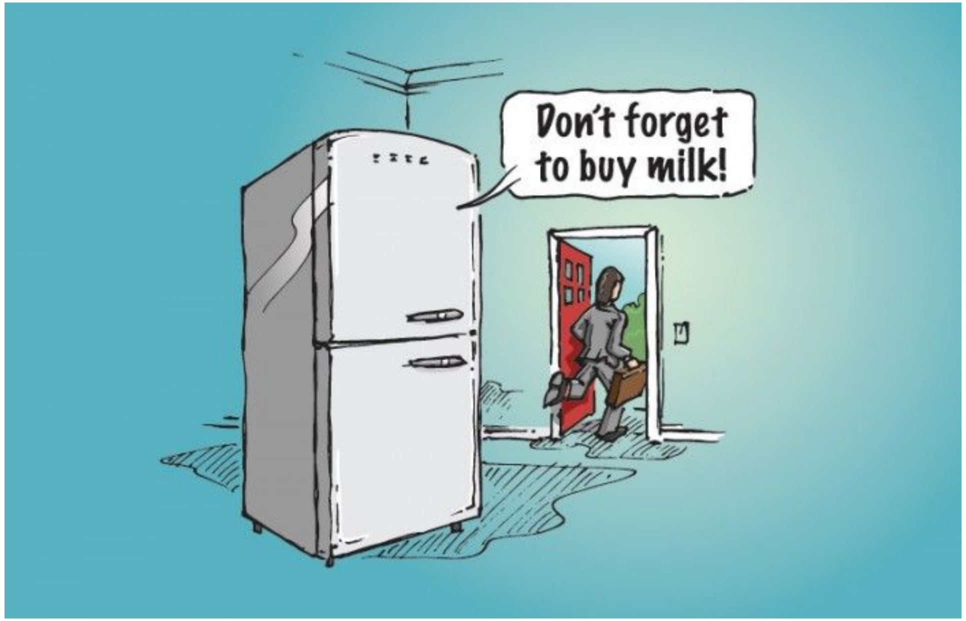
“

2017 proved to be the year of the **smart** home.

(Source: Forbes - 1/12/2018)



Sparkling SmartWater

Optimizing Energy Usage with Smart Meters using Forecasting, Machine Learning and Apache Spark

Chris Dong 🍱
Lingzhi Du 😇
Feiran Ji 🐔
Amber Song 🍭
Vanessa Zheng 🙈

Why?

We choose smart meter data because…

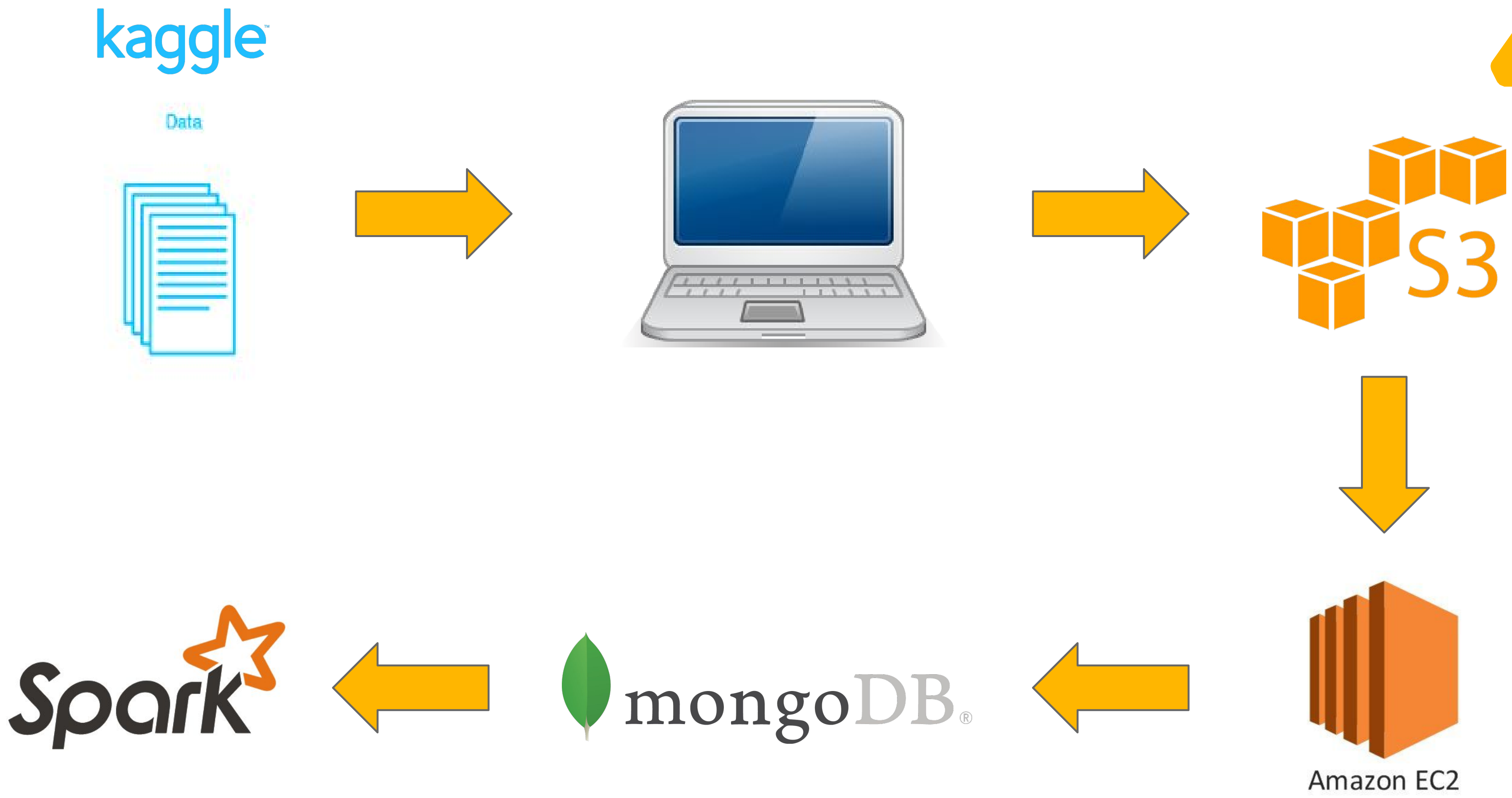
- Smart meters in every home in London
- Optimize energy usage
- High electricity prices
- Helps consumer understand their own energy consumption

