Big data management and predictive analytics for customer transactions & operations using Apache Spark and AWS

Large Scale Distributed Machine Learning

BDA761 Big Data Mgmt in a Supercomputing environment

MUDIT UPPAL

Problem with Big Data(s)

- Machine learning practices at scale for PB/TB data
- A framework which provides and computes models using virtual nodes with processors and memory getting cheaper every year
- Using GPU + multi-threading + make use of multiple cores
- Goal: Thinking in ‘big data’; create a tool which can be used in any operations/Sales/customer analysis
- Parallelizing — DATA and MODEL

Case study

- *Rossmann*: operates over 3000 drug stores in 7 european countries
- Store sales are influenced by many factors, including promotions, competition, school and state holidays, seasonality, and locality.
- DECISION TASKS:
  - Forecast Sales for upto 6 weeks using stores data, customers, promotion data et cetera
  - Predicting Sales of ~1000 stores daily

Data Source: Rossmann, Walmart via Kagle

Id	Store	DayOfWeek	Date	Open	Promo	StateHoliday	SchoolHoliday
1	1	4	2015-09-17	1	1	0	0
2	3	4	2015-09-17	1	1	0	
3	7	4	2015-09-17	1	1	0	
4	8	4	2015-09-17	1	1	0	
5	9	4	2015-09-17	1	1	0	
6	10	4	2015-09-17	1	1	0	
7	11	4	2015-09-17	1	1	0	
8	12	4	2015-09-17	1	1	0	
9	13	4	2015-09-17	1	1	0	
10	14	4	2015-09-17	1	1	0	
11	15	4	2015-09-17	1	1	0	
12	16	4	2015-09-17	1	1	0	
13	19	4	2015-09-17	1	1	0	
14	20	4	2015-09-17	1	1	0	
15	21	4	2015-09-17	1	1	0	
16	22	4	2015-09-17	1	1	0	
17	23	4	2015-09-17	1	1	0	

Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear	PromoInterval
1	c	a	1270	9	2008	0			
2	a	a	570	11	2007	1	13	2010	Jan,Apr,Jul,Oct
3	a	a	14130	12	2006	1	14	2011	Jan,Apr,Jul,Oct
4	c	c	620	9	2009	0			
5	a	a	29910	4	2015	0			
6	a	a	310	12	2013	0			
7	a	c	24000	4	2013	0			
8	a	a	7520	10	2014	0			
9	a	c	2030	8	2000	0			
10	a	a	3160	9	2009	0			
11	a	c	960	11	2011	1	1	2012	Jan,Apr,Jul,Oct
12	a	c	1070			1	13	2010	Jan,Apr,Jul,Oct
13	d	a	310			1	45	2009	Feb,May,Aug,Nov
14	a	a	1300	3	2014	1	40	2011	Jan,Apr,Jul,Oct
15	d	c	4110	3	2010	1	14	2011	Jan,Apr,Jul,Oct
16	a	c	3270			0			
17	a	a	50	12	2005	1	26	2010	Jan,Apr,Jul,Oct
18	d	c	13840	6	2010	1	14	2012	Jan,Apr,Jul,Oct
19	a	c	3240			1	22	2011	Mar,Jun,Sept,Dec
20	d	a	2340	5	2009	1	40	2014	Jan,Apr,Jul,Oct
21	c	c	550	10	1999	1	45	2009	Jan,Apr,Jul,Oct
22	a	a	1040			1	22	2012	Jan,Apr,Jul,Oct
23	d	a	4060	8	2005	0			
24	a	c	4590	3	2000	1	40	2011	Jan,Apr,Jul,Oct
25	c	a	430	4	2003	0			
26	d	a	2300			0			
27	a	a	60	1	2005	1	5	2011	Jan,Apr,Jul,Oct
28	a	a	1200	10	2014	1	6	2015	Mar,Jun,Sept,Dec
29	d	c	2170			0			
30	a	a	40	2	2014	1	10	2014	Mar,Jun,Sept,Dec
31	d	c	9800	7	2012	0			
32	a	a	2910			1	45	2009	Feb,May,Aug,Nov

