Python + Spark

머신러닝을 위한 완벽한 결혼

James Ahn

DeepNumbers

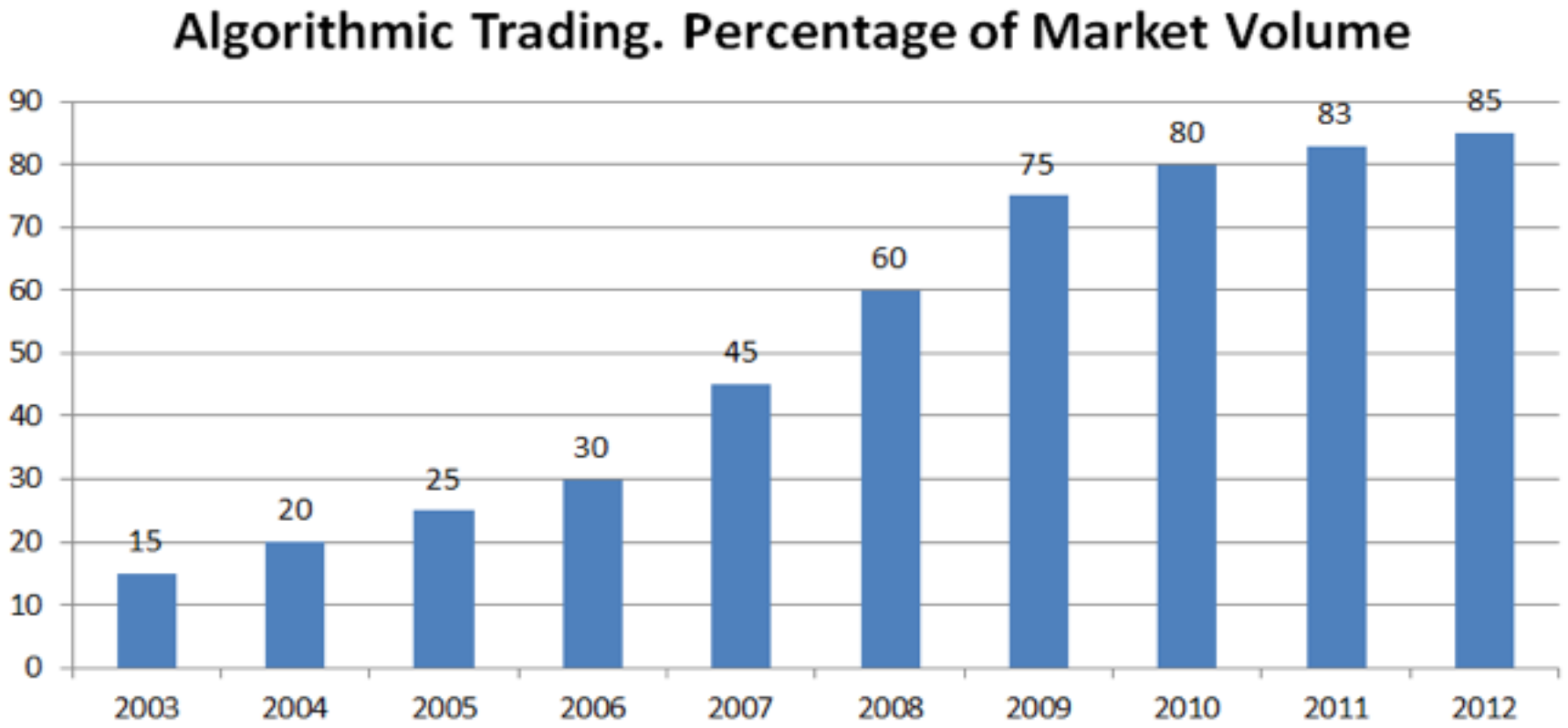
The Problem

Algorithm Trading?

- 알고리즘 트레이딩은 수학적 계산과 IT 시스템을 이용해 트레이딩을 하는 것으로 시스템 트레이딩, Algo Trading 혹은 Blackbox 트레이딩이라고 한다.
- 알고리즘 트레이딩은 투자은행, 연기금, 헤지펀드, 증권회사등 많은 곳에서 사용되고 있으며, 최근 몇 년사이에는 수학적지식과 IT지식을 가진 개인들도 많이 참여하고 있다.

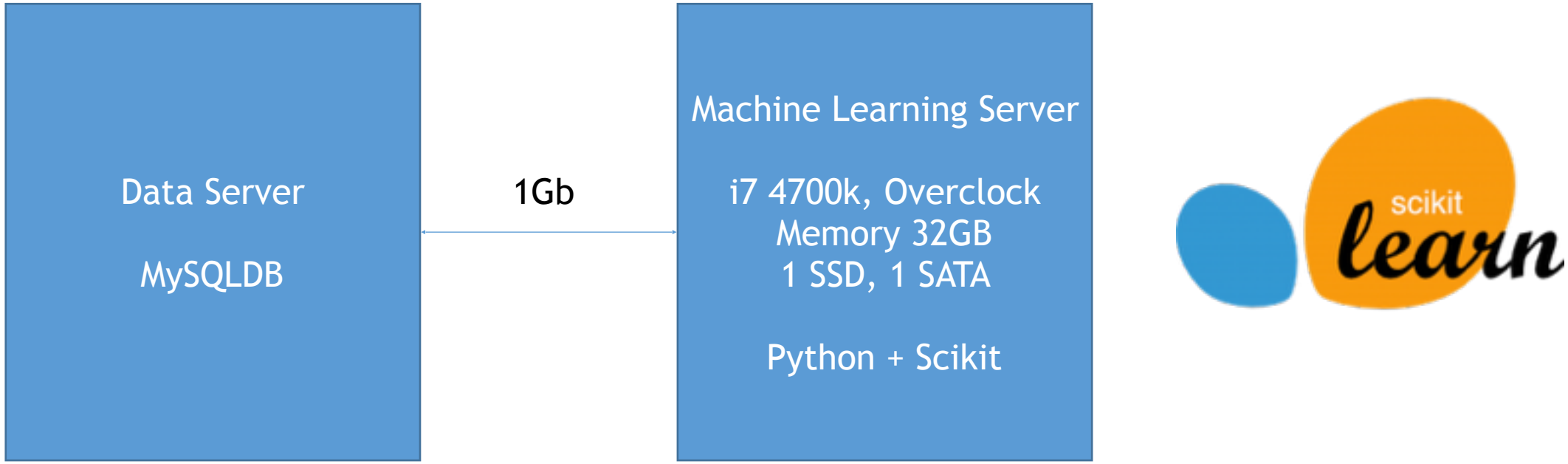
Algorithm Trading의 현재

미국은 2012년 알고리즘 트레이딩 거래량이 85%에 달할만큼 알고리즘 트레이딩은 가파르게 증가



당시의 개발환경

처음에는 아래의 환경으로도 즐거울 수 있었다.