Outline

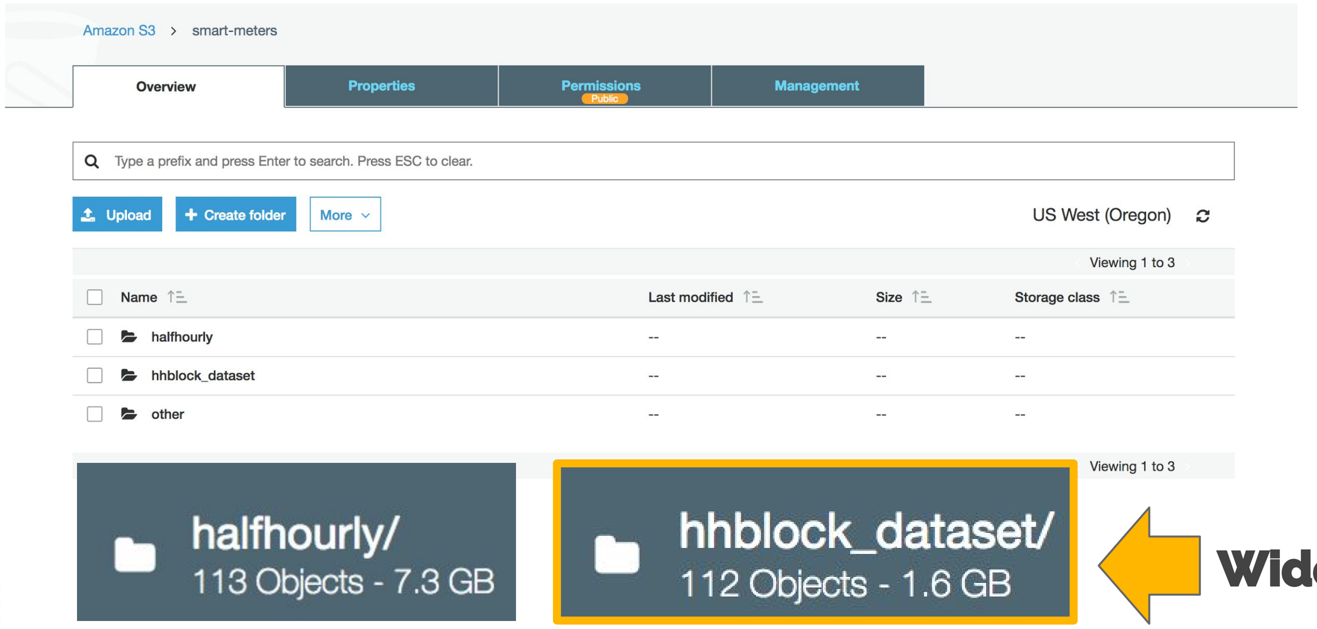
- Data Pipeline
- MongoDB
- Spark
- Machine Learning & Output
- Lesson Learned

Data Pipeline

Choose data, save to S3, import to EC2


kaggle
Data
S3
mongoDB
Spark
Amazon EC2

Amazon S3



Amazon S3 > smart-meters

Overview | Properties | Permissions (Public) | Management

Type a prefix and press Enter to search. Press ESC to clear.

Upload | Create folder | More

US West (Oregon)

Viewing 1 to 3

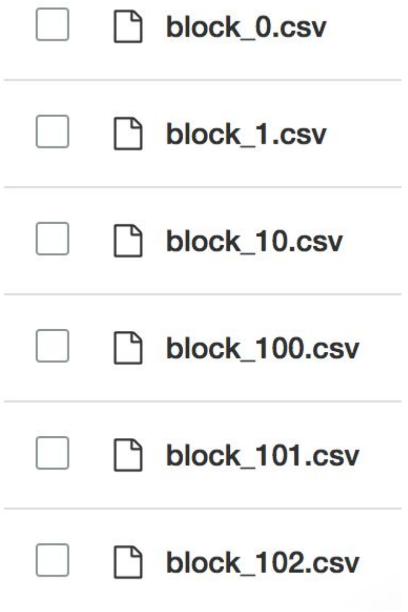
Name	Last modified	Size	Storage class
halfhourly	--	--	--
hhblock_dataset	--	--	--
other	--	--	--