TripType	VisitNumber	Weekday	Upc	ScanCount	DepartmentDescription	FinelineNumber
999	5	Friday	68113152929	-1	FINANCIAL SERVICES	1000
30	7	Friday	60538815980	1	SHOES	8931
			7410811099	1	PERSONAL CARE	4504
			2238403510	2	PAINT AND ACCESSORIES	3565
			2006613744	2	PAINT AND ACCESSORIES	1017
			2006618783	2	PAINT AND ACCESSORIES	1017
			2006613743	1	PAINT AND ACCESSORIES	1017
			7004802737	1	PAINT AND ACCESSORIES	2802
			2238495318	1	PAINT AND ACCESSORIES	4501
			2238400200	-1	PAINT AND ACCESSORIES	3565
			5200010239	1	DSD GROCERY	4606
			38679300501	2	PAINT AND ACCESSORIES	3504
			22006000000	1	MEAT - FRESH & FROZEN	6009
			2236760452	1	PAINT AND ACCESSORIES	7
			38679300501	-1	PAINT AND ACCESSORIES	3504
			2238400200	2	PAINT AND ACCESSORIES	3565
			3019294203	1	PAINT AND ACCESSORIES	2801
			72450408840	1	PAINT AND ACCESSORIES	1028
			25541500000	2	DAIRY	1305
			2310010776	1	PETS AND SUPPLIES	3300
			72450403700	2	PAINT AND ACCESSORIES	1018
			7874204967	1	HOUSEHOLD CHEMICALS/SUPP	707
			5114139038	1	PAINT AND ACCESSORIES	4415
			5114197561	1	PAINT AND ACCESSORIES	4415
			3270011053	3	PETS AND SUPPLIES	1001
				1	NULL	

~2.8 million rows data points

Label: Sales

Features: Store, Sales, customers, Open, date, stateholiday, schoolHoliday, storetype, Assortment, competitionDistance, CompetitionSinceMonth, sinceYear, Promo, Promo2, PromoeInterval, Promo2SinceMonth, PromoSince year, DayOfWeek, etc…

Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear	PromoInterval
1	c	a	1270	9	2008	0			
2	a	a	570	11	2007	1	13	2010	Jan,Apr,Jul,Oct
3	a	a	14130	12	2006	1	14	2011	Jan,Apr,Jul,Oct

Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
1	5	2015-07-31	5263	555	1	1	0	1
2	5	2015-07-31	6064	625	1	1	0	1
3	5	2015-07-31	8314	821	1	1	0	1

Goal

- Aim is to create a Predictive Analytics Framework
  - distributed machine learning for any size dataset
- 3 Demos in this presentation
  - Apache Spark
  - Exploratory DA with R
  - xgBoost(python) ML

