/blog/predicting-flight-arrivals-with-the-apache-kafka-streams-api/
7. Y.Holtz, “Parallel coordinates plot”, 2018
https://www.data-to-viz.com/graph/parallel.html
8. A. Dehghani, “The K-Means Clustering Algorithm in Java”
https://www.baeldung.com/java-k-means-clustering-algorithm
9. A.T. Dinh, “Metrics for clustering”
https://dinhanhthi.com/metrics-for-clustering
10. EPA, Technical Assistance Document for the Reporting of Daily Air Quality – the Air Quality Index (AQI), 2018
https://www.airnow.gov/sites/default/files/2020-05/aqi-technical-assistance-document-sept2018.pdf
11. Dz.U. 2012 poz. 1031, Rozporządzenie Ministra Środowiska w sprawie poziomów niektórych substancji w powietrzu
http://isap.sejm.gov.pl/isap.nsf/DocDetails.xsp?id=WDU20120001031
12. W.P. Bejeck Jr., "Kafka Streams in Action", 2018, Manning Publications, ISBN 9781617294471

Thank you!

WorldRemit