Viewing 1 to 3

halfhourly/
113 Objects - 7.3 GB

hhblock_dataset/
112 Objects - 1.6 GB

Wide

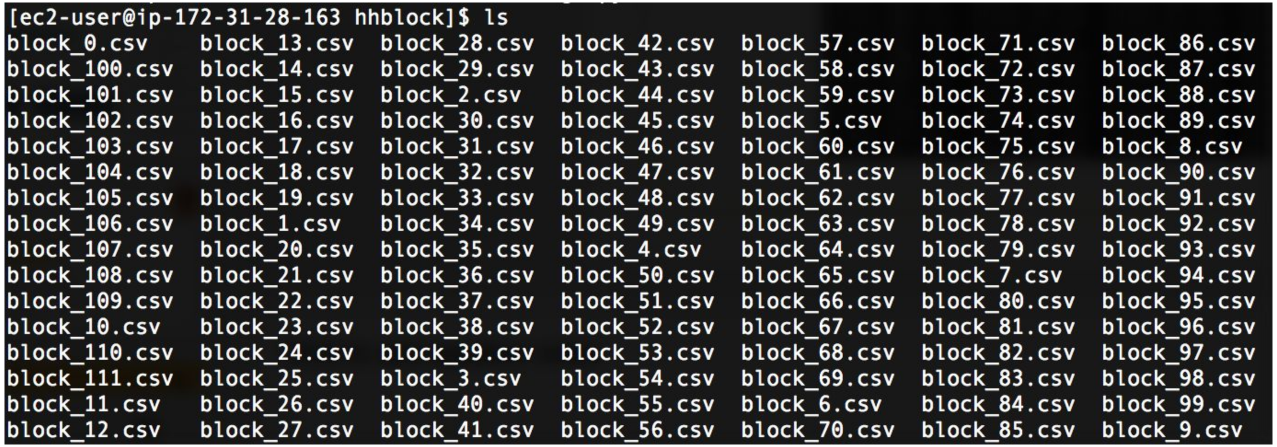


block_0.csv
block_1.csv
block_10.csv
block_100.csv
block_101.csv
block_102.csv

Data | MongoDB | Spark | ML Outcome | Lesson Learned

Importing data from S3 to EC2

- aws s3 cp s3://smart-meters/halfhourly/ . --recursive --acl public-read
- aws s3 cp s3://smart-meters/hhblock_dataset/ . --recursive --acl public-read
- aws s3 cp s3://smart-meters/other/ . --recursive --acl public-read



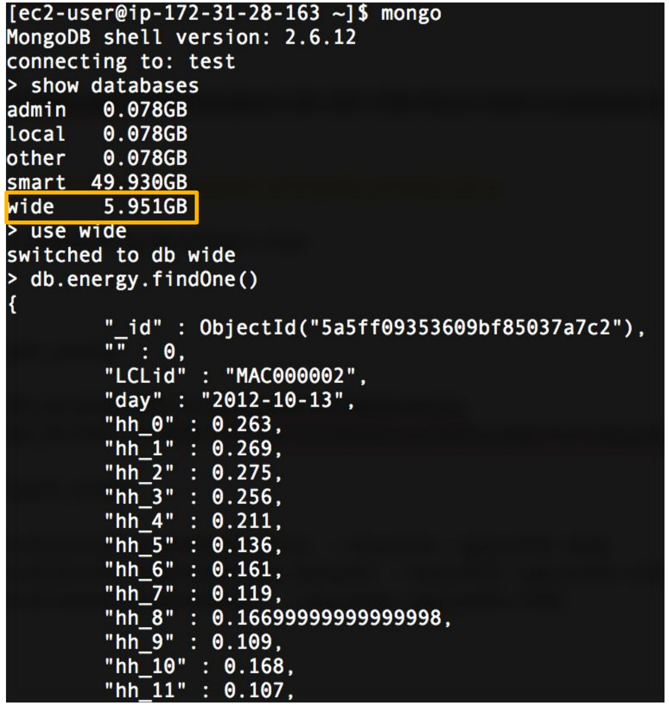
```
[ec2-user@ip-172-31-28-163 hhblock]$ ls
block_0.csv    block_13.csv  block_28.csv  block_42.csv  block_57.csv  block_71.csv  block_86.csv
block_100.csv  block_14.csv  block_29.csv  block_43.csv  block_58.csv  block_72.csv  block_87.csv
block_101.csv  block_15.csv  block_2.csv   block_44.csv  block_59.csv  block_73.csv  block_88.csv
block_102.csv  block_16.csv  block_30.csv  block_45.csv  block_5.csv   block_74.csv  block_89.csv
block_103.csv  block_17.csv  block_31.csv  block_46.csv  block_60.csv  block_75.csv  block_8.csv
block_104.csv  block_18.csv  block_32.csv  block_47.csv  block_61.csv  block_76.csv  block_90.csv
block_105.csv  block_19.csv  block_33.csv  block_48.csv  block_62.csv  block_77.csv  block_91.csv
block_106.csv  block_1.csv   block_34.csv  block_49.csv  block_63.csv  block_78.csv  block_92.csv
block_107.csv  block_20.csv  block_35.csv  block_4.csv   block_64.csv  block_79.csv  block_93.csv
block_108.csv  block_21.csv  block_36.csv  block_50.csv  block_65.csv  block_7.csv   block_94.csv
block_109.csv  block_22.csv  block_37.csv  block_51.csv  block_66.csv  block_80.csv  block_95.csv
block_10.csv   block_23.csv  block_38.csv  block_52.csv  block_67.csv  block_81.csv  block_96.csv
block_110.csv  block_24.csv  block_39.csv  block_53.csv  block_68.csv  block_82.csv  block_97.csv
block_111.csv  block_25.csv  block_3.csv   block_54.csv  block_69.csv  block_83.csv  block_98.csv
block_11.csv   block_26.csv  block_40.csv  block_55.csv  block_6.csv   block_84.csv  block_99.csv
block_12.csv   block_27.csv  block_41.csv  block_56.csv  block_70.csv  block_85.csv  block_9.csv
```

MongoDB

Importing from S3; database size; MongoDB query

Importing data into MongoDB from S3

- Add a new column indicating the filename
- for i in *.csv; do mongoimport -d smart -c energy --type csv --file $i --headerline ; done
- for i in *.csv; do mongoimport -d wide -c energy --type csv --file $i --headerline ; done
- for i in *.csv; do mongoimport -d other -c $i --type csv --file $i --headerline ; done

