Spark ML vs Sparkling Water: An Empirical Analysis of Distributed Machine Learning

Sidharth Singla • Varshanth R Rao

CS848 Course Project

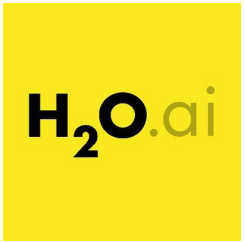
Agenda

1. Motivation
2. Distributed ML Frameworks
3. Overview of ML Algorithms
4. Experimental Setup
5. Research Questions (RQ)
6. RQ Experiments
7. Conclusion
8. Future Work

Motivation

(A)

Big Data Distributed Frameworks



(B)

ML Libraries

(A+B)

Distributed ML Frameworks

Distributed Framework	Spark	Spark	Spark	Spark
ML/DL Support	Strong ML/Basic DL	Strong ML/Strong DL	Strong DL (Keras)	Strong ML/Basic DL
Data Abstraction	Spark Dataframe	H2O Frame (In mem)	Spark DF/RDD	RDD
Public Release	Stable	Stable	Stable	Alpha
Dev Language	Multiple (Scala Base)	Multiple (Scala Base)	Python	Scala

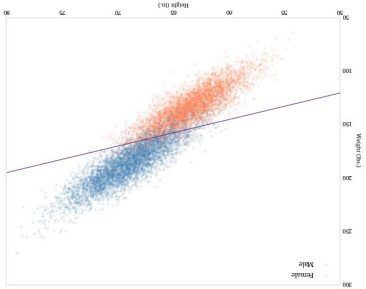
Focus:

(Py) Spark ML vs (Py) Sparkling Water

ML Algorithms

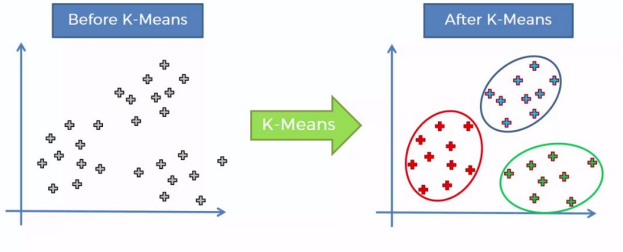
Logistic Regression

- Solves binary classification problem by assigning probabilities to predictions
- Parameter Learning
- Parameter updates through gradient computation



K-Means Clustering

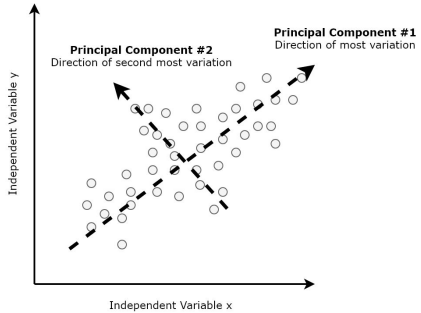
- Unsupervised non parametric learning
- Solves data grouping problem
- Relies on feature similarity of neighbors



ML Algorithms

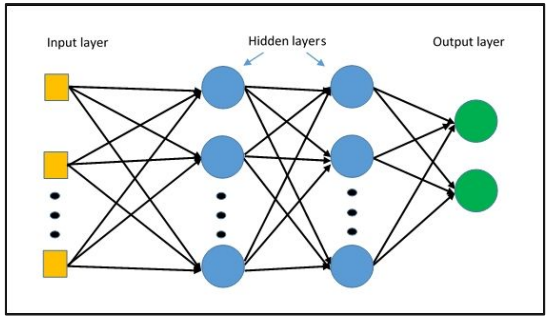
Principal Component Analysis

- Dimensionality Reduction Technique
- Singular Value Decomposition of Covariance Matrix
- Used to “uplift” the “curse of dimensionality”



Multi-Layer Perceptron (MLP)

- Parametric Universal Function Approximator
- Applies non linearity & parameter updates performed through gradient computation