Data Source: Rossmann, Walmart via Kagle

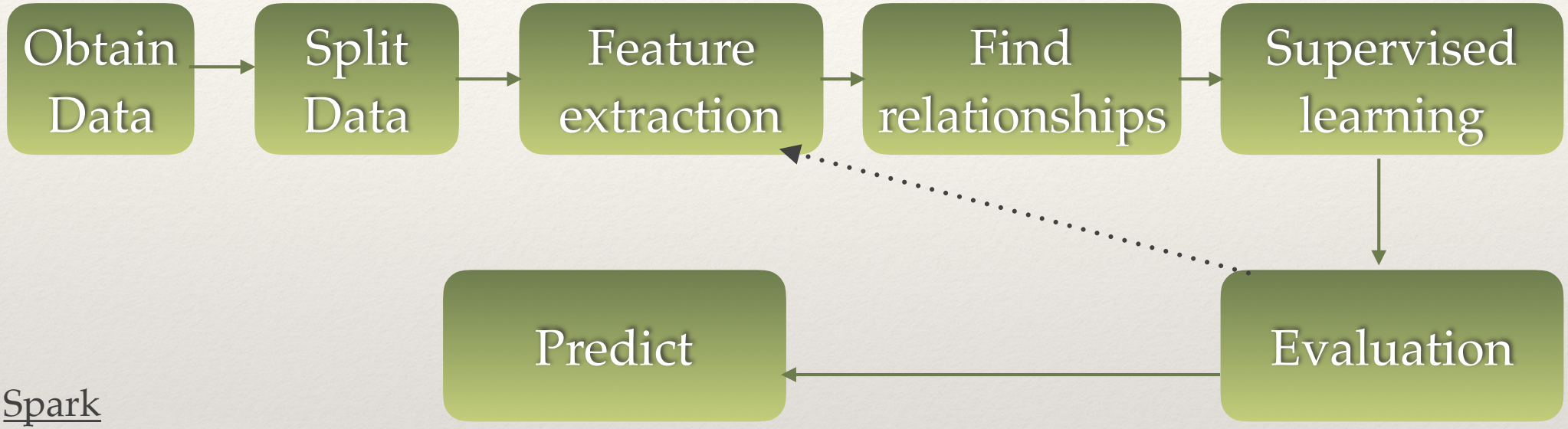
Main Demos

- Exploratory data analysis
- Apache Spark(sql spark context) - Distributed ML across multiple machines/nodes
  - Linear Regression analysis
  - Gradient descent
- Ensemble and boosting algorithms (XGBoost)

Tools

- R + python for exploratory analysis
- Apache Spark for implementation
- Hadoop
- Spark MLlib
- AWS cloud(m4 large instances)
- Ganglia(distributed monitoring system ti work with clusters)
- pyspark

Data Pipeline



Spark

Parse Initial dataset

Use 'LabeledPoint' class

Visualize features

Shift labels(starting from zero)