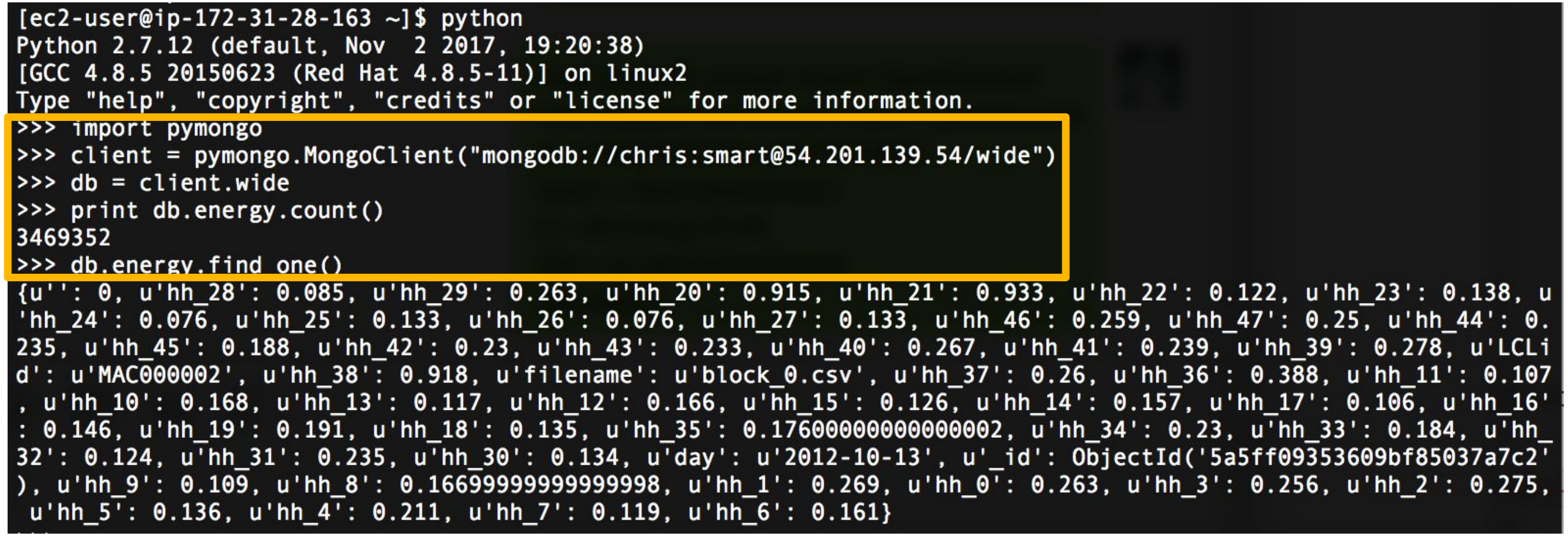


```
[ec2-user@ip-172-31-28-163 ~]$ mongo
MongoDB shell version: 2.6.12
connecting to: test
> show databases
admin    0.078GB
local    0.078GB
other    0.078GB
smart   49.930GB
wide     5.951GB
> use wide
switched to db wide
> db.energy.findOne()
{
        "_id" : ObjectId("5a5ff09353609bf85037a7c2"),
        "" : 0,
        "LCLid" : "MAC000002",
        "day" : "2012-10-13",
        "hh_0" : 0.263,
        "hh_1" : 0.269,
        "hh_2" : 0.275,
        "hh_3" : 0.256,
        "hh_4" : 0.211,
        "hh_5" : 0.136,
        "hh_6" : 0.161,
        "hh_7" : 0.119,
        "hh_8" : 0.16699999999999998,
        "hh_9" : 0.109,
        "hh_10" : 0.168,
        "hh_11" : 0.107,
```

Spark

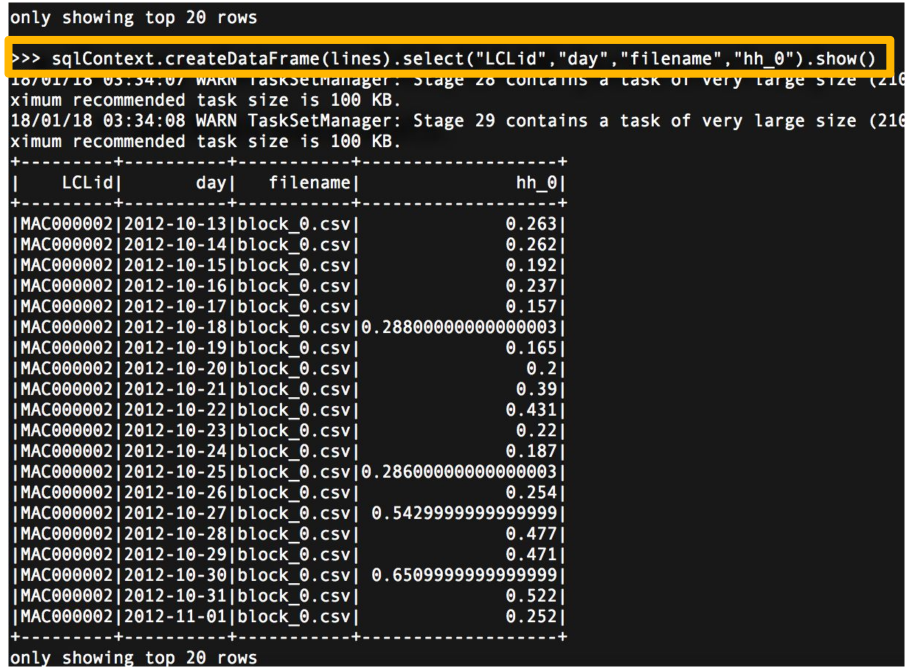
Create RDD; Spark DataFrame; Instance specs