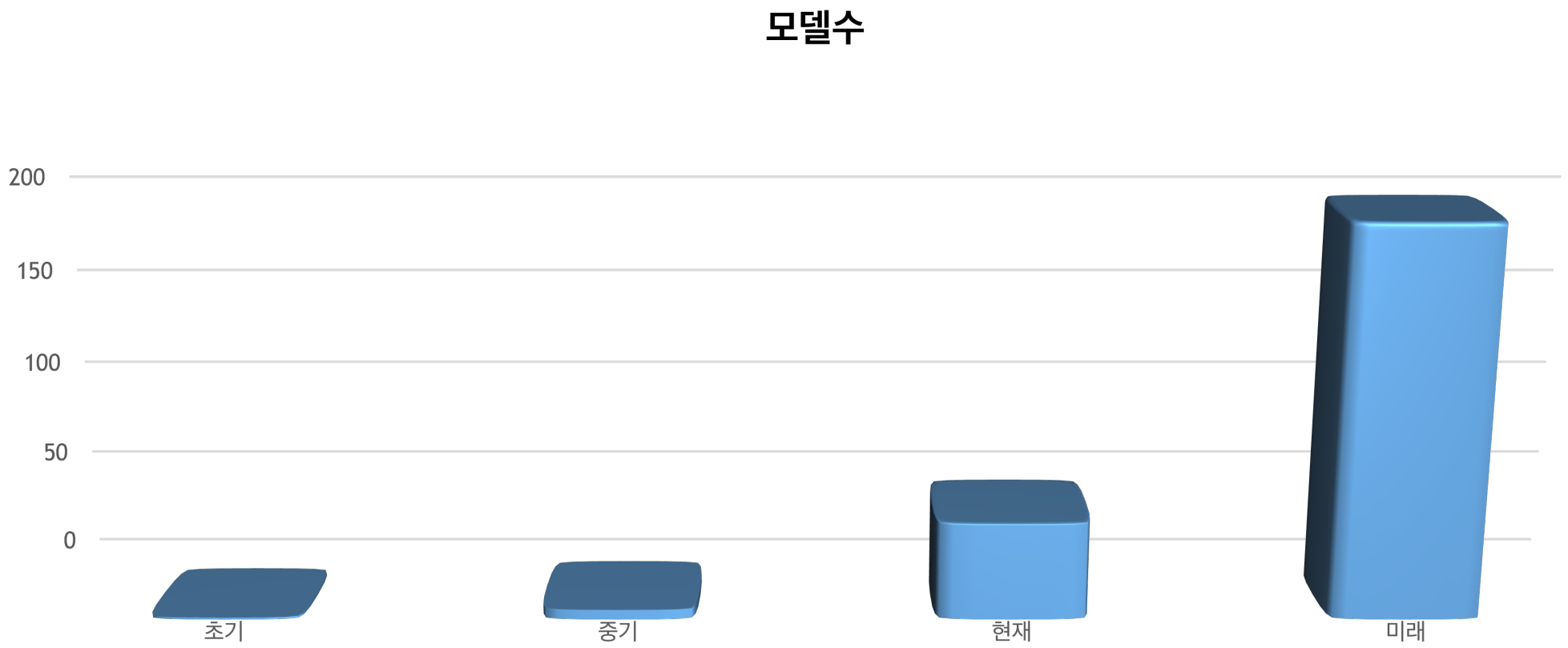
문제는 학습시간!!!

머신러닝 모델들을 학습시키는데 대략 1주일정도의 시간이 소요되었다.!!!

