인공지능개발에 유용한 오픈소스 프로젝트들

DeepNumbers Inc.

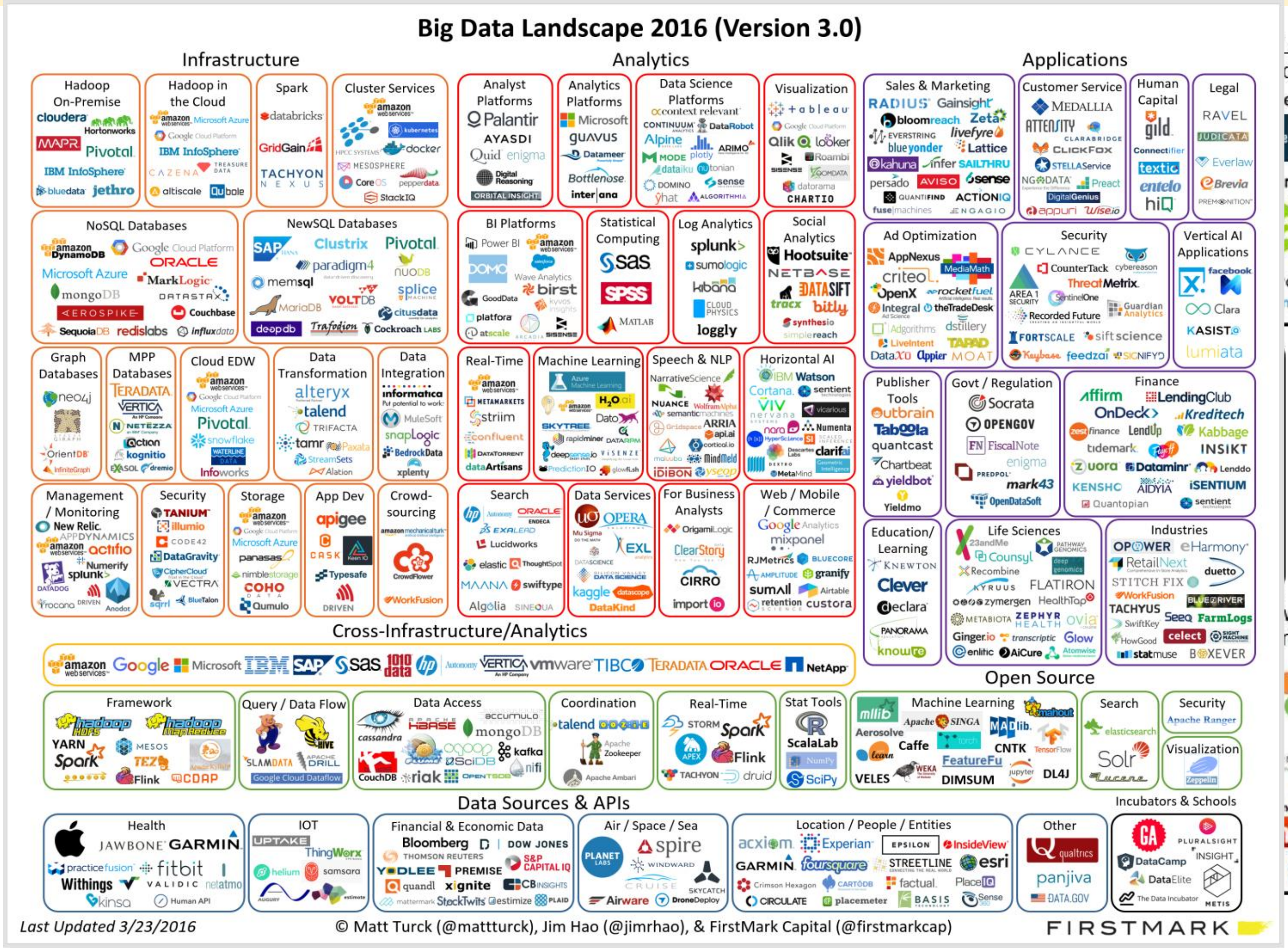
안명호

인공지능 기술의 폭발

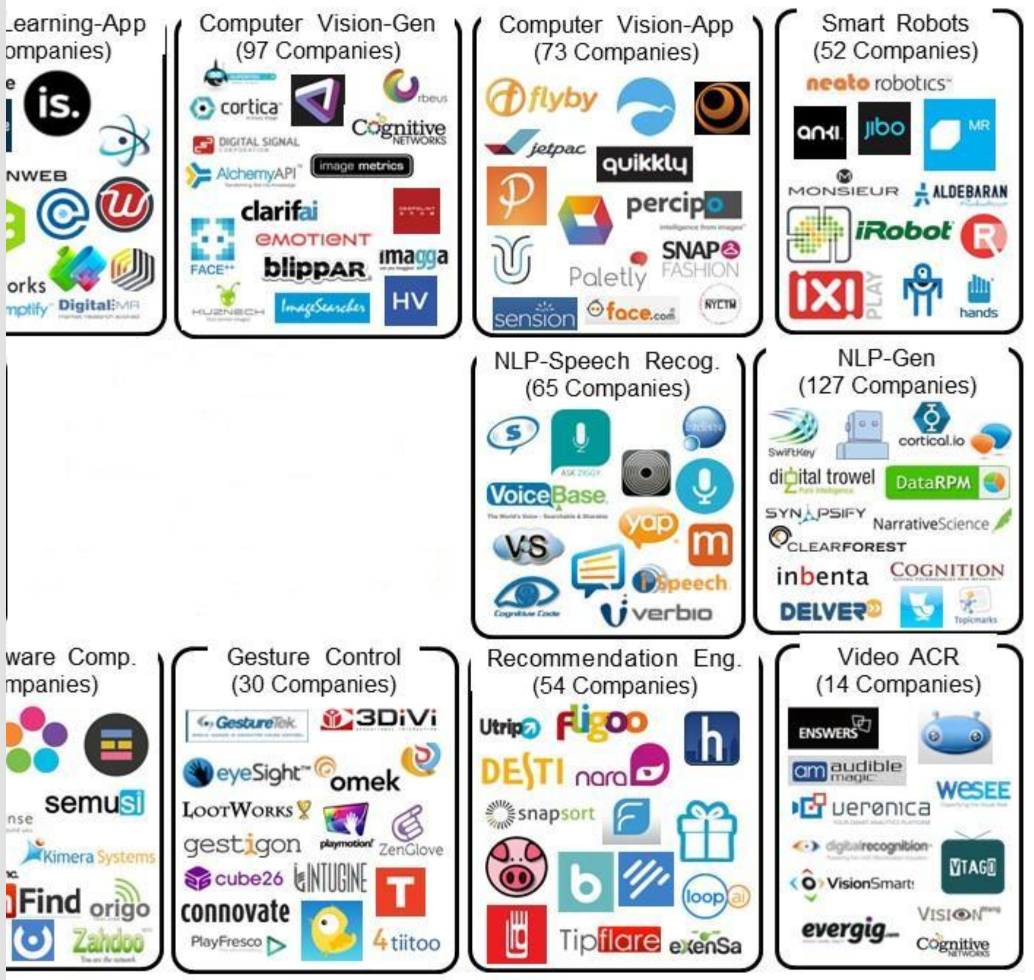
AlphaGo

No Question!!!