Creating RDD from MongoDB on EC2



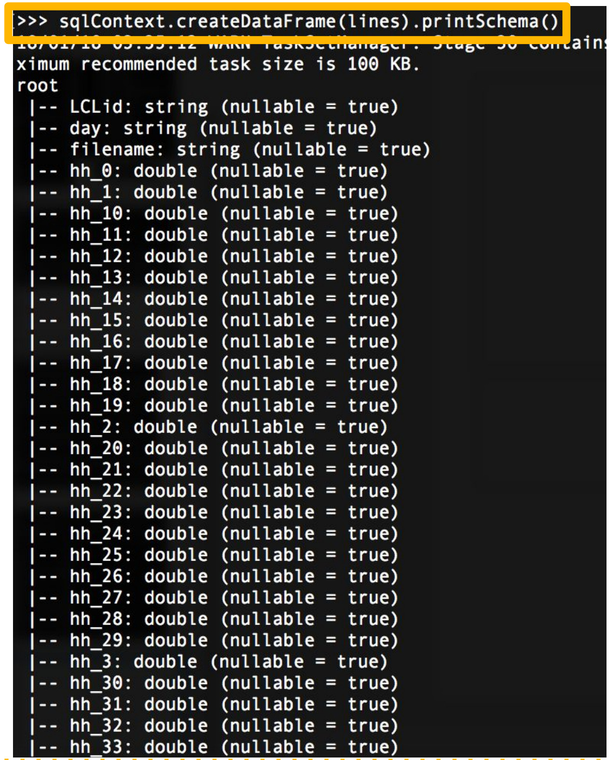
```
[ec2-user@ip-172-31-28-163 ~]$ python
Python 2.7.12 (default, Nov  2 2017, 19:20:38)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-11)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import pymongo
>>> client = pymongo.MongoClient("mongodb://chris:smart@54.201.139.54/wide")
>>> db = client.wide
>>> print db.energy.count()
3469352
>>> db.energy.find_one()
{u'': 0, u'hh_28': 0.085, u'hh_29': 0.263, u'hh_20': 0.915, u'hh_21': 0.933, u'hh_22': 0.122, u'hh_23': 0.138, u
'hh_24': 0.076, u'hh_25': 0.133, u'hh_26': 0.076, u'hh_27': 0.133, u'hh_46': 0.259, u'hh_47': 0.25, u'hh_44': 0.
235, u'hh_45': 0.188, u'hh_42': 0.23, u'hh_43': 0.233, u'hh_40': 0.267, u'hh_41': 0.239, u'hh_39': 0.278, u'LCLi
d': u'MAC000002', u'hh_38': 0.918, u'filename': u'block_0.csv', u'hh_37': 0.26, u'hh_36': 0.388, u'hh_11': 0.107
, u'hh_10': 0.168, u'hh_13': 0.117, u'hh_12': 0.166, u'hh_15': 0.126, u'hh_14': 0.157, u'hh_17': 0.106, u'hh_16'
: 0.146, u'hh_19': 0.191, u'hh_18': 0.135, u'hh_35': 0.17600000000000002, u'hh_34': 0.23, u'hh_33': 0.184, u'hh_
32': 0.124, u'hh_31': 0.235, u'hh_30': 0.134, u'day': u'2012-10-13', u'_id': ObjectId('5a5ff09353609bf85037a7c2'
), u'hh_9': 0.109, u'hh_8': 0.16699999999999998, u'hh_1': 0.269, u'hh_0': 0.263, u'hh_3': 0.256, u'hh_2': 0.275,
 u'hh_5': 0.136, u'hh_4': 0.211, u'hh_7': 0.119, u'hh_6': 0.161}
```

Querying Spark DataFrame data on EC2



```
only showing top 20 rows

>>> sqlContext.createDataFrame(lines).select("LCLid","day","filename","hh_0").show()
18/01/18 03:34:07 WARN TaskSetManager: Stage 28 contains a task of very large size (216
ximum recommended task size is 100 KB.
18/01/18 03:34:08 WARN TaskSetManager: Stage 29 contains a task of very large size (216
ximum recommended task size is 100 KB.
+---------+----------+-----------+------------------+
|    LCLid|       day|   filename|              hh_0|
+---------+----------+-----------+------------------+
|MAC000002|2012-10-13|block_0.csv|             0.263|
|MAC000002|2012-10-14|block_0.csv|             0.262|
|MAC000002|2012-10-15|block_0.csv|             0.192|
|MAC000002|2012-10-16|block_0.csv|             0.237|
|MAC000002|2012-10-17|block_0.csv|             0.157|
|MAC000002|2012-10-18|block_0.csv|0.2880000000000003|
|MAC000002|2012-10-19|block_0.csv|             0.165|
|MAC000002|2012-10-20|block_0.csv|               0.2|
|MAC000002|2012-10-21|block_0.csv|              0.39|
|MAC000002|2012-10-22|block_0.csv|             0.431|
|MAC000002|2012-10-23|block_0.csv|              0.22|
|MAC000002|2012-10-24|block_0.csv|             0.187|
|MAC000002|2012-10-25|block_0.csv|0.2860000000000003|
|MAC000002|2012-10-26|block_0.csv|             0.254|
|MAC000002|2012-10-27|block_0.csv| 0.5429999999999999|
|MAC000002|2012-10-28|block_0.csv|             0.477|
|MAC000002|2012-10-29|block_0.csv|             0.471|
|MAC000002|2012-10-30|block_0.csv| 0.6509999999999999|
|MAC000002|2012-10-31|block_0.csv|             0.522|
|MAC000002|2012-11-01|block_0.csv|             0.252|
+---------+----------+-----------+------------------+
only showing top 20 rows
```



```
>>> sqlContext.createDataFrame(lines).printSchema()
18/01/18 03:35:12 WARN TaskSetManager: Stage 30 contains
ximum recommended task size is 100 KB.
root
 |-- LCLid: string (nullable = true)
 |-- day: string (nullable = true)
 |-- filename: string (nullable = true)
 |-- hh_0: double (nullable = true)
 |-- hh_1: double (nullable = true)
 |-- hh_10: double (nullable = true)
 |-- hh_11: double (nullable = true)
 |-- hh_12: double (nullable = true)
 |-- hh_13: double (nullable = true)
 |-- hh_14: double (nullable = true)
 |-- hh_15: double (nullable = true)
 |-- hh_16: double (nullable = true)
 |-- hh_17: double (nullable = true)
 |-- hh_18: double (nullable = true)
 |-- hh_19: double (nullable = true)
 |-- hh_2: double (nullable = true)
 |-- hh_20: double (nullable = true)
 |-- hh_21: double (nullable = true)
 |-- hh_22: double (nullable = true)
 |-- hh_23: double (nullable = true)
 |-- hh_24: double (nullable = true)
 |-- hh_25: double (nullable = true)
 |-- hh_26: double (nullable = true)
 |-- hh_27: double (nullable = true)
 |-- hh_28: double (nullable = true)
 |-- hh_29: double (nullable = true)
 |-- hh_3: double (nullable = true)
 |-- hh_30: double (nullable = true)
 |-- hh_31: double (nullable = true)
 |-- hh_32: double (nullable = true)
 |-- hh_33: double (nullable = true)
```