ML Algorithms

	LR	KMC	PCA	MLP
Time Complexity (Per Iteration)	O (n * d)	O (n * d * k)	O (n * d^2 + d^3)	O (n * (d * h_1 + h_1 * h_2 + .. h_m * c))
Space Complexity	O (d)	O ((n + k) * d)	O (n * d + d^2)	O (d * h_1 + h_1 * h_2 + .. h_m * c)

n = Number of samples in dataset

d = Number of features representing each sample

k = Number of clusters

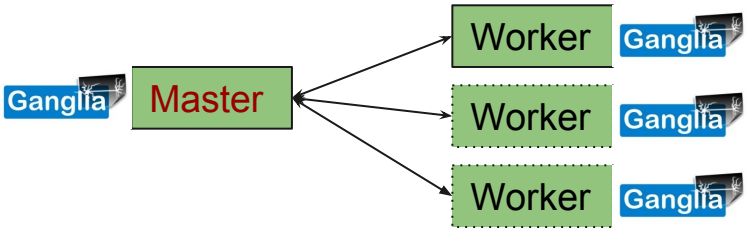
h_i = Number of hidden nodes at layer i

c = Number of output classes

Experimental Setup

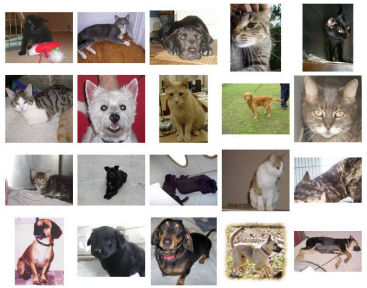
Cluster Config

- Spark Standalone Cluster Mode
- 1 Master Node & [1,2,3] Worker Nodes
- 12 Core CPU, 16GB RAM Per Node
- Ganglia monitoring setup on each node with metrics sinks as the driver node



Dataset: Cats vs Dogs

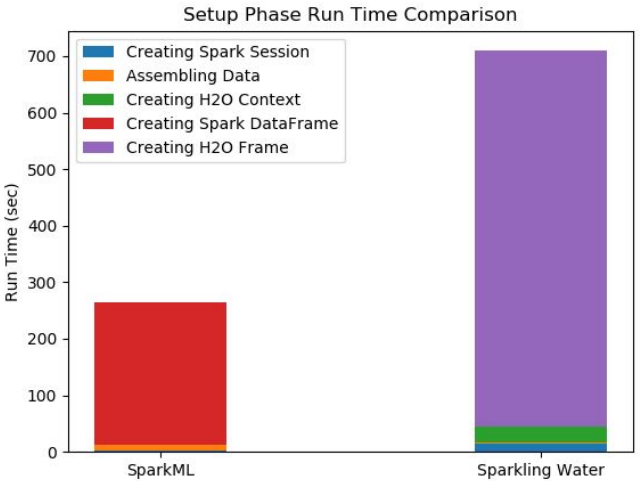
- ~130,000 images of cats and dogs
- 2048-d ResNet 50 features as data
- Total Size = 3GB
- Training:Testing = 5:1