관련 오픈소스의 창궐!!!



http://mattturck.com/big-data-landscape/



http://www.datascienceassn.org/content/machine-learning-firm-landscape-2016

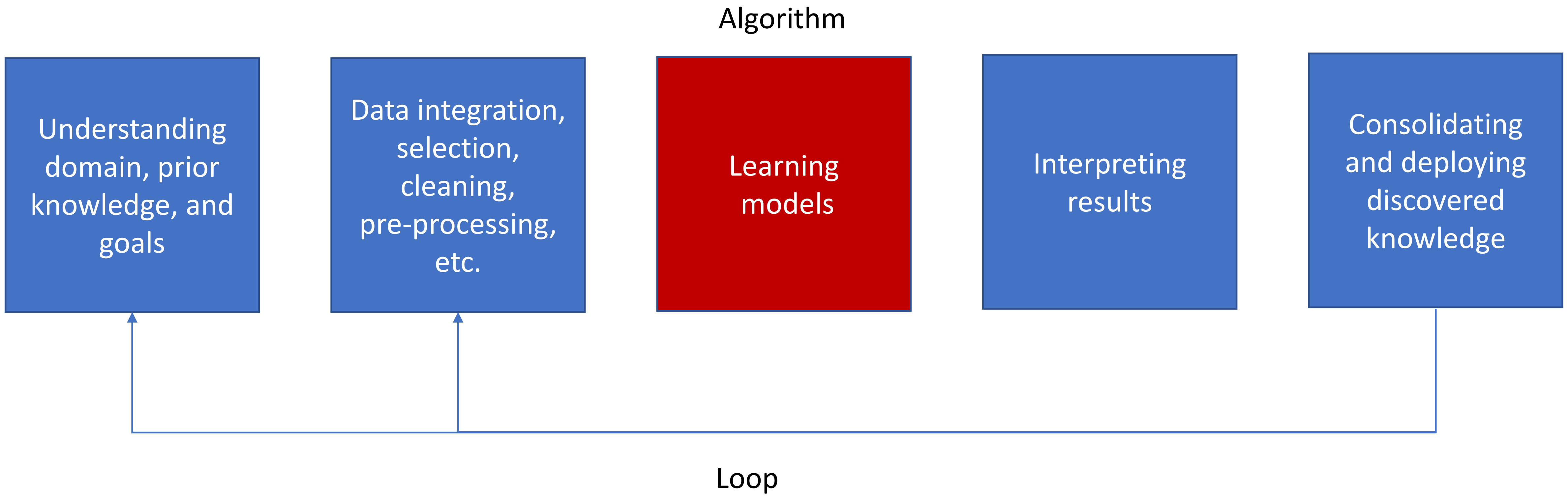
무엇이 중요한가?

인공지능개발?

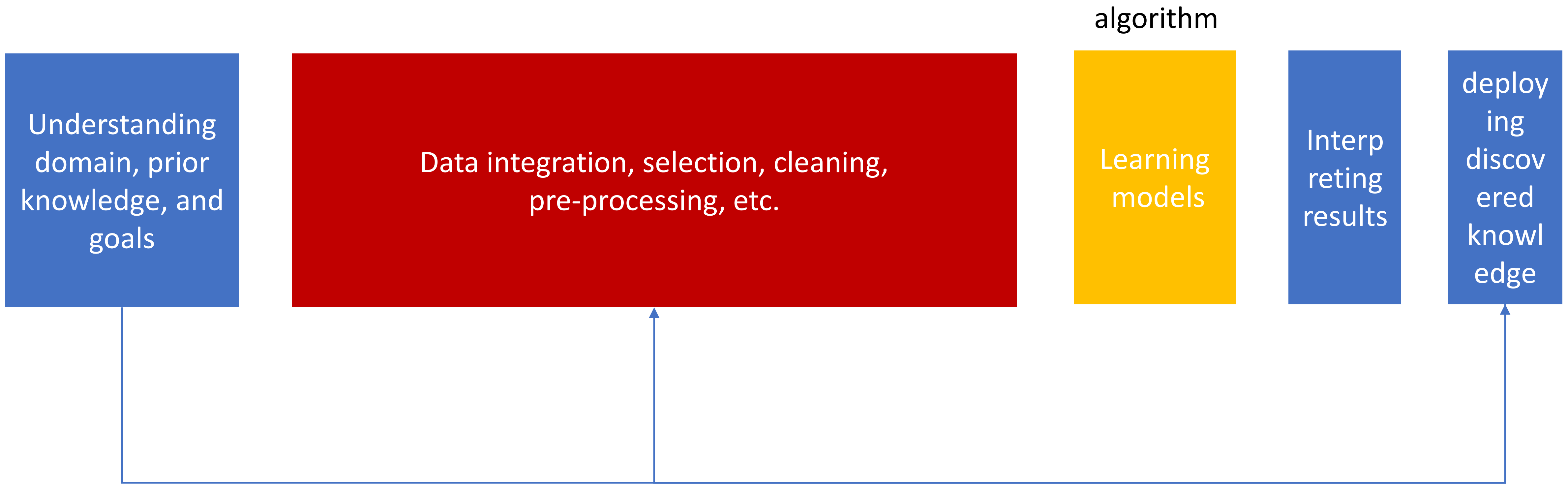
정원을 가꾸는 것과 유사하게 노력과 시간이 많이 필요함

- Seeds = Algorithms
- Nutrients = Data
- Gardener = You
- Plants = Programs

인공지능 개발 전체 프로세스

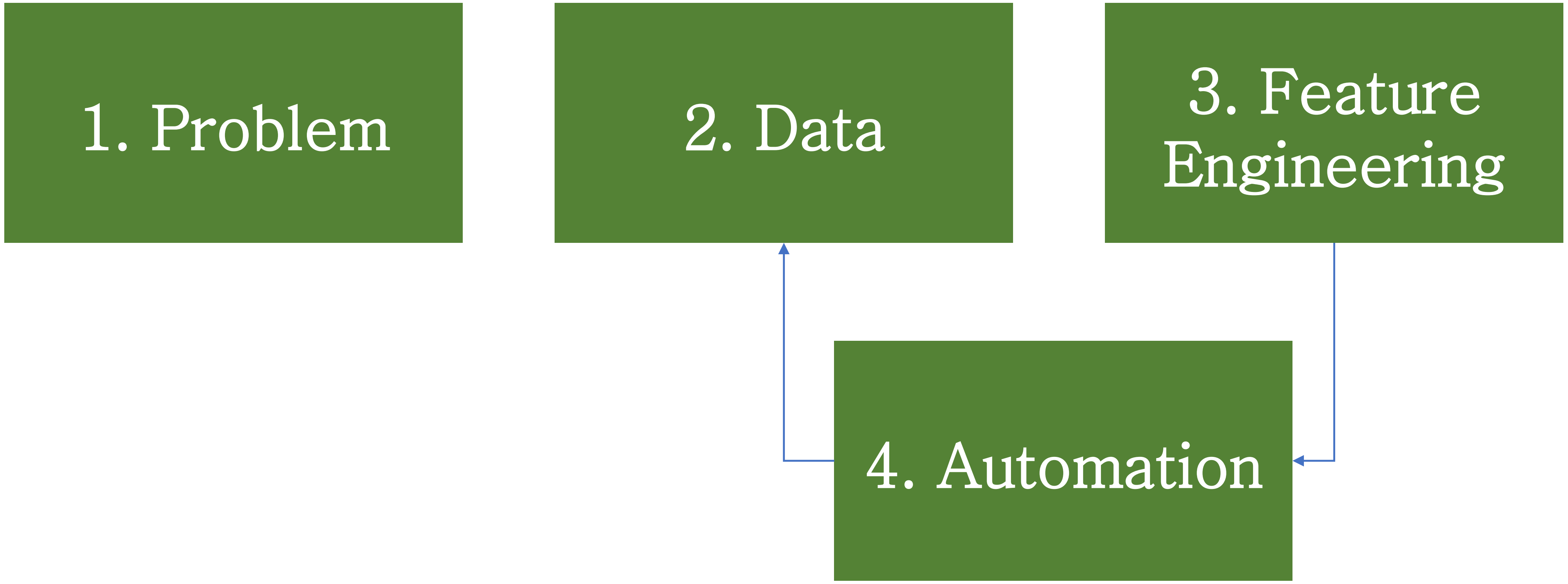

Algorithm
Understanding domain, prior knowledge, and goals
Data integration, selection, cleaning, pre-processing, etc.
Learning models
Interpreting results
Consolidating and deploying discovered knowledge
Loop