AWS Instance Specs

EC2: 1 Instance:

- t2.large
- 2 cores
- 8 GB RAM
- 300 GB Storage
- 0.09/hour

Standalone : 5 instances, 4 workers

- C3.2xlarge
- 8 cores
- 15 GB RAM
- 160 GB Storage
- 2.1/hour

YARN : 5 instances, 4 workers

- C3.2xlarge
- 16 cores
- 15 GB RAM
- 160 GB Storage
- 2.1/hour

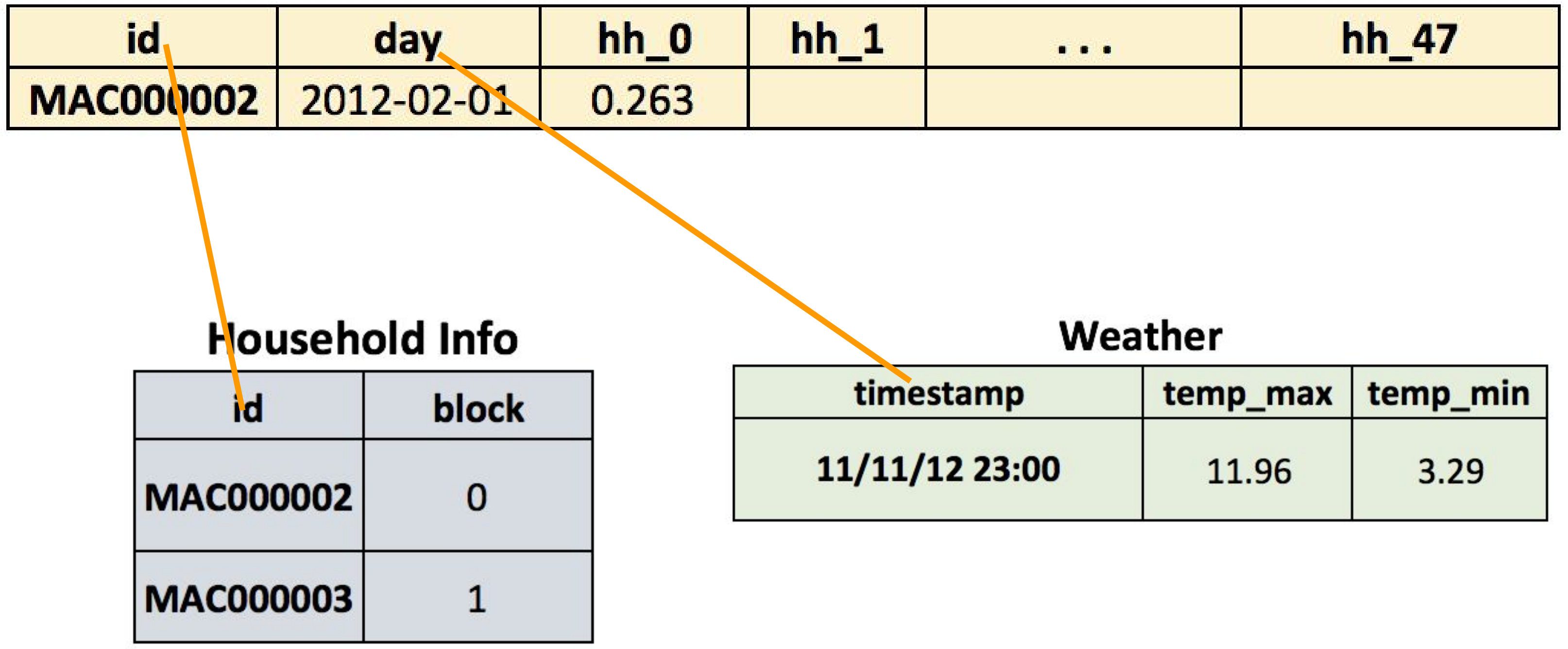
YARN: 1 Instance

- C3.8xlarge
- 32 cores
- 60 GB RAM
- 640 GB Storage
- 1.68/hour

Machine Learning

Data Overview; Analytic Goals; Feature Engineering; Model; Specs

Data Overview



Energy Use per Half-Hour (Wide)