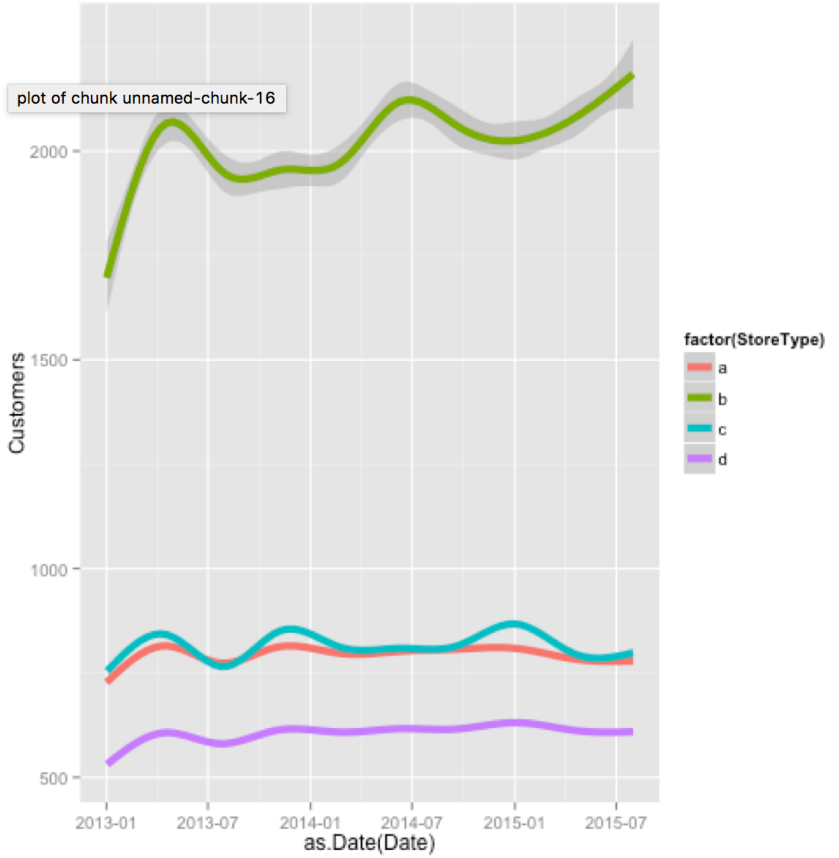
Use weights to make predictions

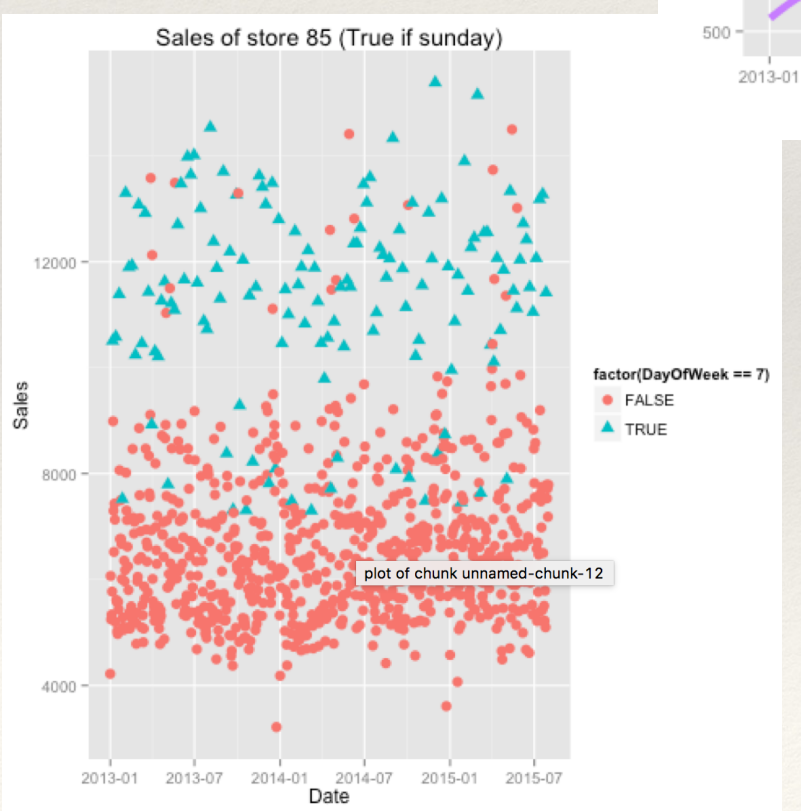
Train (via gradient descent) and evaluate a linear regression