인공지능 개발 실제 프로세스


algorithm
Understanding domain, prior knowledge, and goals
Data integration, selection, cleaning, pre-processing, etc.
Learning models
Interpreting results
deploying discovered knowledge

Infinite loop

좋은 결과물을 만들기 위한 요소

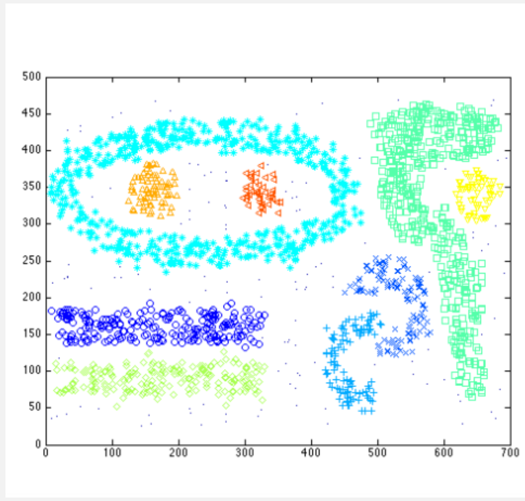


1. Problem

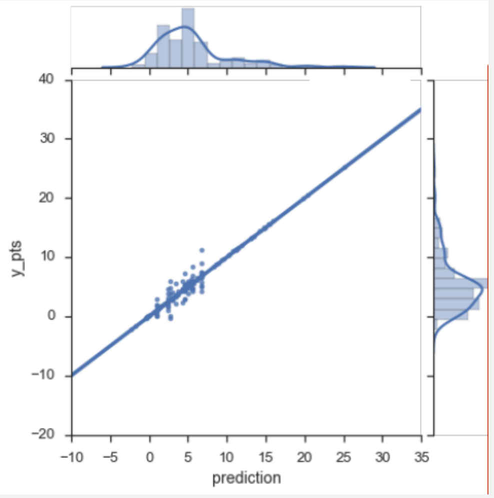
인공지능기술에 적합한 문제인가?

Deep Learning을 사용해야 하는가?

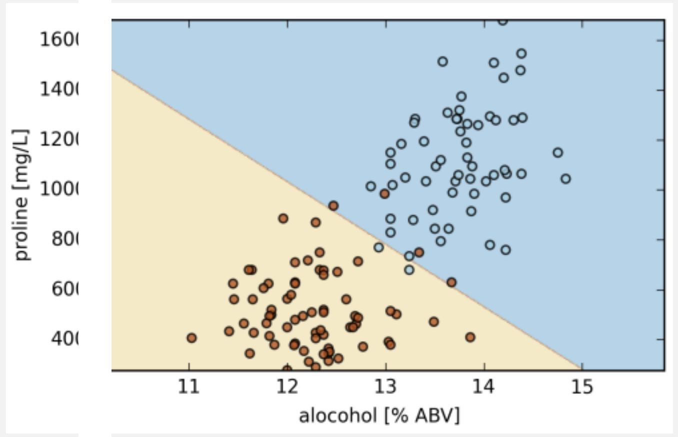
인공지능기술이 할 수 있는 3가지



Clustering



Regression



Classification

2. Data

문제에 연관된 데이터를 충분히 가지고 있는가?

No Data,

No Machine Learning

3. Feature Engineering

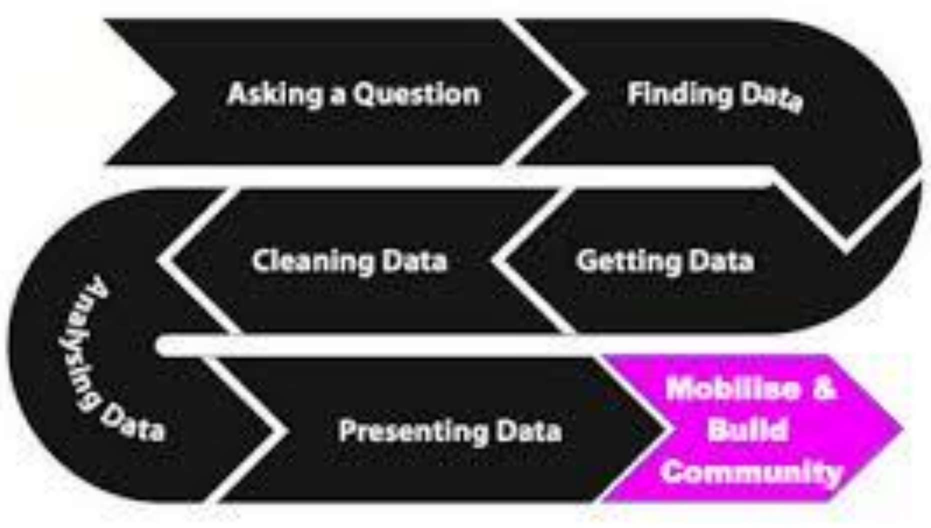
확보한 데이터는 학습에 적합한가?

Garbage In

,Garbage Out

4. Automation

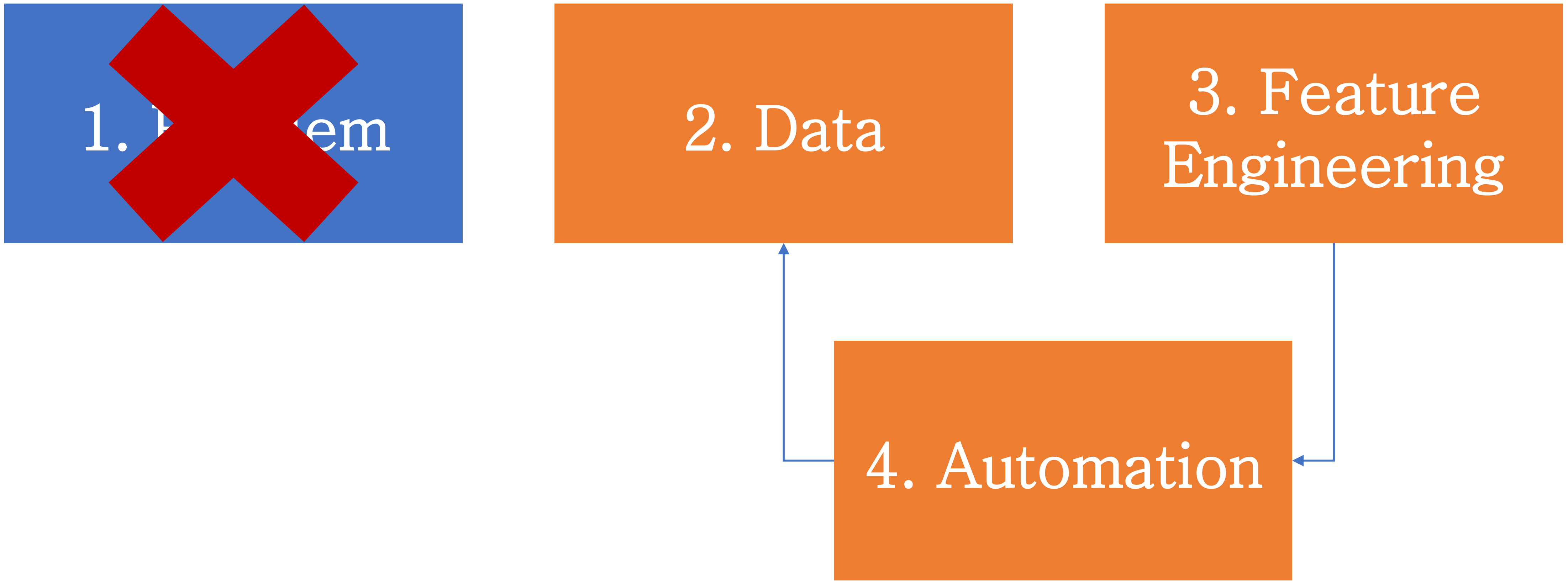
복잡한 과정을 효율적으로 처리할 수 있는가?