머신러닝 지식이 늘어남에 따라 모델증가

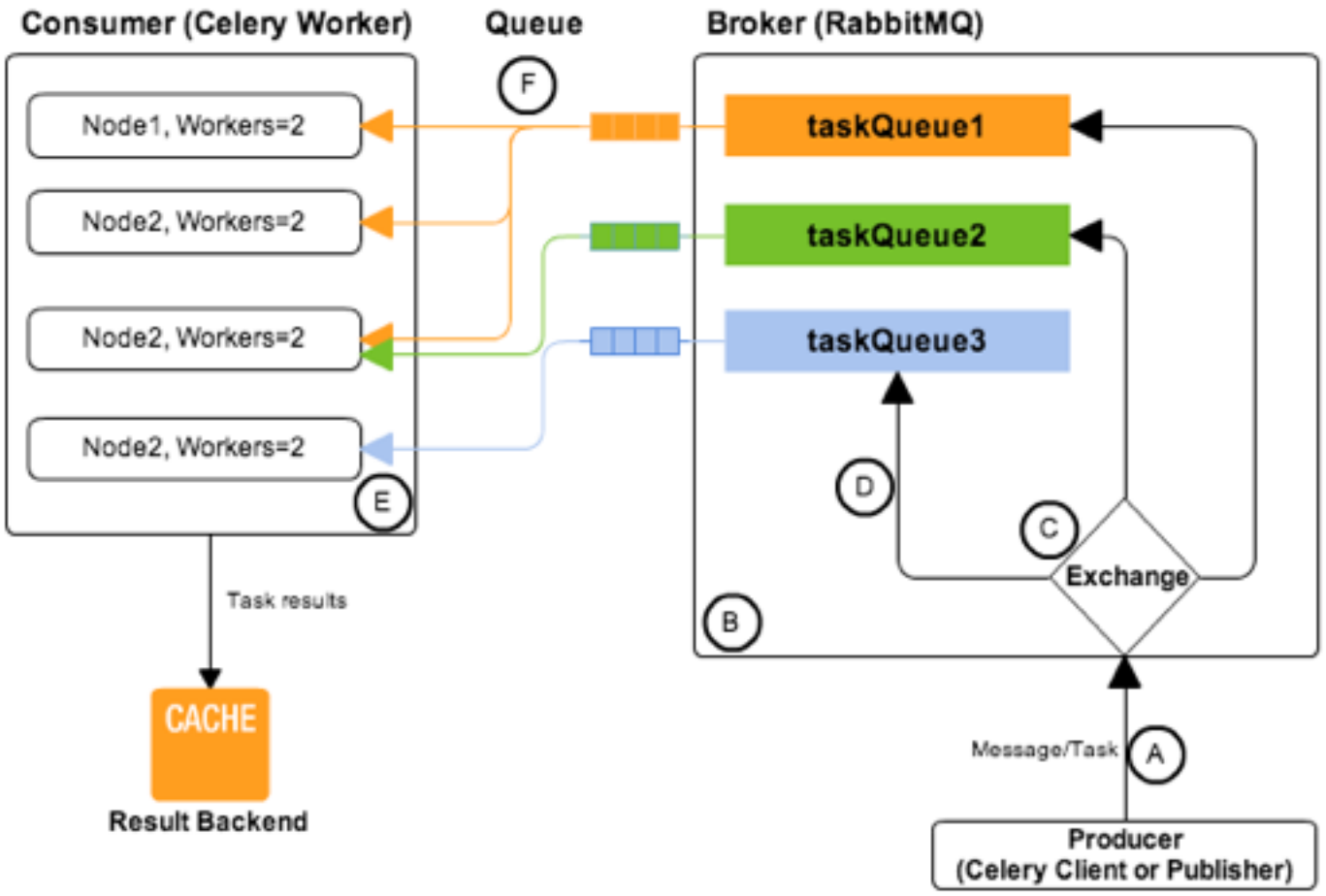
잔기술이 증가하니, 자연스럽게 많은 모델들이 생겨나고 학습시간이 비약적으로 증가하였다!!!

시간이 지날수록 더욱 잡다한 모델들이 생겨날 것으로 강력하게 예상된다.