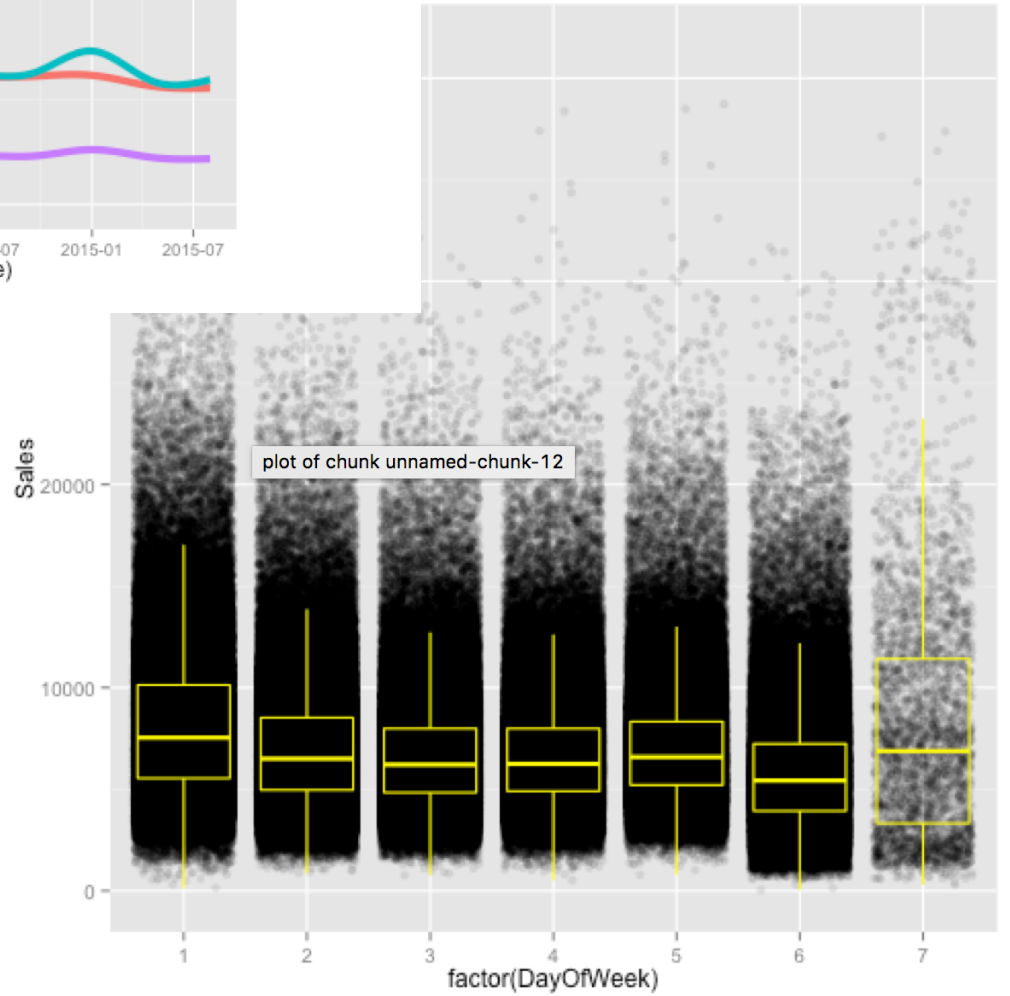
Create and Evaluate Baseline model

Demo 1: exploratory analysis

- R+python
- **Data and source code can be downloaded from:** http://muppal.com







ApacheSpark <—–> xgboost

- **Spark** excels at distributing operations across a cluster while abstracting away many of the underlying implementation details.
- Thinking in terms of RDD(transformations and actions)
- Still under development

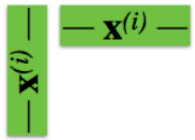
XGBoost:

- start off with a rough prediction and then building a series of decision trees; with each trying to correct the prediction error of the one before
- - Large-scale and Distributed Gradient Boosting (GBDT, GBRT or GBM) Library, on single node, hadoop yarn etc
- XGBoost can also be distributed and scale to Terascale data
- You can define threads with “nthreads =..”
- https://github.com/dmlc/xgboost