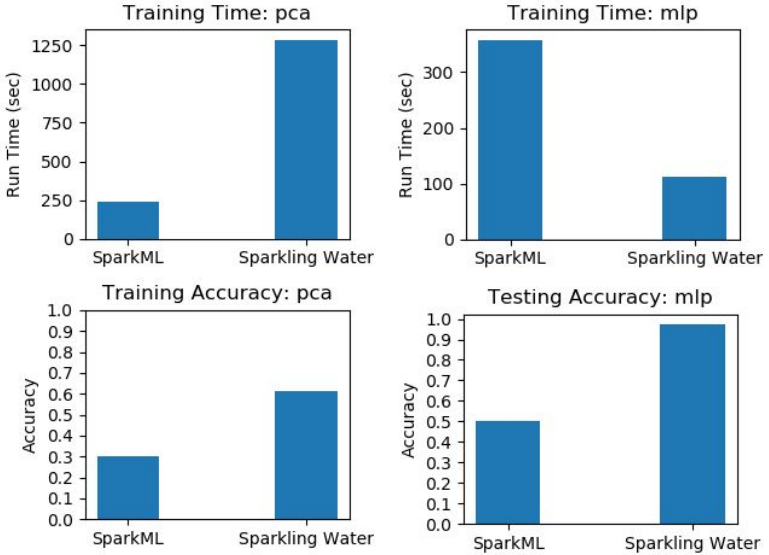


Research Questions

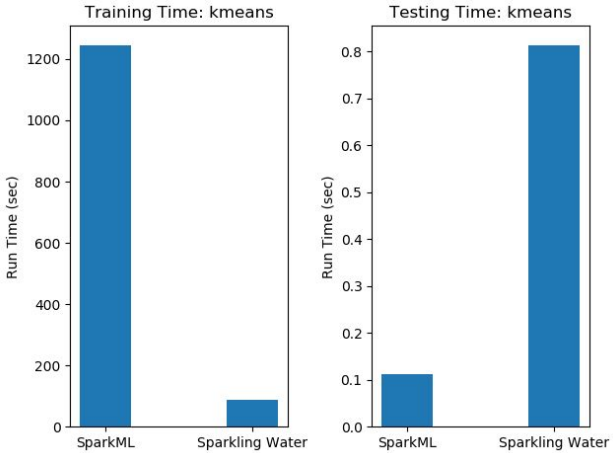
1) Given a dataset of fixed size and fixed cluster:
    a) How does each library compare in terms of runtime & accuracy?
    b) What are the characteristics of the network usage of the libraries?