Automated Workflow!!!

<u>Infinite loop</u>

좋은 결과물을 만들기 위해 필요한 기술들



필요한 기술 : Data

역할	다양한 데이터소스에서 대용량 데이터를 수집하고 가공할 수 있어야 함
Data Model	• Streaming, Batch Processing을 지원하는가?
Guarantee	• 태스크 실행종료를 담보할 수 있는가?
Fault-Tolerance	• 에러발생시 자동으로 복구할 수 있는가?
Programming Interface	• 다양한 종류의 언어를 지원하는가?
Scalability	• 시스템 확장을 할 수 있는가?
Latency	• 시스템에 의한 지연시간이 긴가?
Maturity	• 기술은 충분히 성숙되어 있고, 커뮤니티도 활발한가?

Data를 위한 오픈소스 프로젝트

APACHE Spark™

APACHE STORM™

Flink

samza

Apache Spark

설명	Fast and General engine for large-scale data processing
특징	• Speed : 100x faster than Hadoop • Ease of use : java, scala, python, R • Generality : SQL, Streaming, Batch • Runs Everywhere : HDFS, Hbase, S3, Cassandra • Strong Community : !!!
강점	• 통합된 고급 데이터 분석 기능 제공 • 데이터 병렬 처리 가능 • MapReduce보다 더 빠르고 효율적인 기능 제공 • Streaming API를 이용한 연속적인 micro-batch 처리 가능 • Machine Learning 기능제공(MLLib)

Apache Flink

설명	Streaming processing framework for distributed, high-performing, always available and accurate data streaming applications
특징	• Streaming-First : Continuous processing • Fault-tolerant : Stateful computations • Scalable : up to 1000s of nodes and beyond • Performance : High throughput, low latency
강점	• 모든 데이터를 Streaming Data로 처리하는 진정한 Streaming Framework • 매우 빠른 데이터 처리속도 • MapReduce와 직접적 연동가능 • 대규모 클러스터 운영 가능 • Machine Learning 기능 제공 (FlinkML)

Apache Storm

APACHE STORM™

설명	Distributed realtime computation system
특징	• Guaranteed data processing • Horizontal scalability • Fault-tolerance • High level abstraction
강점	• DAG(Directed Acyclic Graph)지원으로 유연한 Workflow • 다양한 프로그래밍 언어와 연동 가능 • 대규모 클러스터 운영 가능 • Machine Learning 기능 제공 (SAMOA)

Apache Storm

samza

설명	Distributed stream processing framework
특징	• Simple API • Managed State • Fault-tolerance • Scalability • Pluggable
강점	• Kfaka, YARN을 사용해 클러스터 운영이 용이함 • 리소스관리와 보안 기능을 제공함 • 대규모 클러스터 운영 가능 • CGA(Call Graph Assembly)기반의 Workflow 기능제공 • LinkedIn에서 헌납한 오픈소스 프로젝트

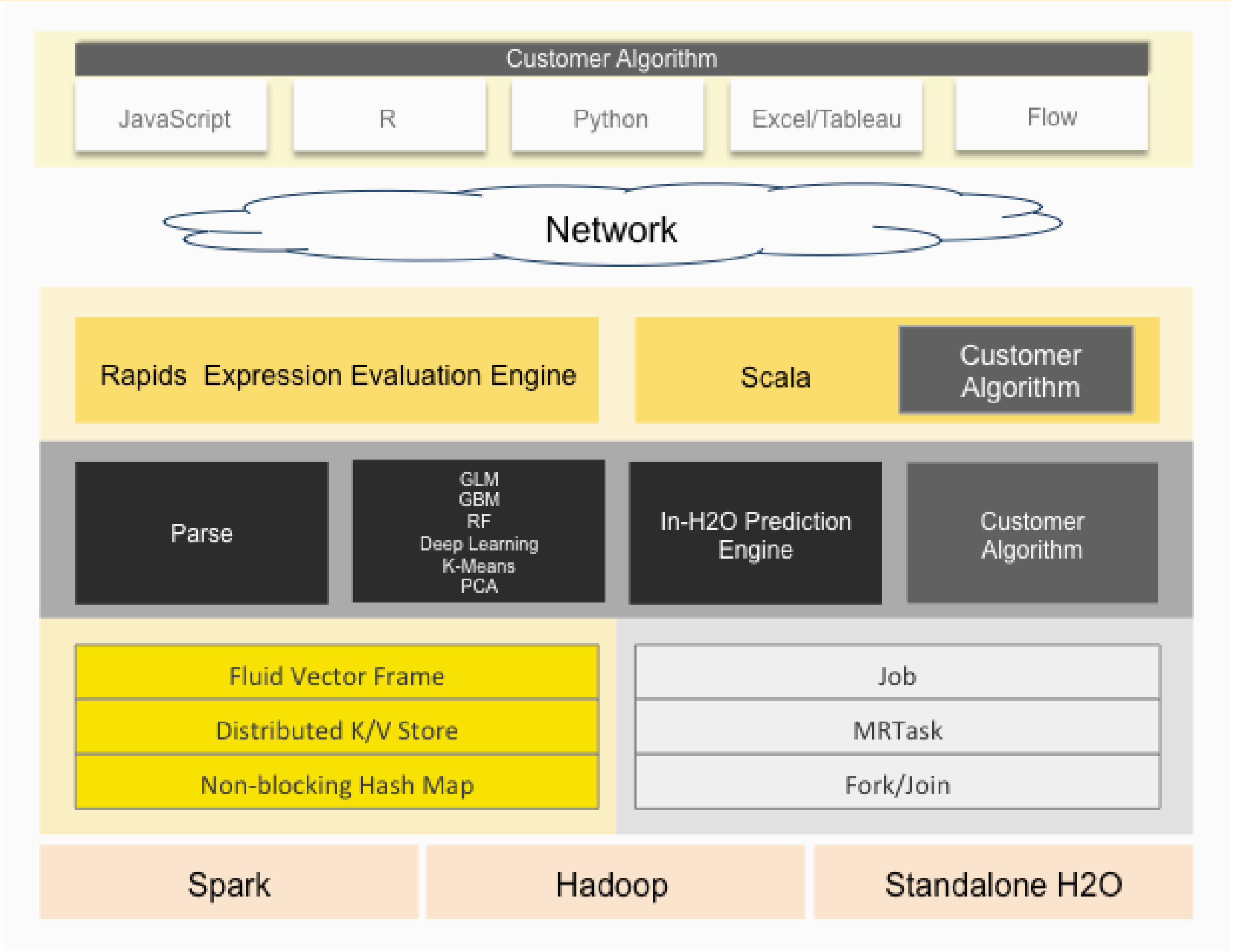
Spark vs Storm vs Flink VS Samza

	Spark	Storm	Flink	Samza
Data	Streaming, Batching	Streaming, Batching	Streaming, Batching	Streaming, Batching
Guarantee	At-least-once	Exactly-once	Exactly-once	At-least-once
Fault-Tolerance	RDD Checkpoint	Record ACK	Checkpoint	Log
Throughput	High	Low	High	High
Latency	High	Low	Low	Low
Maturity	High	High	Low	Medium