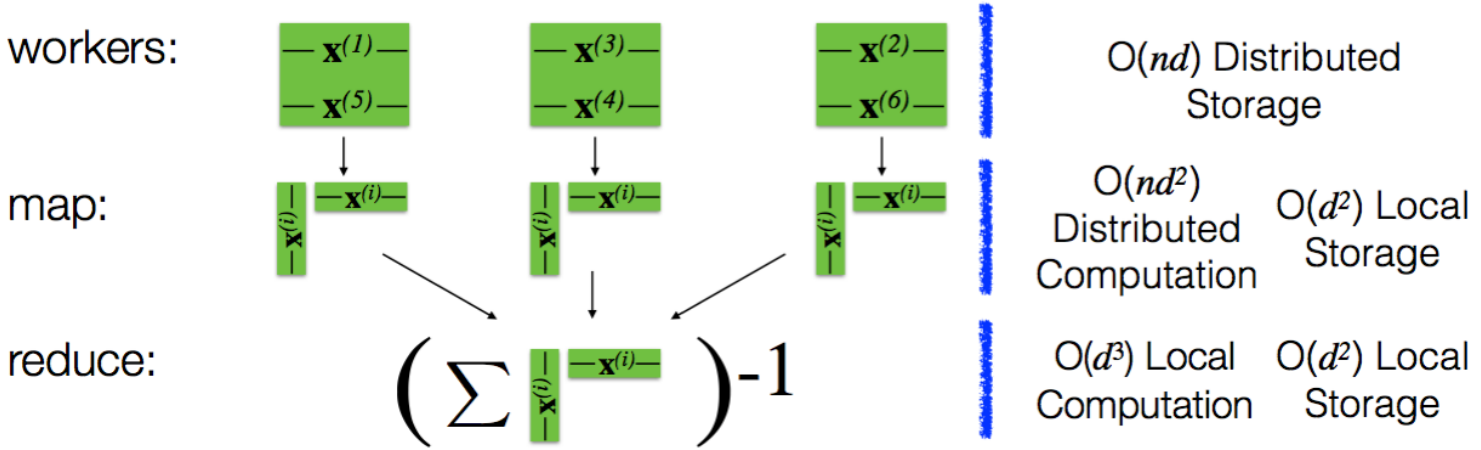
$$\begin{bmatrix} 9 & 3 & 5 \\ 4 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & -5 \\ 2 & 3 \end{bmatrix} = \begin{bmatrix} 28 & 18 \\ 11 & 9 \end{bmatrix}$$

$$\begin{bmatrix} 9 & 18 \\ 4 & 8 \end{bmatrix} + \begin{bmatrix} 9 & -15 \\ 3 & -5 \end{bmatrix} + \begin{bmatrix} 10 & 15 \\ 4 & 6 \end{bmatrix}$$

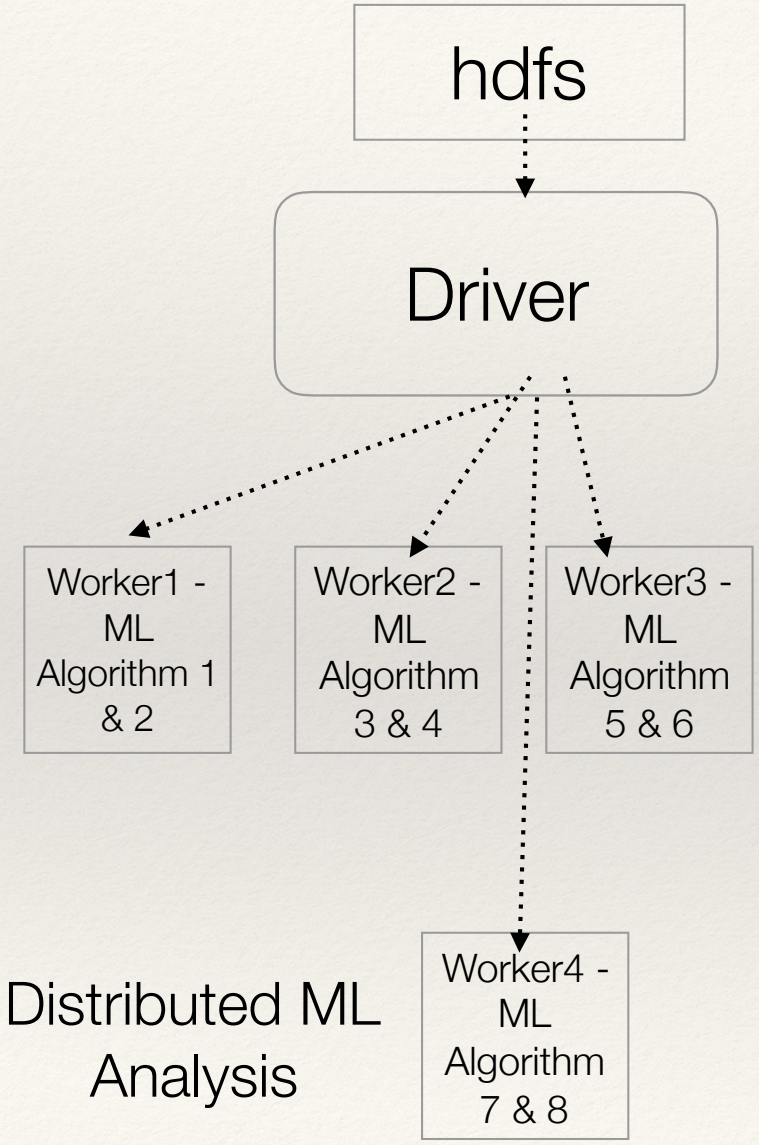
$$\mathbf{X}^\top \mathbf{X} = \begin{matrix} & \overset{n}{\cdots} & \\ d & \cdots & \\ & & \end{matrix} \begin{matrix} d \\ \vdots \\ n \end{matrix} = \sum_{i=1}^{n}$$