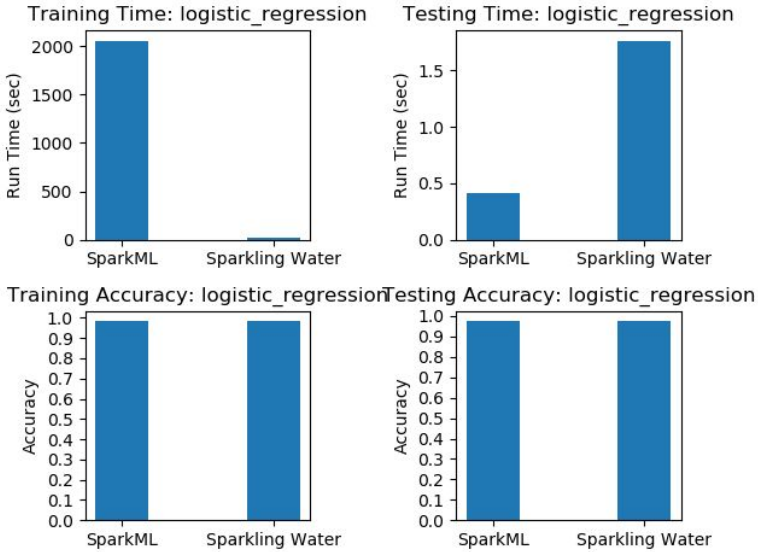
RQ1a: Dataset Size & Cluster Fixed





RQ1a: Dataset Size & Cluster Fixed





RQ1b: Dataset Size & Cluster Fixed - Network Usage

PCA

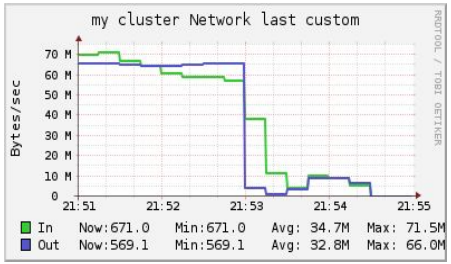
MLP

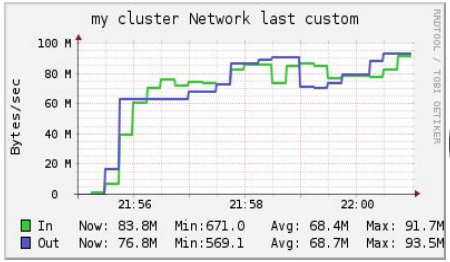
K-Means

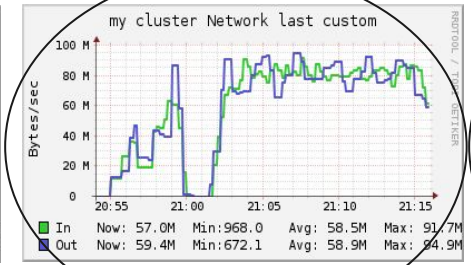
Logistic Regression

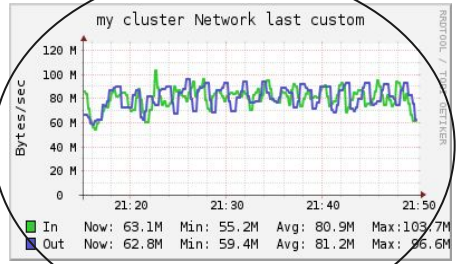
Spark ML

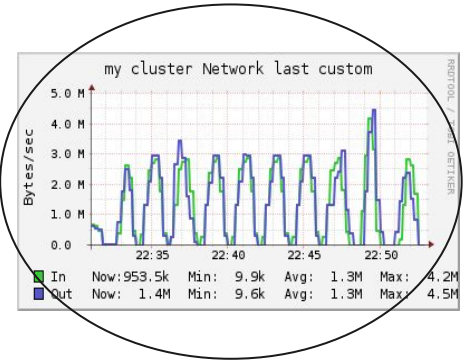
Sparkling Water