First Trial

Powered By Celery & RSync



Celery를 이용한 작업의 분산

그리고

Rsync를 이용한 결과 파일의 업데이트

Celery = Distributed Async Task Queue

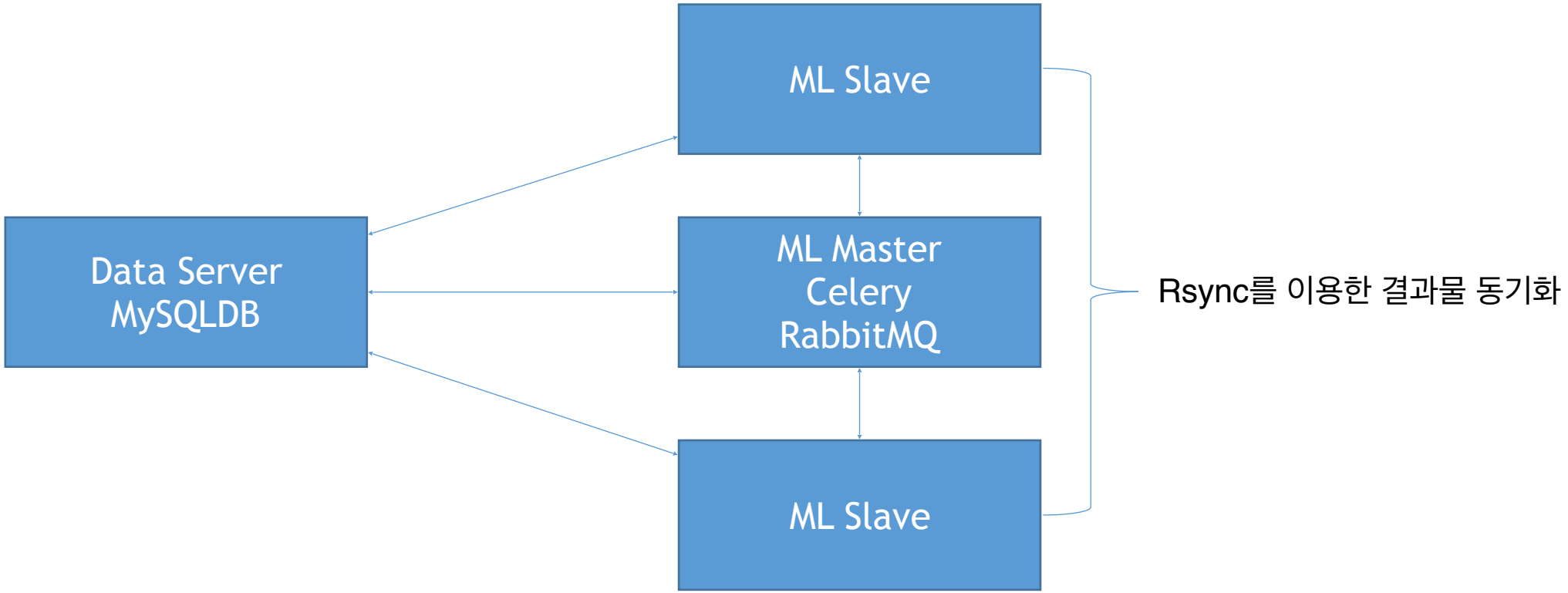
긴 실행시간을 필요로 하는 작업

백그라운드로 실행할 필요가 있는 작업

주기적으로 실행할 필요가 있는 작업

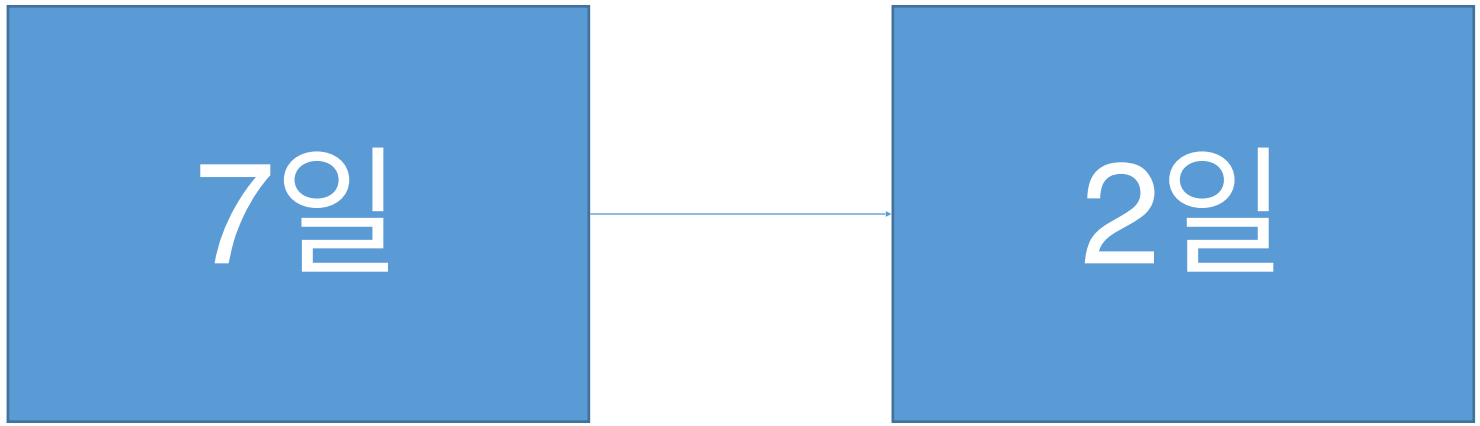
개선된 개발환경

2대의 서버를 구입하고, Celery를 이용한 머신러닝 학습 분산 실시를 하였다.



학습시간이 짧아졌다.

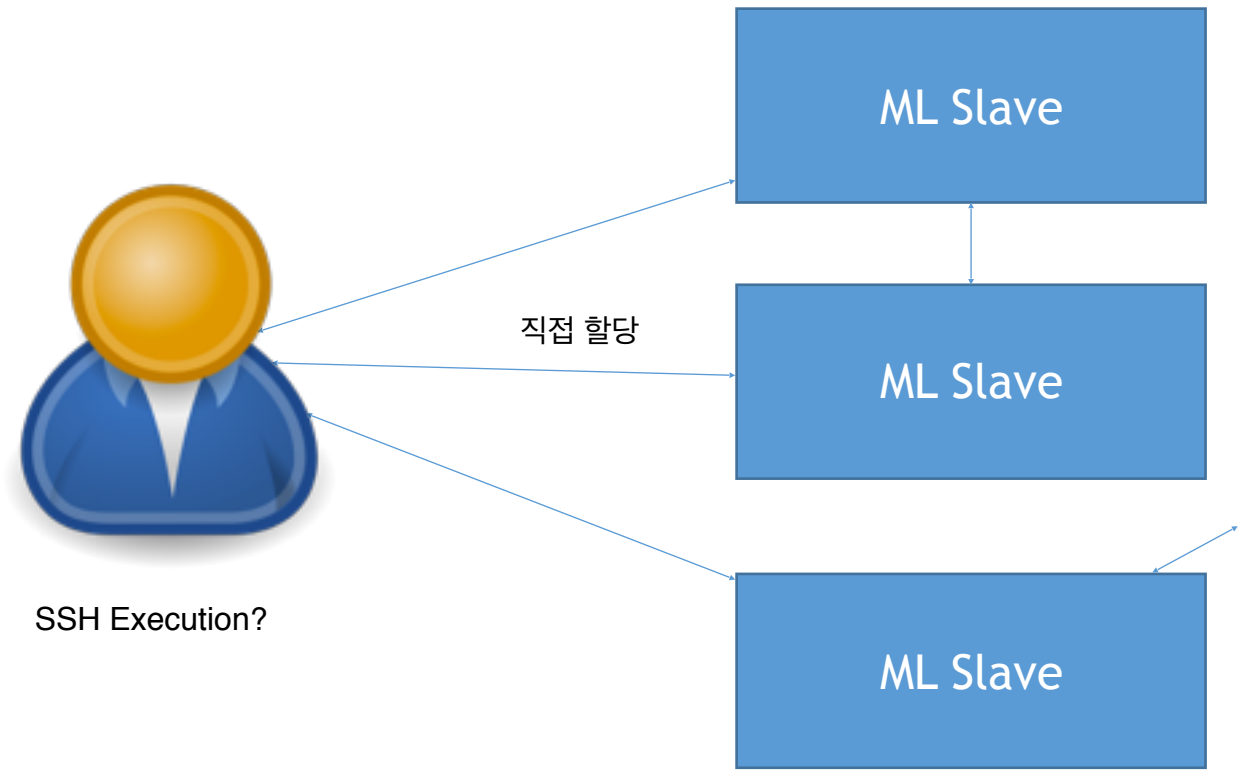
총 3대의 서버를 이용해 학습을 시키니 시간이 단축되었다. 당연하게도 …