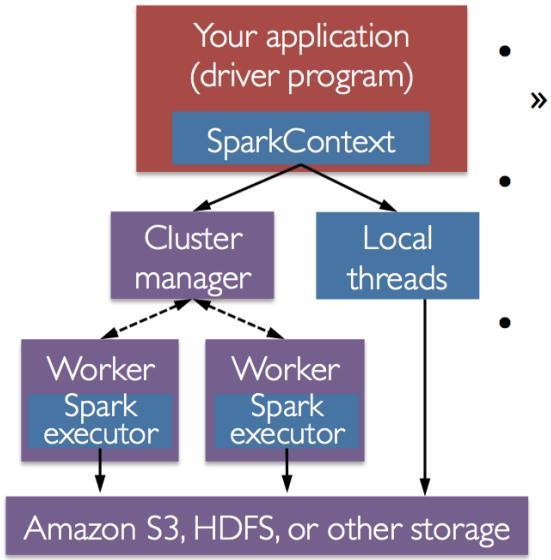
Example: n = 6; 3 workers



Data pipeline/Tools



Spark Driver and Workers



- A Spark program is two programs:
  - » A driver program and a workers program
- Worker programs run on cluster nodes or in local threads
- RDDs are distributed across workers

Demo 2/3: xgboost + ML Spark on clusters

- xgboost + Apache Spark/pyspark
- Findings: [1199] train-rmspe:0.103954 eval-rmspe:0.094526
- Validating — **RMSPE: 0.094526**
- **Data and source code can be downloaded from:** http://muppal.com

```
[1182]  train-rmspe:0.104195    eval-rmspe:0.094690
[1183]  train-rmspe:0.104192    eval-rmspe:0.094686
[1184]  train-rmspe:0.104186    eval-rmspe:0.094678
[1185]  train-rmspe:0.104167    eval-rmspe:0.094676
[1186]  train-rmspe:0.104151    eval-rmspe:0.094668
[1187]  train-rmspe:0.104136    eval-rmspe:0.094649
[1188]  train-rmspe:0.104137    eval-rmspe:0.094649
[1189]  train-rmspe:0.104120    eval-rmspe:0.094639
[1190]  train-rmspe:0.104109    eval-rmspe:0.094637
[1191]  train-rmspe:0.104107    eval-rmspe:0.094637
[1192]  train-rmspe:0.104065    eval-rmspe:0.094622
[1193]  train-rmspe:0.104015    eval-rmspe:0.094553
[1194]  train-rmspe:0.104018    eval-rmspe:0.094555
[1195]  train-rmspe:0.104014    eval-rmspe:0.094551
[1196]  train-rmspe:0.103998    eval-rmspe:0.094550
[1197]  train-rmspe:0.103995    eval-rmspe:0.094550
[1198]  train-rmspe:0.103958    eval-rmspe:0.094525
[1199]  train-rmspe:0.103954    eval-rmspe:0.094526
Validating
RMSPE: 0.094526
```

Thank you!

References

- http://rcarneva.github.io/understanding-gradient-boosting-part-1.html
- https://www.kaggle.com/c/walmart-recruiting-trip-type-classification