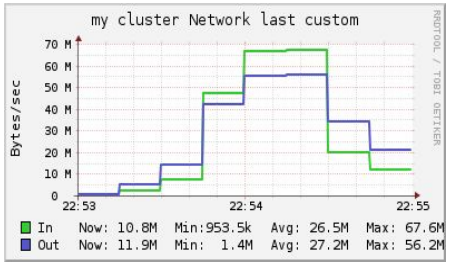


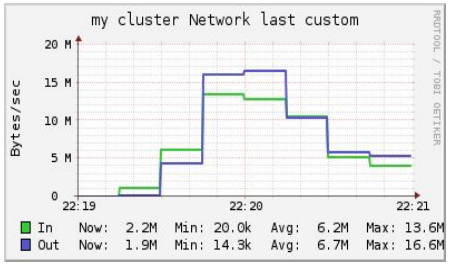


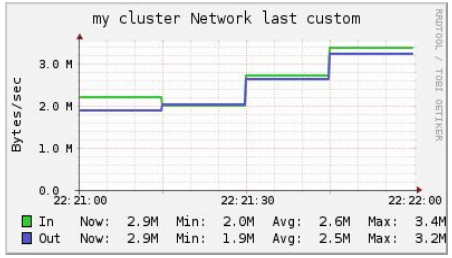








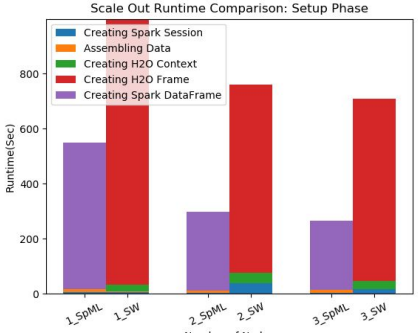


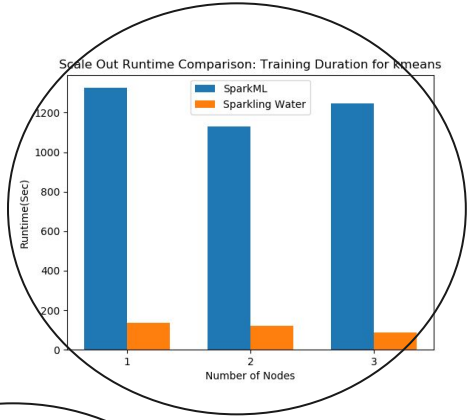


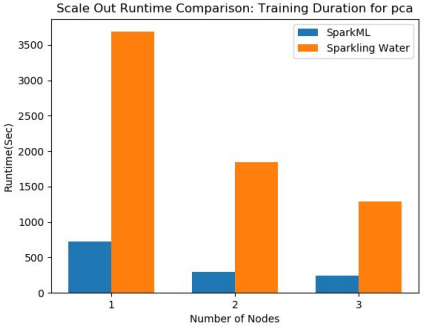
Research Questions

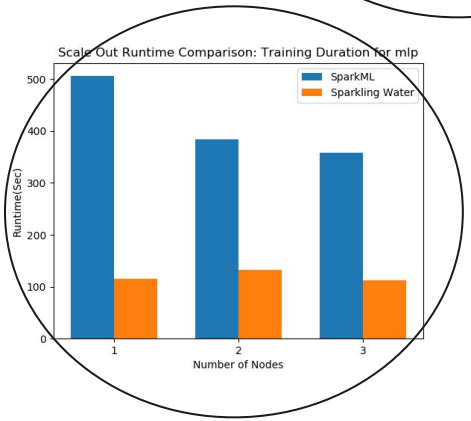
2) **Given a dataset of fixed size:**
   a) Does scale out reduce the runtime linearly as expected?
   b) Does the model accuracy vary with scale out?