id	day	hh_0	hh_1	. . .	hh_47
MAC000002	2012-02-01	0.263			

Household Info

id	block
MAC000002	0
MAC000003	1

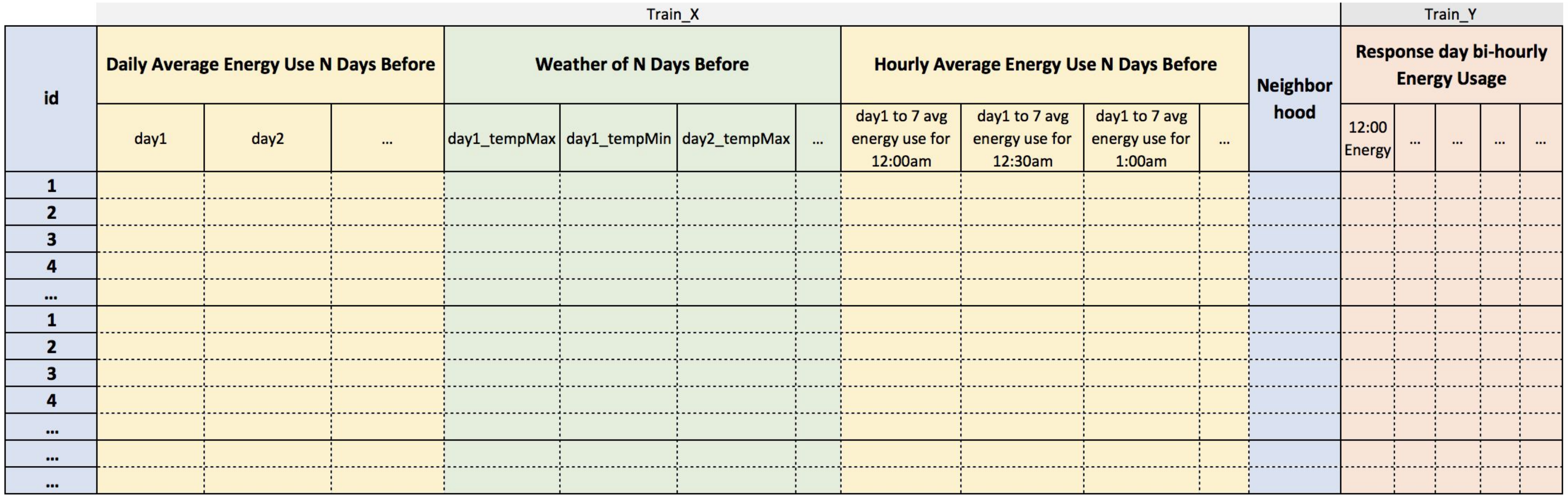
Weather

timestamp	temp_max	temp_min
11/11/12 23:00	11.96	3.29

Analytical Goals

- **Goal:**
  - To predict bi-hourly energy usage for one day ahead (12 data points)
- **Model approach:**
  - Spark ML RandomForestRegressor()
- **Challenge:**
  - To solve a time series problem with ML
- **Key:**
  - Feature engineering