PySpark + Mesos

First Trial의 가벼움

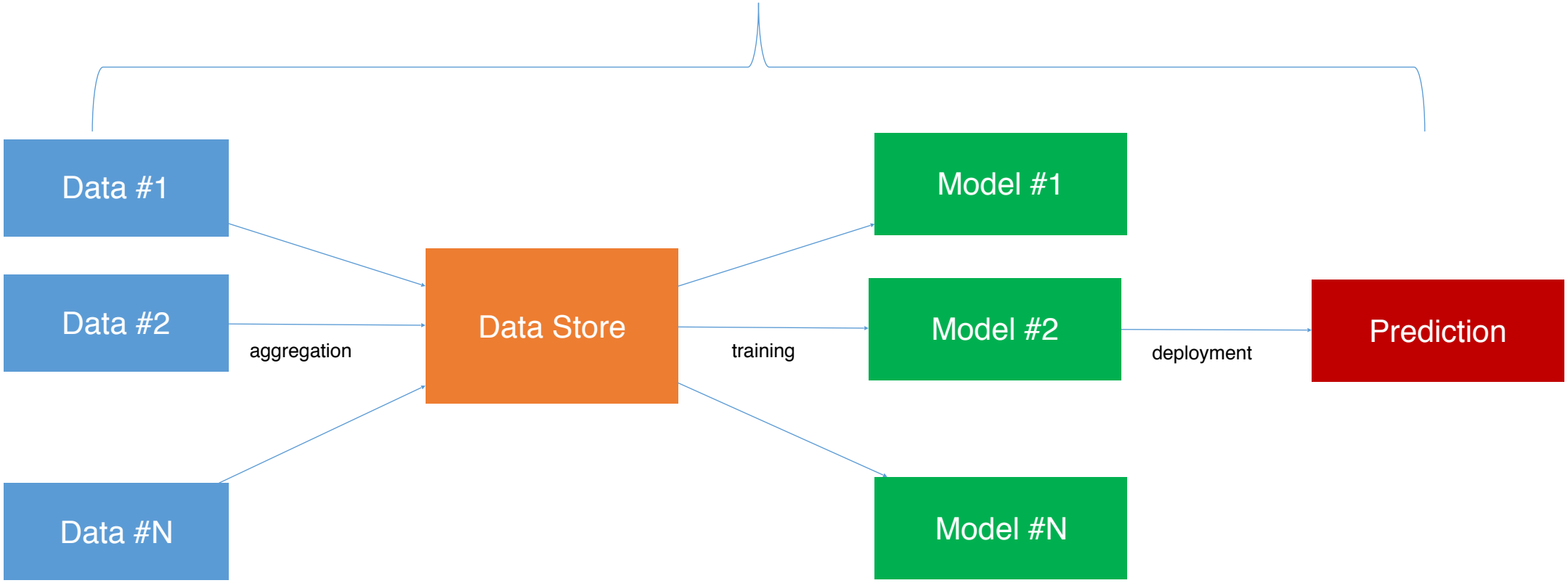
미친듯이 많은 모델들을 실시간으로 생성할 수 있는 뭔가 보다 세련된 방법이 필요하다!!!



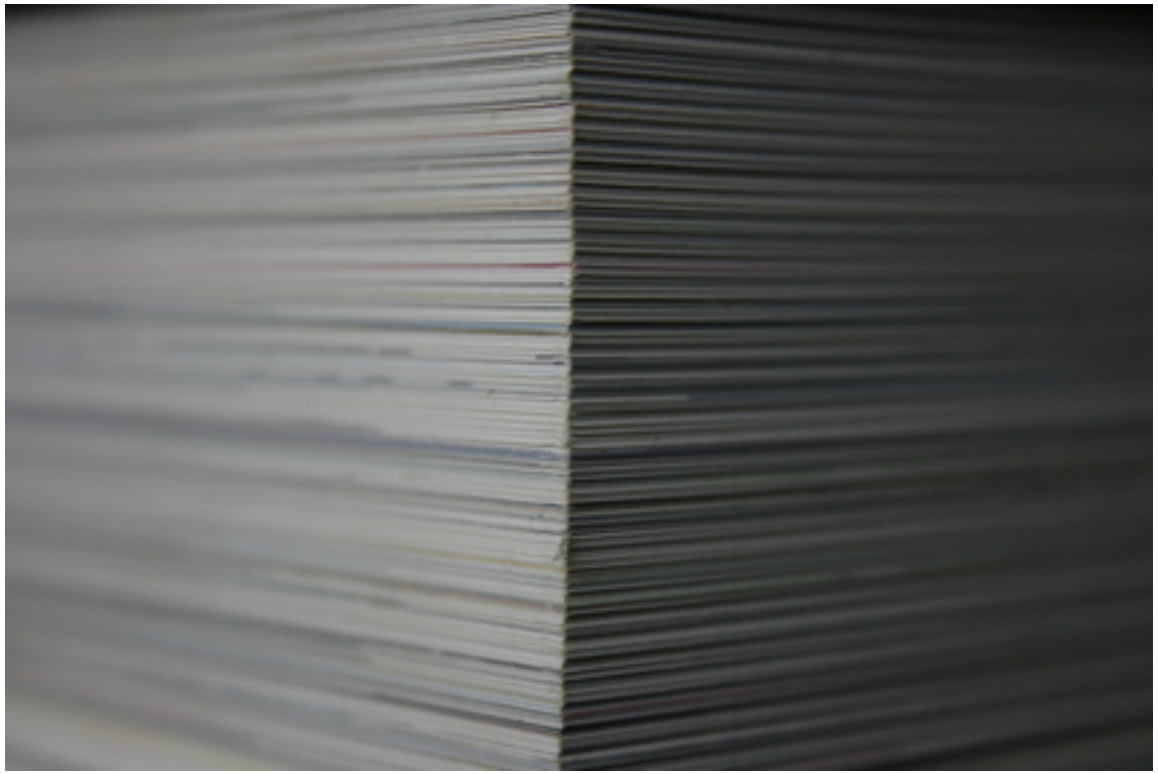
- 리소스관리의 어려움
- 노드 확장성의 문제
- 아키텍쳐의 문제
- 생각보다 까다로운 Celery
- 데이터 공유 문제

내가 필요한 것!

학습에서 최종모델 배포까지 1-2시간이내 완료!!!