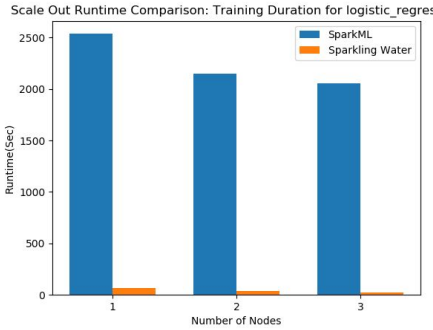
RQ2a: Scale Out Effect on Runtime



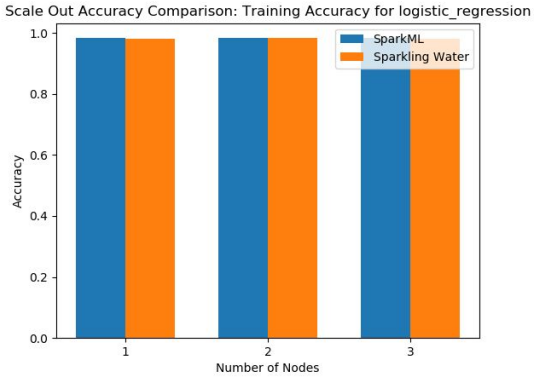


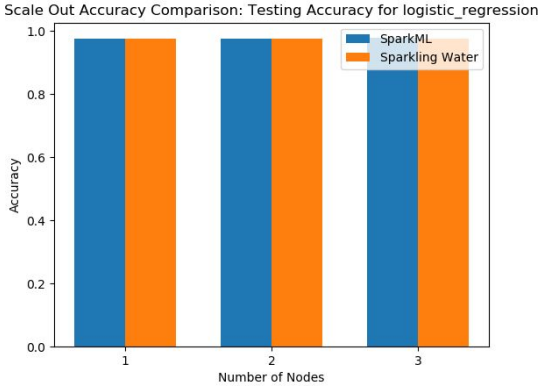


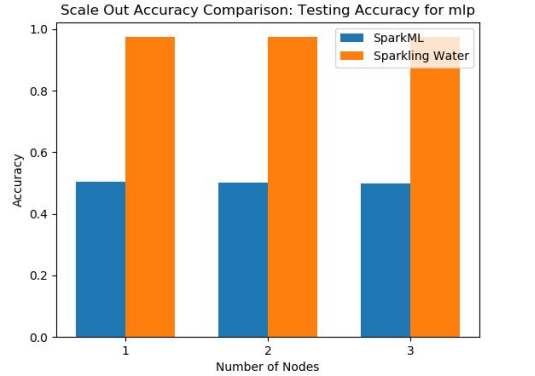


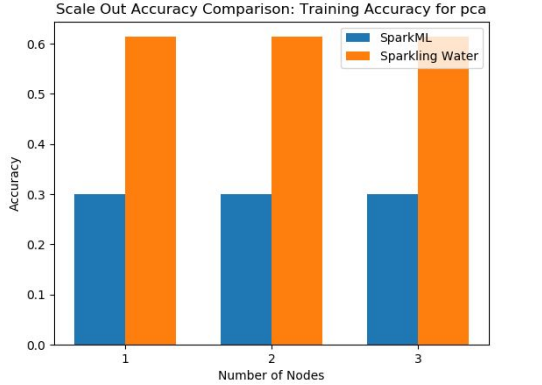


RQ2b: Scale Out Effect on Accuracy





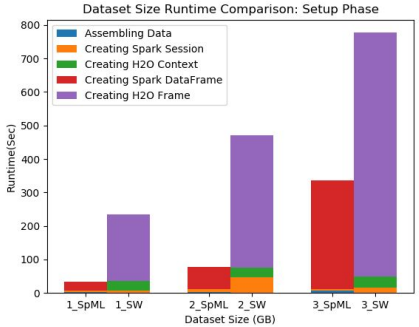


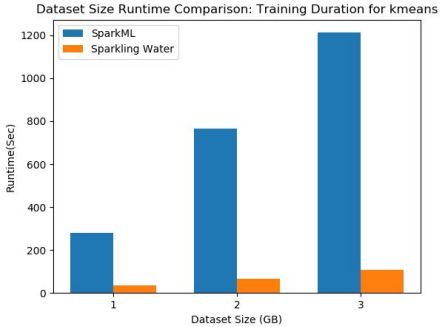


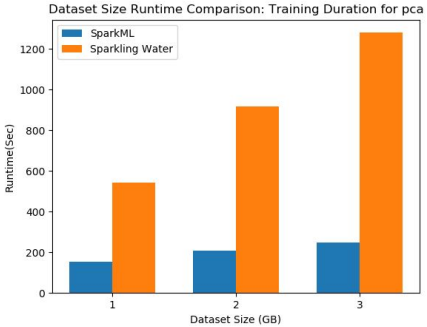
Research Questions

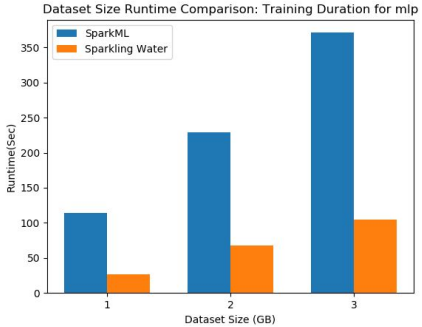
3) **Given a cluster with fixed number of worker nodes:**
How does each framework behave when the dataset size is increased linearly?