Dedicated ML Framework : H20

설명	Dedicated Machine Learning Framework
특징	• Best of breed Open source Technology • Easy to use Web UI and interface • Data agnostic support for all common database and file types • Massively scalable big data analysis • Real-time data scoring
강점	• ML에 특화된 프레임워크 • Web UI를 이용해 ML 프로그램 개발가능 • 다양한 언어지원(R, Python, Java, Scala) • 최적화된 ML 알고리즘 제공 • In-memory processing을 이용한 빠른 처리 속도

H2O – Software Stack



필요한 기술 : Feature Engineering

ML 알고리즘에 적합한 형태로 데이터를 가공

EDA

- 데이터의 탐색 및 분석을 통한 문제 이해도 향상
- 통계기능 필요
- 시각화가 중요

Preprocessing

- 사용하려는 알고리즘의 특성에 맞도록 데이터 처리
- 기존 데이터를 조합한 새로운 데이터 생성

Feature Selection

- 최적의 결과가 예상되는 데이터 선택
- Curse of Dimensionality

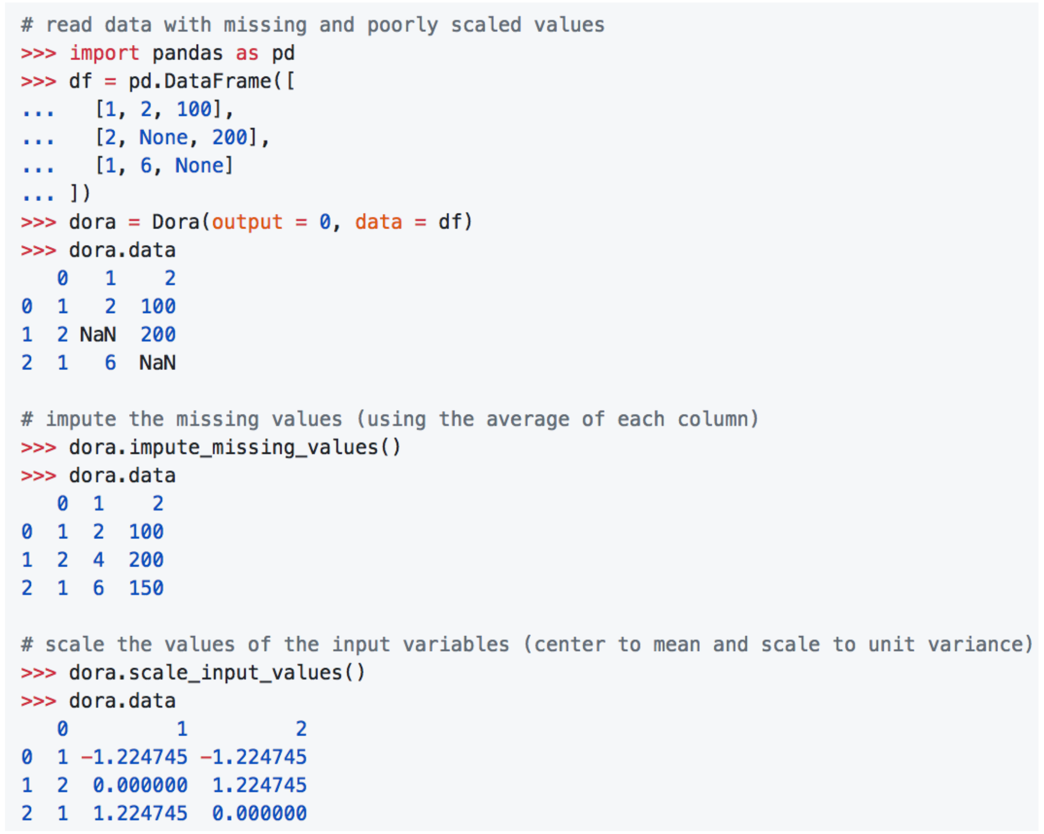
SK-Learn

설명	ML을 위한 종합선물세트(Deep Learning을 제외)
특징	• Simple and efficient tools for data mining and data analysis • Accessible to everybody, and reusable in various contexts • Built on NumPy, SciPy, and matplotlib • Open source, commercially usable – BSD license • Machine Learning in Python
강점	• Classification, Regression, Clustering • Dimensionality Reduction • Model Selection • Preprocessing • 강력한 커뮤니티와 풍부한 자료 • 학계와 산업계가 모두 참가해 이론과 실제가 검증된 기술 포함

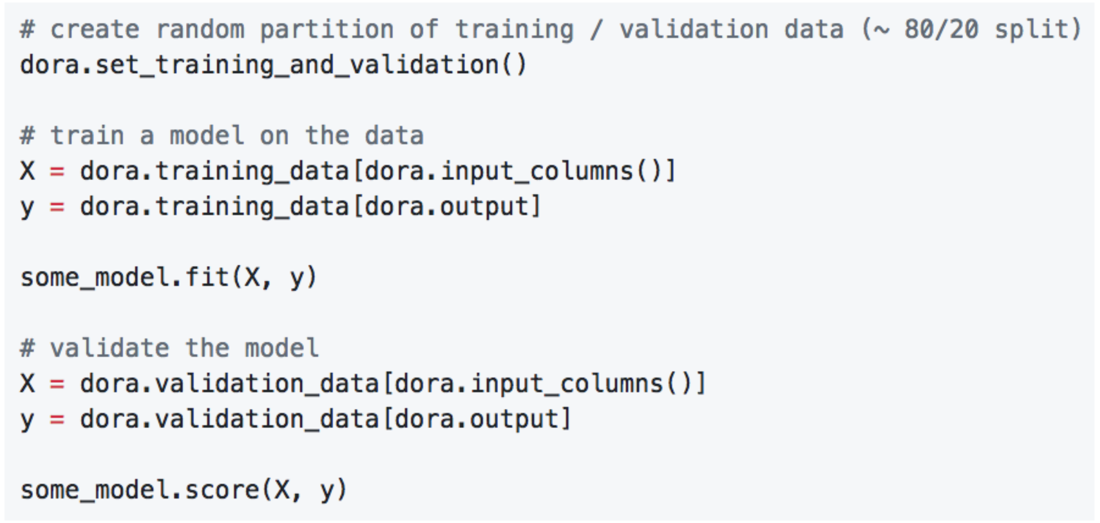
DORA

Designed to automate the painful parts of exploratory data analysis

data cleaning, feature selection & extraction, visualization, partitioning data for model validation



```
# read data with missing and poorly scaled values
>>> import pandas as pd
>>> df = pd.DataFrame([
...     [1, 2, 100],
...     [2, None, 200],
...     [1, 6, None]
... ])
>>> dora = Dora(output = 0, data = df)
>>> dora.data
   0    1    2
0  1    2  100
1  2  NaN  200
2  1    6  NaN

# impute the missing values (using the average of each column)
>>> dora.impute_missing_values()
>>> dora.data
   0  1    2
0  1  2  100
1  2  4  200
2  1  6  150

# scale the values of the input variables (center to mean and scale to unit variance)
>>> dora.scale_input_values()
>>> dora.data
   0         1         2
0  1 -1.224745 -1.224745
1  2  0.000000  1.224745
2  1  1.224745  0.000000
```