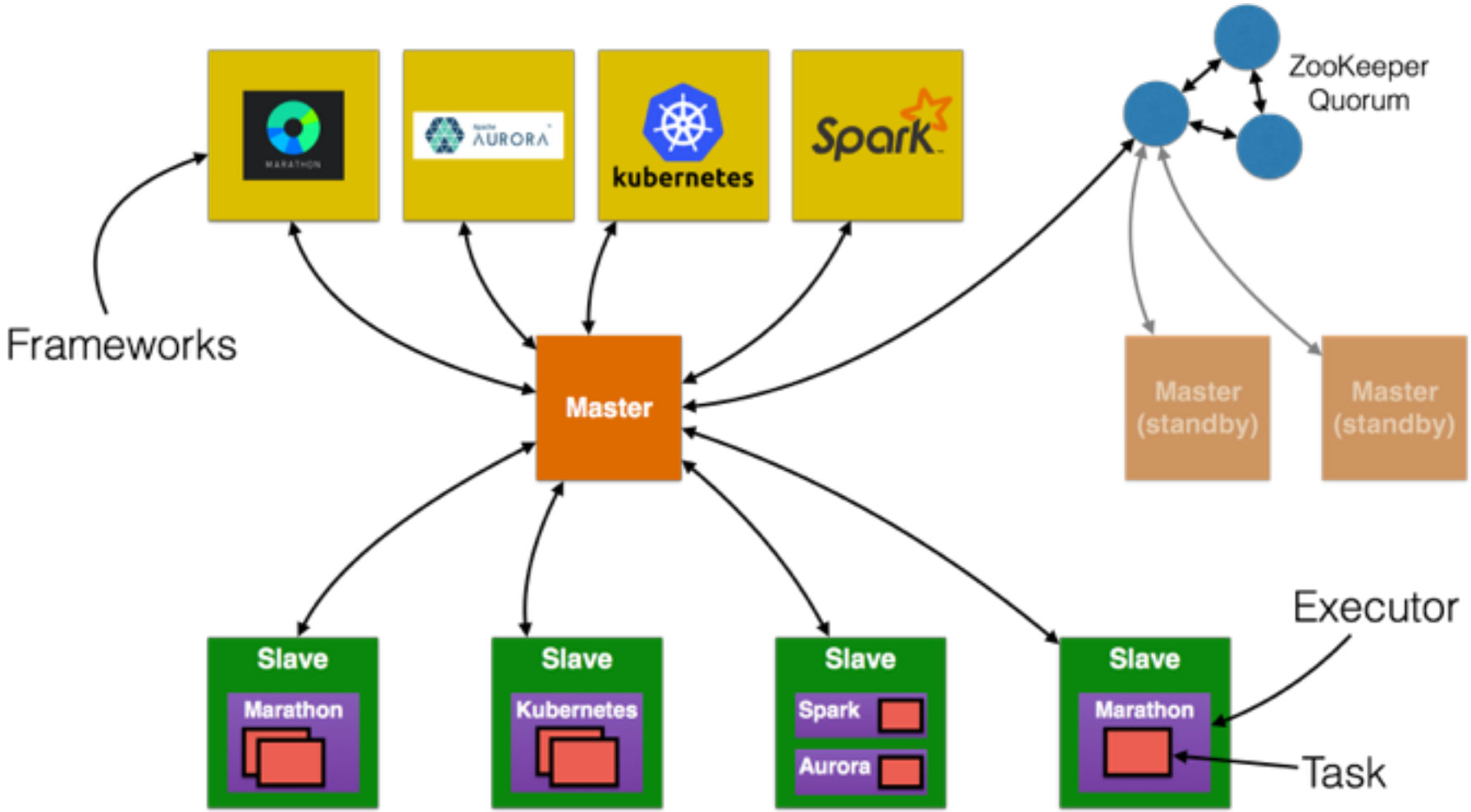


좀 더 구체적으로 표현하면



- 다양한 종류의 데이터 소스 사용
- 실시간 데이터 분석 및 학습
- 대용량 데이터 처리 기능
- 학습종료된 모델의 실시간 배치
- 머신러닝 노드 클러스터 확장
- 보다 효율적인 리소스 활용
- 머신러닝 클러스터 관리의 편리함
- Docker 연동
- 기타등등