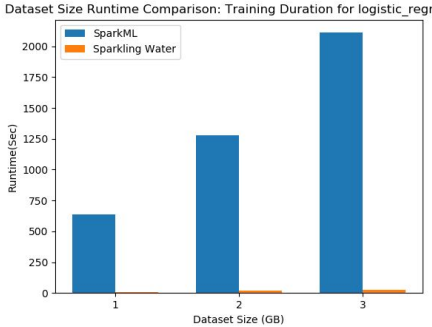
RQ3: 3-Node Setup - Effect of Dataset Size (Runtime)











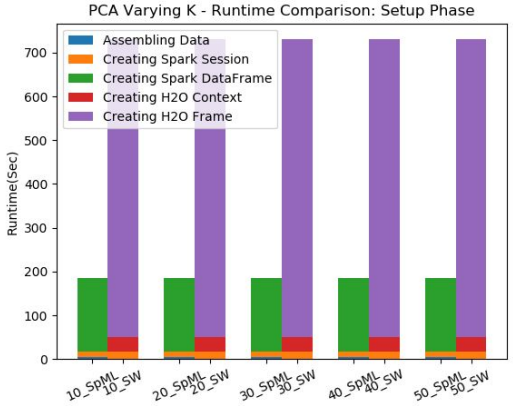
RQ3: 3-Node Setup - Effect of Dataset Size (Memory)

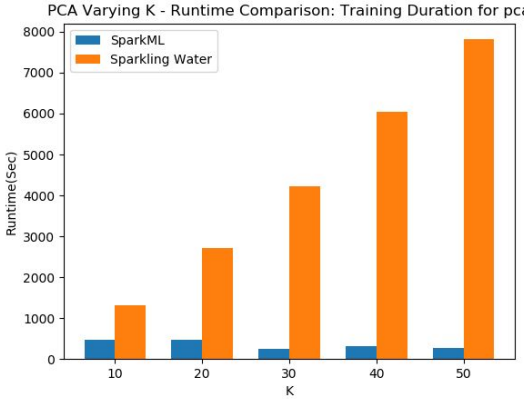
Algorithm/DS Size	1G-SpML	1G-SW	2G-SpML	2G-SW	3G-SpML	3G-SW
Logistic Regression	37.3G	15.5G	41.9G	23.8G	48.5G	29.6G
K-Means Clustering	32.6G	14.7G	41.8G	22.4G	44.9G	29.6G
PCA	38.2G	22.1G	42.5G	26.9G	48.7G	30.8G
MLP	38.2G	22.3G	43.4G	27.4G	49.5G	31.8G

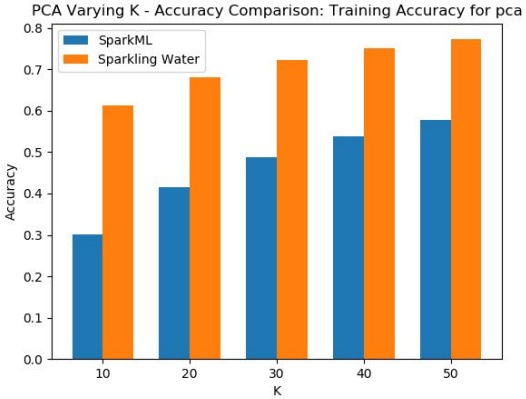
Research Questions

4) **Case Study:**
Deep Dive into PCA Observations

RQ4: 3 Node Setup - Vary “K” in PCA







Numerical Processing Library	Breeze	Matrix Toolkits Java

Answers: Research Questions

RQ1: Initial Looks

- Sparkling Water Accuracy ≥ Spark ML Accuracy (Except PCA)
- Sparkling Water Training Time < Spark ML Training Time (Except PCA)
- Sparkling Water Network Usage << Spark ML Network Usage

RQ2: Scale Out Effect

- Gain in runtime performance but algorithm dependent.
- Scale Out Does Not Affect Training/Testing Accuracy

Answers: Research Questions

RQ3: Dataset Size Increase Effect:

- Setup phase takes a hit due to data distribution & in-memory store/swap
- Training time increases linearly as dataset size increases
- Memory footprint for algos independent of space complexity (in-memory data)