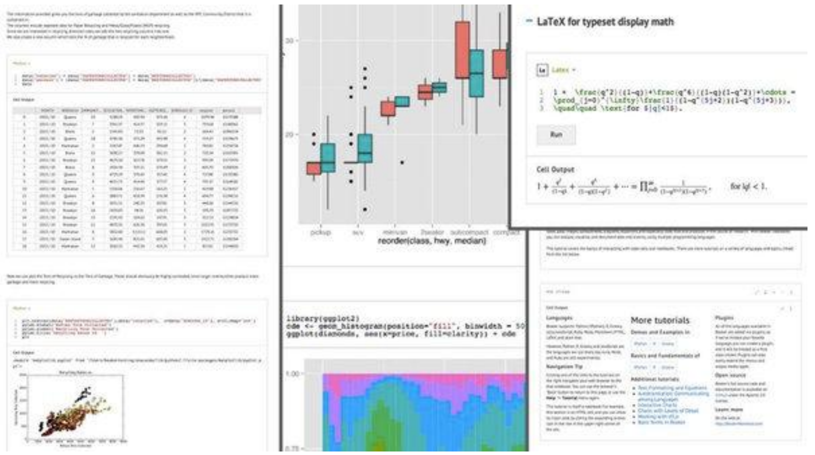


```
# create random partition of training / validation data (~ 80/20 split)
dora.set_training_and_validation()

# train a model on the data
X = dora.training_data[dora.input_columns()]
y = dora.training_data[dora.output]

some_model.fit(X, y)

# validate the model
X = dora.validation_data[dora.input_columns()]
y = dora.validation_data[dora.output]

some_model.score(X, y)
```

https://github.com/NathanEpstein/Dora

Beaker

notebook-style development environment for working interactively with large and complex datasets.



```
\frac{q^2}{(1-q)}+\frac{q^6}{(1-q)(1-q^2)}+\cdots =
\prod_{j=0}^{\infty}\frac{1}{(1-q^{5j+2})(1-q^{5j+3})},
\quad\quad \text{for $|q|<1$}.
```

```
library(ggplot2)
cde <- geom_histogram(position="fill", binwidth = 50
ggplot(diamonds, aes(x=price, fill=clarity)) + cde
```

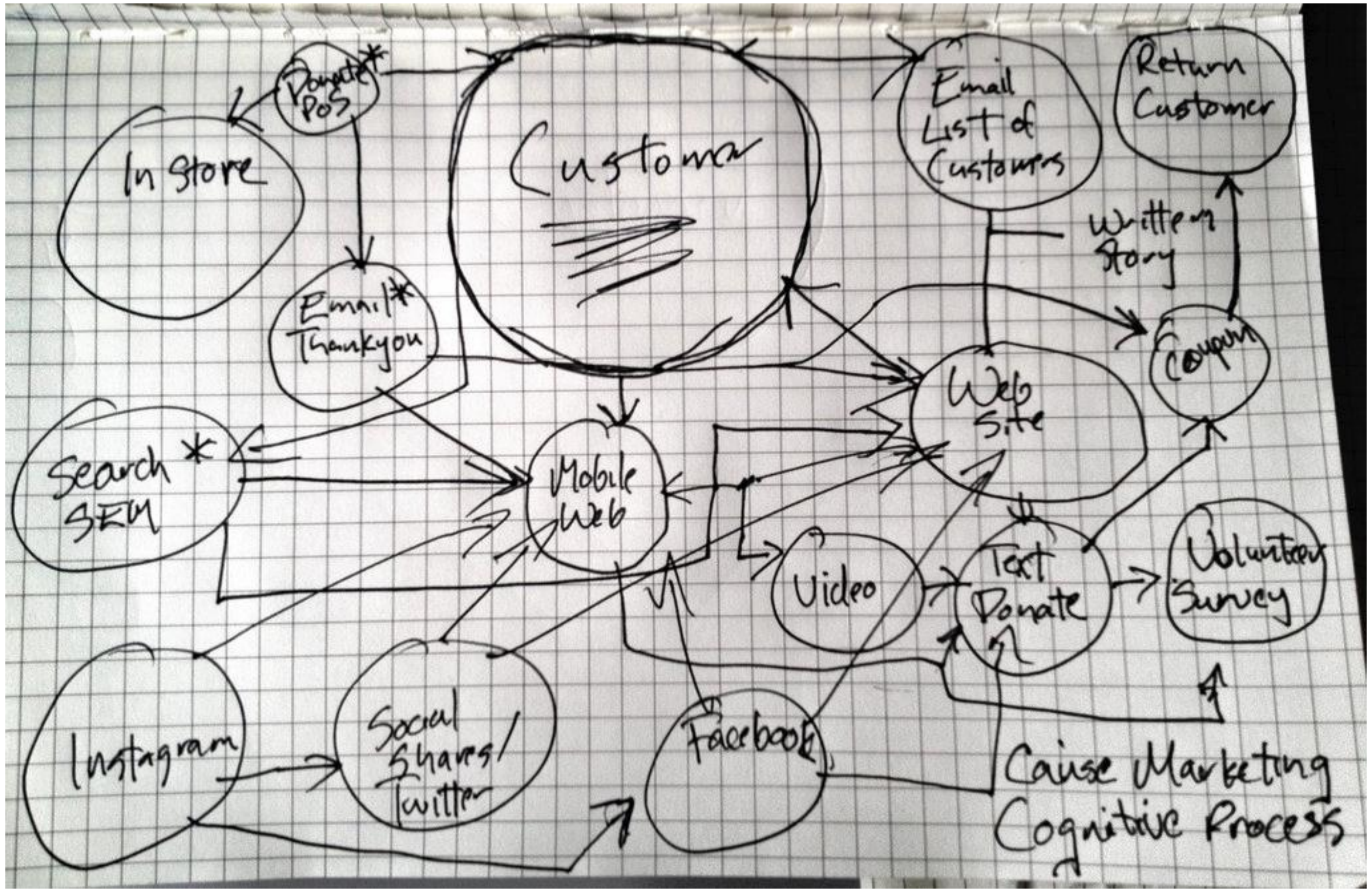
- Multi-Language Support
- Rich Visualization – D3,bokeh, matplotlib, ggplot2
- Notebooks enable iterative Exploration
- Translate Data between languages
- Advanced markup and Hierarchical Organization