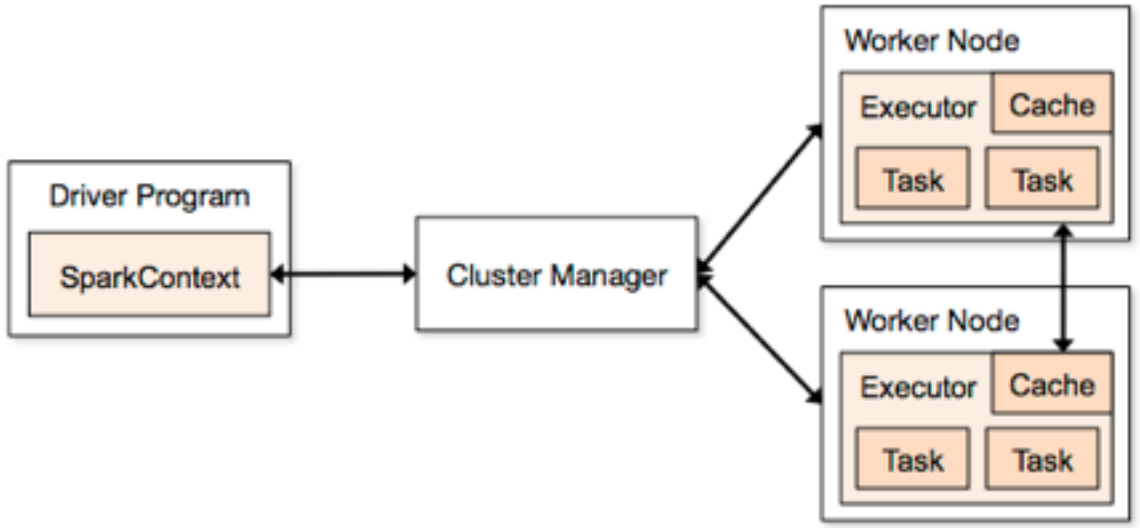
Apache Mesos + Spark



Apache Spark

Lightning fast cluster computing

Fast and scalable general data processing framework

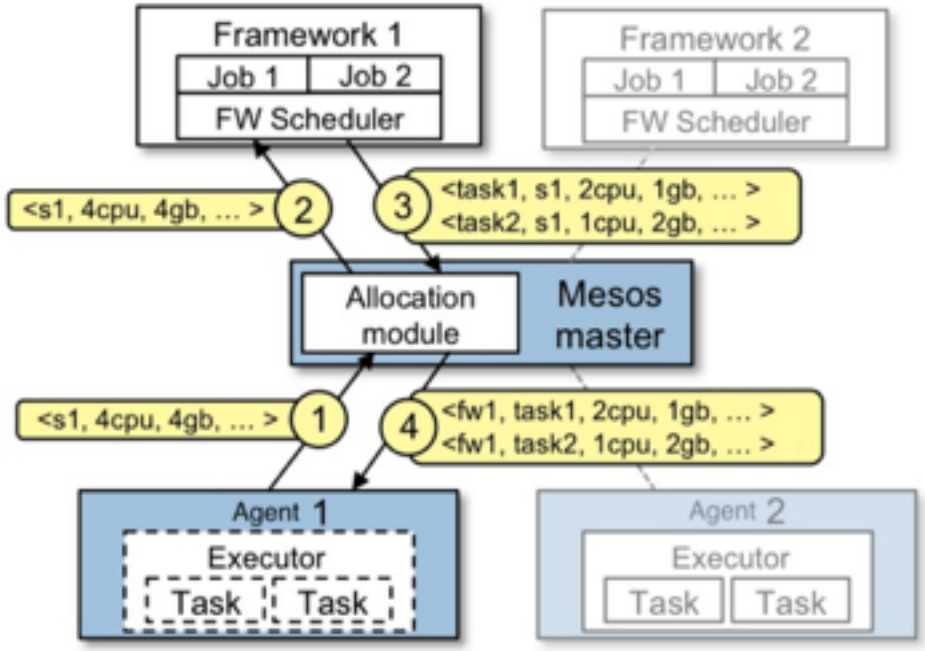


- Speed
  - Up to 100x faster than hadoop
- Ease of Use
  - Python, Scala, Java, R
- Generality
  - Combine SQL, streaming, analytics
- Runs Everywhere
  - Mesos, Hadoop, Standalone
  - HDFS, Cassandra, Hbase, S3

Apache Mesos

A Distributed System Kernel

abstracts CPU, memory, storage, and other compute resources away from machines, enabling fault-tolerant and elastic distributed systems



- Linear Scalability
- Two Level Scheduling
- High Availability
- APIs
- Pluggable Isolation
- Resource Management
- Supports Spark, Hadoop, Kafaa, ElasticSearch

Running Spark in Mesos

Client Mode

- 명령을 실행하는 머신에서 직접 실행
- spark-env.sh에 설정 필요
- MESOS_NATIVE_JAVA_LIBRARY
- SPARK_EXECUTOR_URI

Cluster Mode

- Mesos에 의해 태스크가 실행됨
- MesosClusterDispatcher 를 실행시켜야 함
- Spark-submit 을 이용해 실행
- 실행파일을 클러스터에서 액세스 가능하도록 설정

Mesos Run Modes

Fine-grained

- 메모리는 정적으로 할당받지만 CPU는 공유함
- 다른 태스크의 상황에 따라 성능이 변화됨

- conf.set("spark.mesos.coarse", "false")

Coarse-grained

- 메모리와 CPU를 모두 정적으로 할당받음
- 아래의 값들에 의해 할당됨
- spark.executor.memory

  spark.executor.cores
- Spark.cores.max/spark.executor.cores

- conf.set("spark.mesos.coarse", ”true")

Python Library Management

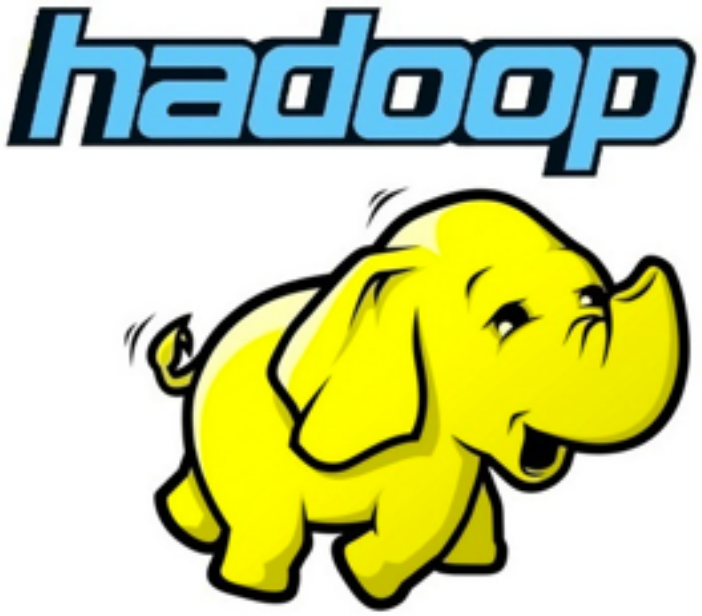
Egg or Zip

- Python 필요한 라이브러리를 egg or zip 파일로 만들어 pyFiles에 추가해 배포할 수 있음
- 실행하려고 할 때마다 매번 라이브러리 배포

Configuration Tool

- Ansible, puppet, chef와 같은 도구를 이용해 필요한 라이브러리를 설치할 수 있음
- 태스크 실행시 라이브러리는 포함하지 않음

Hadoop이 필요할까?