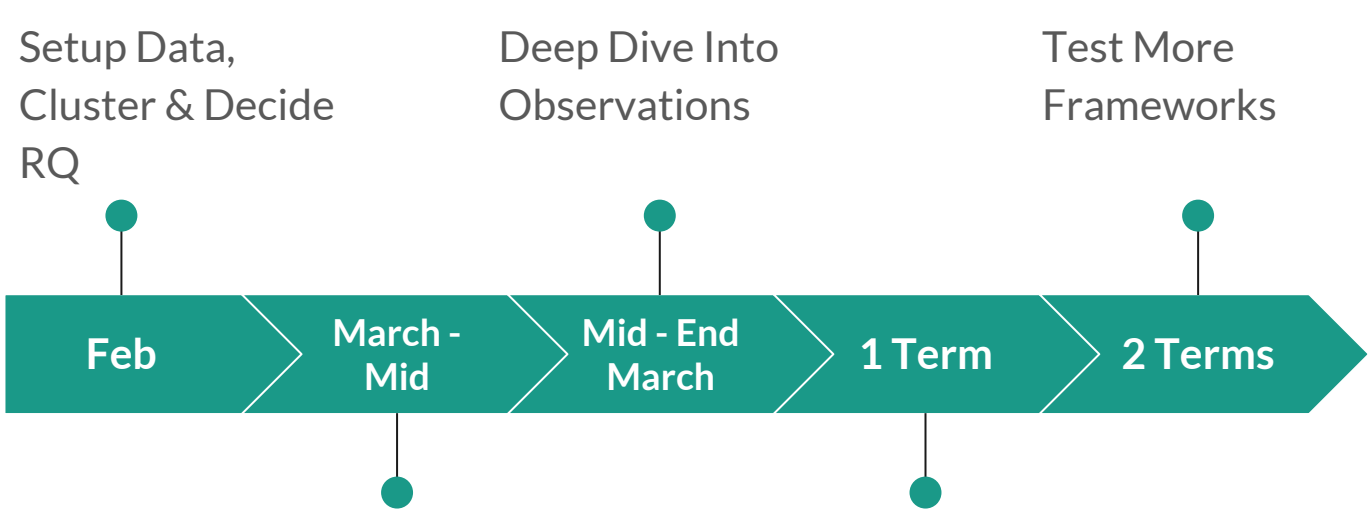
RQ4: PCA - Varying “K” Effect:

- Sparkling Water takes substantially longer for training
- Sparkling Water yields larger (but diminishing) returns on accuracy
- Difference due to usage of different libraries

Conclusions:

	Spark ML Pipelines	Spark + H2O Sparkling Water
Setup Runtime:	**Less** due to lazy load of Spark Dataframe	**More** due to in-memory load of **Compressed** H2O Frame
Training Runtime:	**More** for algos except **PCA (large K). More** affected by dataset size increase.	**Less** for algos except **PCA (large K). Less** affected by dataset size increase.
Memory Footprint:	**More** due to decompression into Java objects in memory	**Less** due to JIT decompression in CPU registers from memory
Network Usage:	**More** due to lazy load paradigm of Spark Dataframe	**Less** due to the exploitation of data locality by algorithms
Model Performance:	Roughly **Equal** for LR, **Less** for MLP & PCA	Roughly **Equal** for LR, **More** for MLP & PCA

Future Work


Setup Data, Cluster & Decide RQ
Deep Dive Into Observations
Test More Frameworks
Feb
March - Mid
Mid - End March
1 Term
2 Terms
Frame & Conduct Experiments
Scale Out With Bigger Data, More Algos

Thank You!

Q & A

Thinking Out Loud ….

“Sparkling Water has more knobs that can be tuned for algorithms”

“Sparkling Water’s default hyperparameters per algorithm yields better performance over Spark ML”

“Log messages in Sparkling Water are more descriptive than Spark ML”

“User friendly guides from Sparkling Water”

“Ganglia lacks command line support for metrics collection”

“H2O.ai came out with H2O Deep Water (Distributed GPU Based DL) when we were doing our project!”