An Open Source from Top-notch Hedge Fund

http://beakernotebook.com/

필요한 기술 : Automation

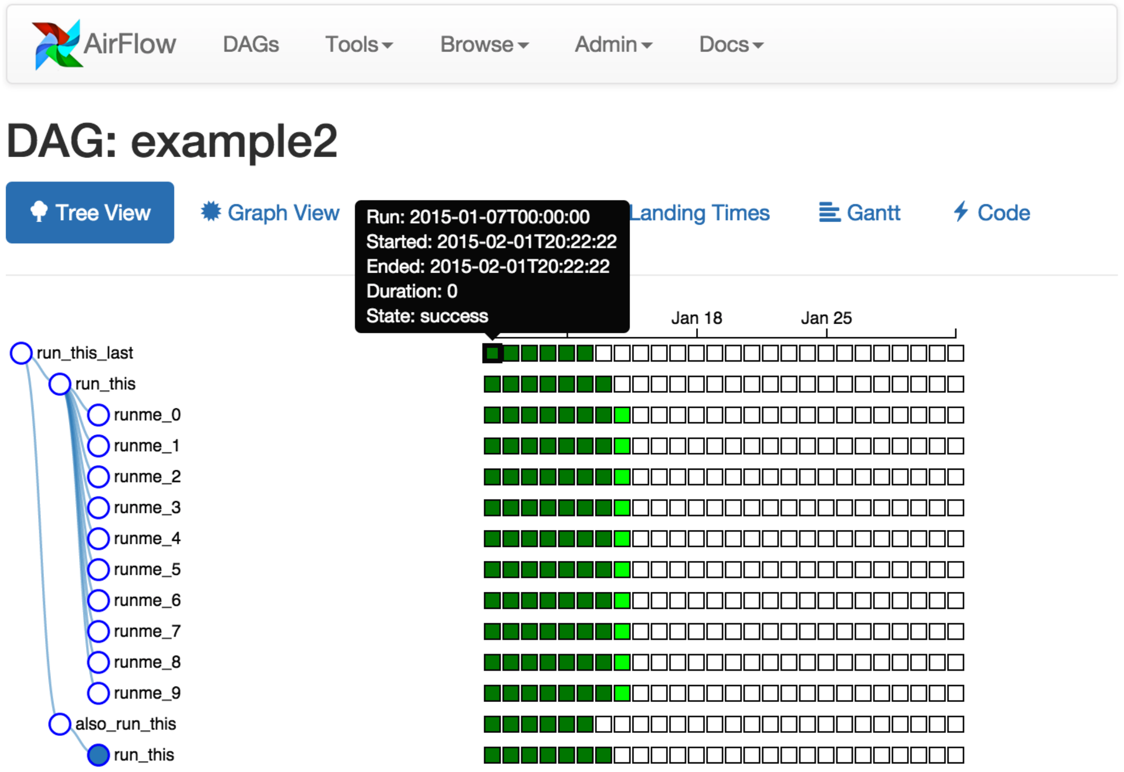
ML을 위한 데이터 수집, 처리 및 평가등의 과정은 생각보다 복잡



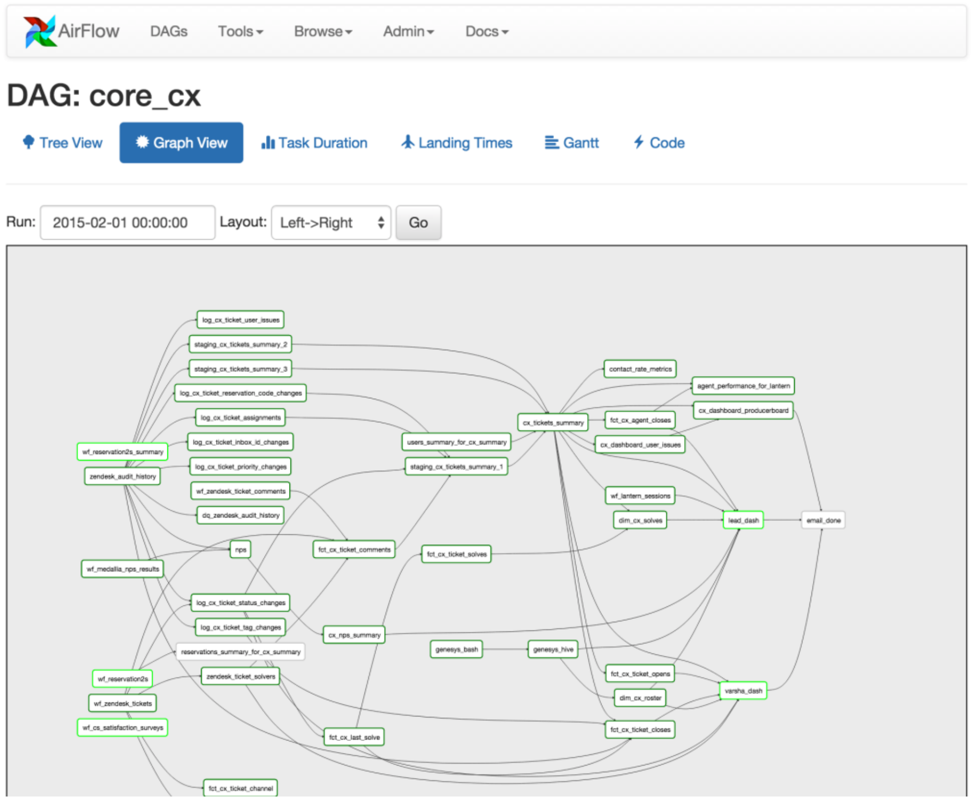
Airflow

설명	To programmaticaly author, schedule and monitor data pipelines
특징	• Dynamic – configuration as python code • Extensible – Easily define your own operators, executors • Elegant – Lean and explicit • Scalable – Modular architecture
강점	• Apache Incubating Project, Initiated by AirBnB • DAG(Directed Acyclic Graphs)를 이용한 workflow management • Rich WEB user interface(Monitoring, Task Status) • Can Work with Apache Spark

Airflow



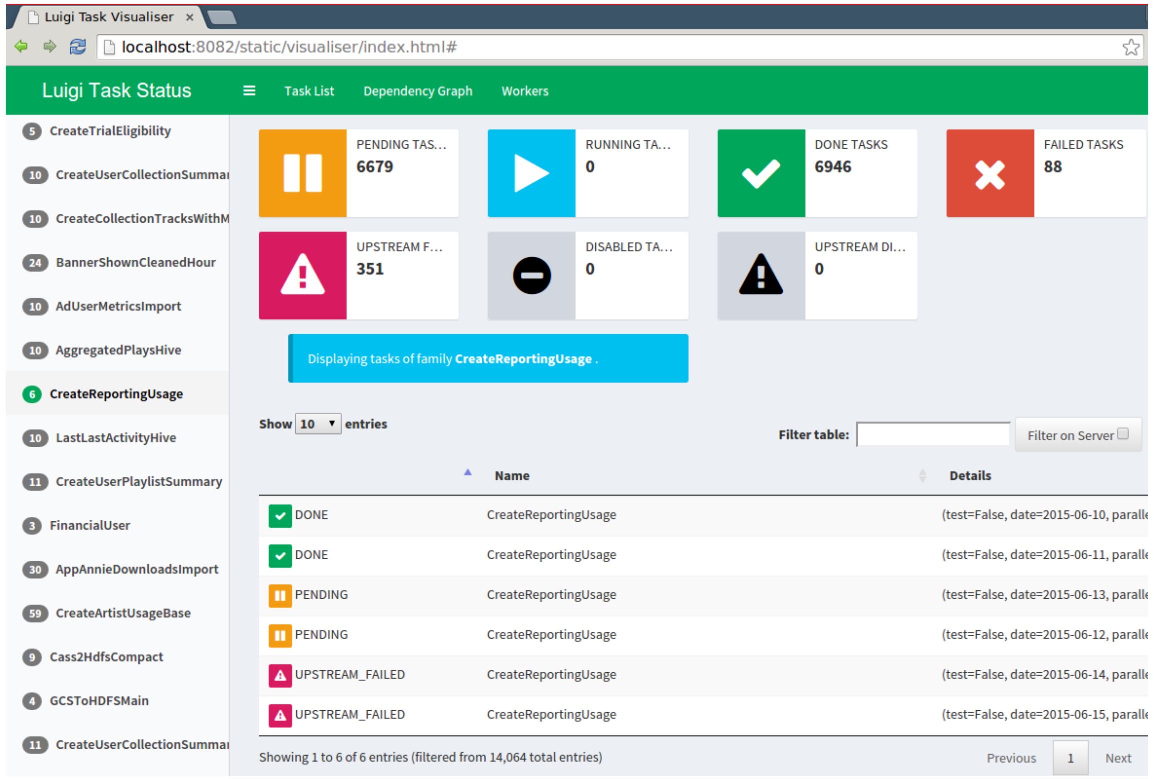
```
Run: 2015-01-07T00:00:00
Started: 2015-02-01T20:22:22
Ended: 2015-02-01T20:22:22
Duration: 0
State: success
```