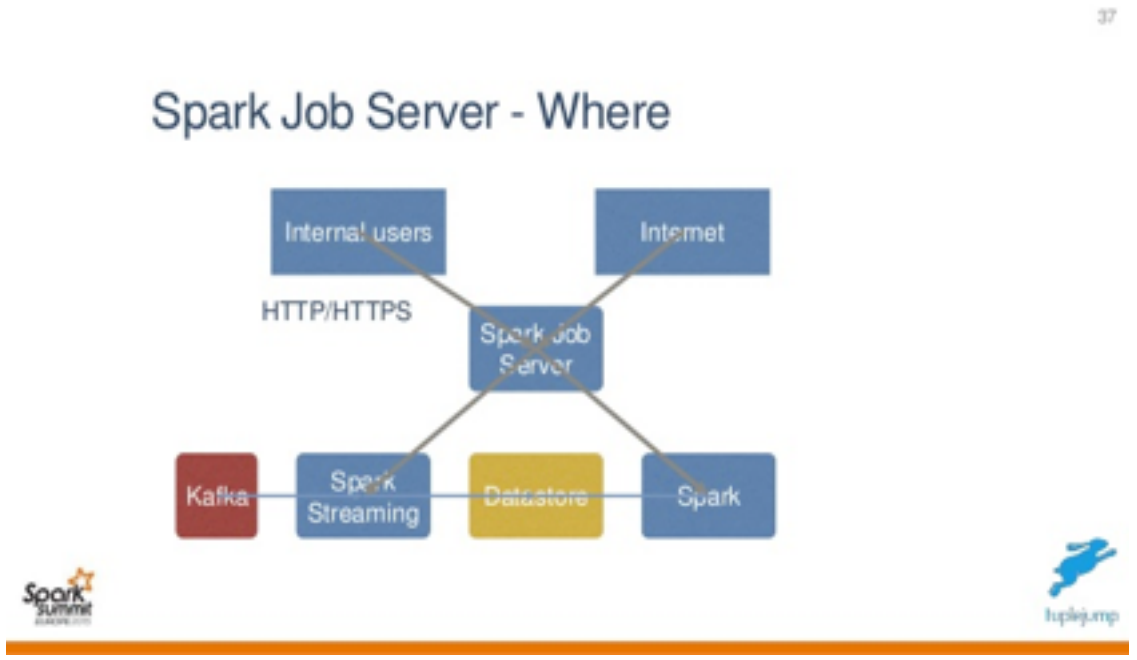


- 머신러닝만 생각하면 굳이 필요는 없었지만 …
  - Distributed File System
  - Disaster Recovery
  - Hadoop Ecosystem

spark-submit, 보다 세련된 뭔가가 필요해

```
./bin/spark-submit --class SimpleApp --
master mesos://dev-goose:7077 --deploy-
mode cluster simple-project_2.10-1.0.jar
```

REST API based Job Server



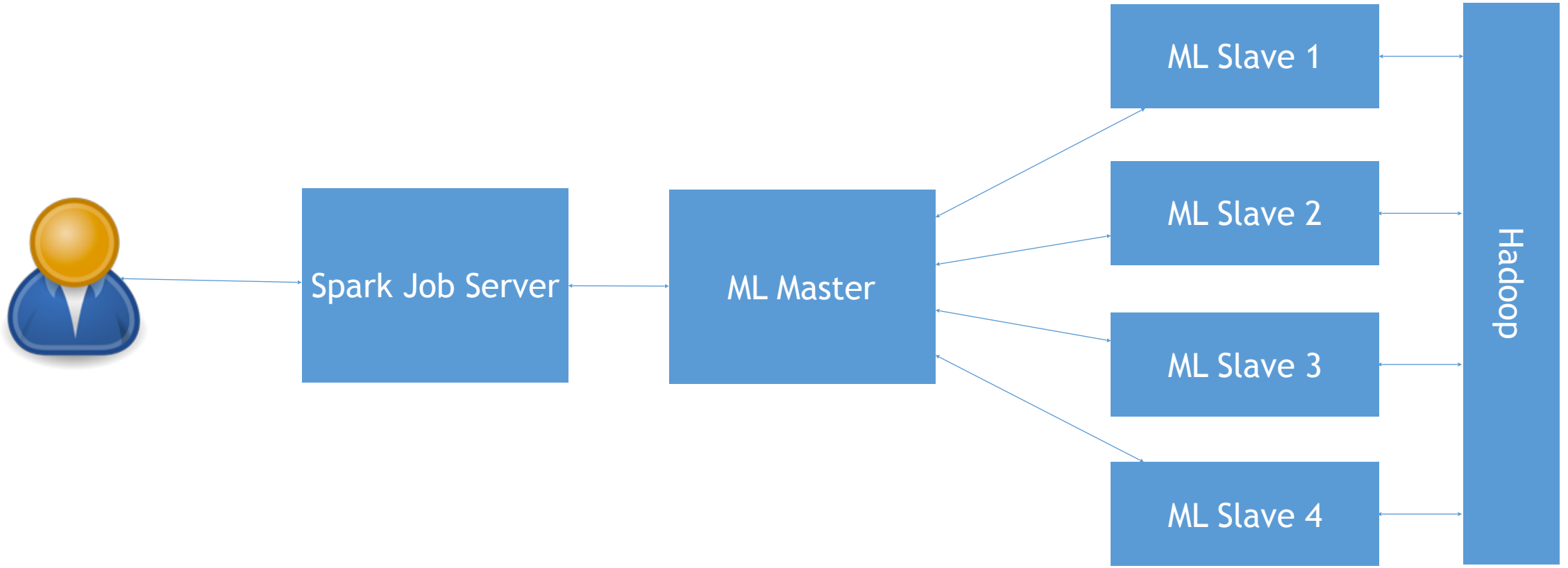
Spark Job Server

- ***"Spark as a Service"*: Simple REST interface (including HTTPS) for all aspects of job, context management**
- Support for Spark SQL, Hive, Streaming Contexts/jobs and custom job contexts!

  See Contexts.
- LDAP Auth support via Apache Shiro integration
- Works with Standalone Spark as well as Mesos and yarn-client
- **Named Objects (such as RDDs or DataFrames) to cache and retrieve RDDs or DataFrames by name, improving object sharing and reuse among jobs.**

https://github.com/spark-jobserver/spark-jobserver

현재의 머신러닝 클러스터



Before vs After

- 노드추가가 편해졌음
  - Mesos Slave, Hadoop 설정만으로 OK
- Streaming 처리가 가능해진 환경
  - Online Learning을 시도할 수 있게 되었음
- Dataframe이 RDD보다 성능이 더욱 좋음.
- Celery보다는 안정적인 환경
- 극적인 시간단축의 핵심은 머신러닝 알고리즘
- Tensorflow를 Spark에서
- 대규모의 데이터를 이용한 학습이 가능해짐

Performance?

5 Things To Remember

- Machine Learning Library
- Hyper-Parameter Optimization
- Machine Learning Models
- Dataframe
- Money

감사합니다.