Luigi

설명	A package that helps you build complex pipelines of batch jobs
특징	• Allows you to parallelize workflows as needed • Toolbox with common task templates • Supports Python mapreduce jobs in Hadoop, Hive, and Pig • Includes file system abstraction for Hadoop Distributed File System and local files that ensure all systems are atomic, preventing them from crashing in a state containing partial data
강점	• Open Source Initiated by Spotify • Matured Technology(Since 2011) & good references • Rich WEB user interface(Monitoring, Task Status) • Python based workflow management • Relatively simple workflow management

Luigi



Luigi Task Visualiser

localhost:8082/static/visualiser/index.html#

Luigi Task Status · Task List · Dependency Graph · Workers

5	CreateTrialEligibility
10	CreateUserCollectionSummai
10	CreateCollectionTracksWithM
24	BannerShownCleanedHour
10	AdUserMetricsImport
10	AggregatedPlaysHive
6	CreateReportingUsage
10	LastLastActivityHive
11	CreateUserPlaylistSummary
3	FinancialUser
30	AppAnnieDownloadsImport
59	CreateArtistUsageBase
9	Cass2HdfsCompact
4	GCSToHDFSMain
11	CreateUserCollectionSummai

PENDING TAS...	6679
RUNNING TA...	0
DONE TASKS	6946
FAILED TASKS	88
UPSTREAM F...	351
DISABLED TA...	0
UPSTREAM DI...	0

Displaying tasks of family **CreateReportingUsage** .

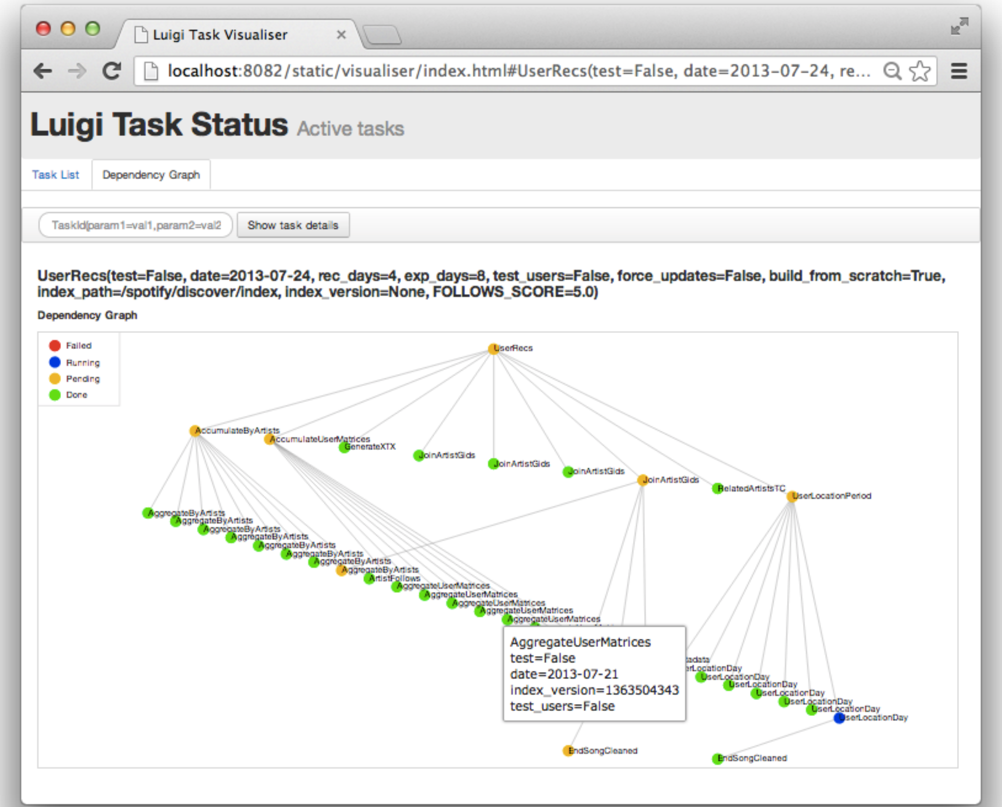
Show 10 entries

Filter table: Filter on Server

	Name	Details
DONE	CreateReportingUsage	(test=False, date=2015-06-10, paralle
DONE	CreateReportingUsage	(test=False, date=2015-06-11, paralle
PENDING	CreateReportingUsage	(test=False, date=2015-06-13, paralle
PENDING	CreateReportingUsage	(test=False, date=2015-06-12, paralle
UPSTREAM_FAILED	CreateReportingUsage	(test=False, date=2015-06-14, paralle
UPSTREAM_FAILED	CreateReportingUsage	(test=False, date=2015-06-15, paralle

Showing 1 to 6 of 6 entries (filtered from 14,064 total entries)

Previous 1 Next



Luigi Task Visualiser

localhost:8082/static/visualiser/index.html#UserRecs(test=False, date=2013-07-24, re...

Luigi Task Status Active tasks

Task List · Dependency Graph

TaskId(param1=val1,param2=val2 · Show task details

UserRecs(test=False, date=2013-07-24, rec_days=4, exp_days=8, test_users=False, force_updates=False, build_from_scratch=True, index_path=/spotify/discover/index, index_version=None, FOLLOWS_SCORE=5.0)

Dependency Graph

AggregateUserMatrices
test=False
date=2013-07-21
index_version=1363504343
test_users=False

Airflow vs Luigi

	Airflow	Luigi
Code	Python	Python
Web UI	Rich	Minimal
Calendar Scheduling	Local Scheduler	Cron
Parallelism	Workers	Workers
DAG	Yes	No
Task Synchronization	Yes	No
Track Historys	Yes	Yes, DB
State mgt	Kindof	Yes, DB
Maturity	Medium	High

오픈소스 선택의 핵심은 Community, 아마도..

현 상황은 ML 오픈소스가 선캄브리아 시대처럼 폭발하는 난감한 상황

<u>오픈소스의 선택은 현재가 아닌,</u>

<u>미래를 위한 선택</u>

<u>특별한 이유가 없다면</u>

<u>열성적인 Community가 주요 기준</u>

감사합니다.