Feature Engineering

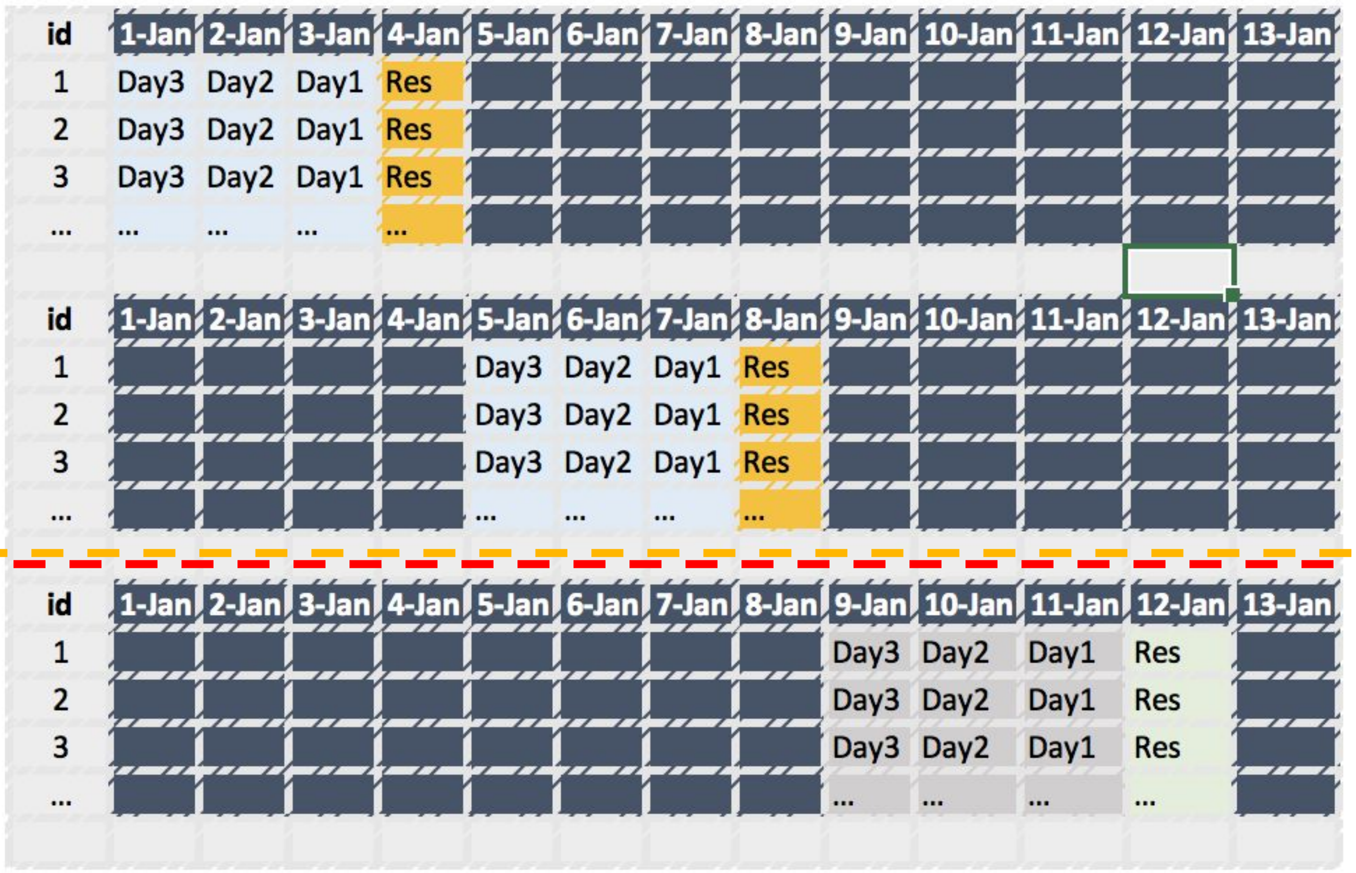


	Train_X													Train_Y				
id	Daily Average Energy Use N Days Before			Weather of N Days Before				Hourly Average Energy Use N Days Before				Neighbor hood	Response day bi-hourly Energy Usage				
	day1	day2	…	day1_tempMax	day1_tempMin	day2_tempMax	…	day1 to 7 avg energy use for 12:00am	day1 to 7 avg energy use for 12:30am	day1 to 7 avg energy use for 1:00am	…		12:00 Energy	…	…	…	…
1																	
2																	
3																	
4																	
…																	
1																	
2																	
3																	
4																	
…																	
…																	
…																	

* day1 = one day before label day

* day2 = two days before label day

* e.g., if label day is 2012-02-01, day1 = 2012-01-31, day2 = 2012-01-30



id	1-Jan	2-Jan	3-Jan	4-Jan	5-Jan	6-Jan	7-Jan	8-Jan	9-Jan	10-Jan	11-Jan	12-Jan	13-Jan
1	Day3	Day2	Day1	Res									
2	Day3	Day2	Day1	Res									
3	Day3	Day2	Day1	Res									
...	...	...	...	...									

id	1-Jan	2-Jan	3-Jan	4-Jan	5-Jan	6-Jan	7-Jan	8-Jan	9-Jan	10-Jan	11-Jan	12-Jan	13-Jan
1					Day3	Day2	Day1	Res					
2					Day3	Day2	Day1	Res					
3					Day3	Day2	Day1	Res					
...					...	...	...	...					

Training

id	1-Jan	2-Jan	3-Jan	4-Jan	5-Jan	6-Jan	7-Jan	8-Jan	9-Jan	10-Jan	11-Jan	12-Jan	13-Jan
1									Day3	Day2	Day1	Res	
2									Day3	Day2	Day1	Res	
3									Day3	Day2	Day1	Res	
...									...	...	...	...	

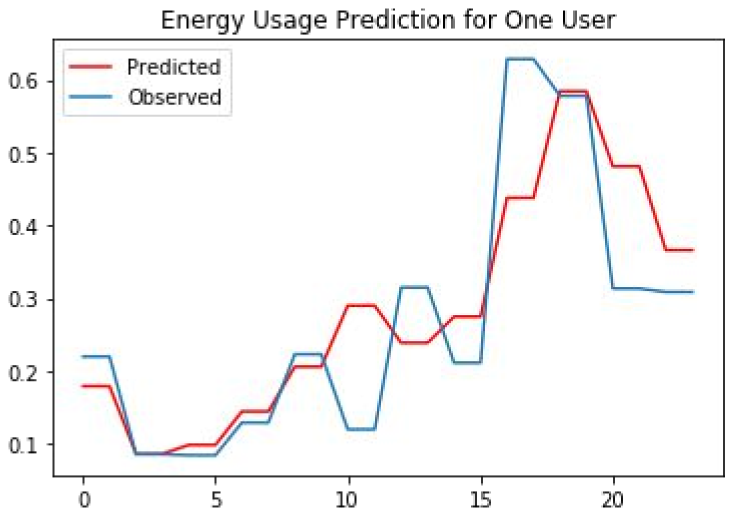
Validation

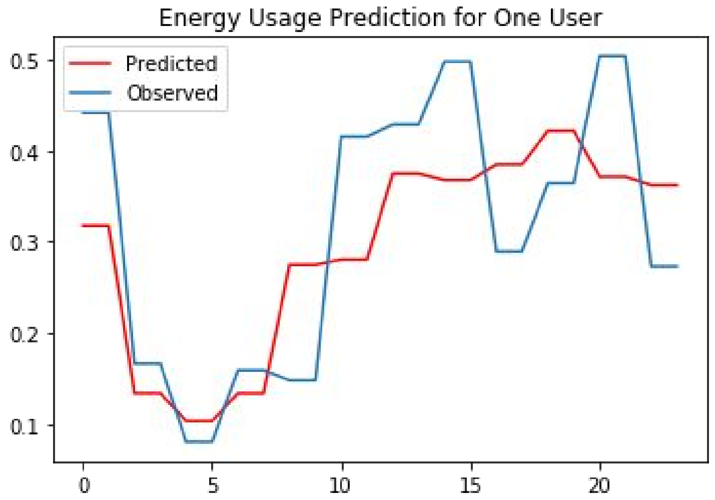
Model Performance

- Model approach: Spark ML RandomForestRegressor()
- Evaluation Metric: RMSE

Period	RMSE	Period	RMSE
1	0. 1246	7	0. 1979
2	0. 1085	8	0. 1958
3	0. 0932	9	0. 2111
4	0. 1038	10	0. 2181
5	0. 1578	11	0. 1954
6	0. 1924	12	0. 1629

Energy Usage Predictions





Time per model

Attempt	Data preprocessing	Model training	Number of Trees in RF
1	Spark SQL	Spark ML	10
2	Pandas	Spark ML	10
3	Pandas	Spark ML	300

Attempt	single ec2 instance	Standalone 5 instances	Yarn 5 instances	Yarn 3 instances	Yarn 1 instances
1	forever	3602 s	3478 s		3186 s
2	34.8 s		36.7 s	35.3 s	43.6 s
3	error		500.0 s	546.0 s	937.4 s

AWS Instance Specs

EC2: 1 Instance:

- Not enough memory

YARN : 5 instances, 4 workers

- 8.3 min/model
- $0.29/model

YARN : 3 instances, 2 workers

- 9.1 min/model
- $0.38/model

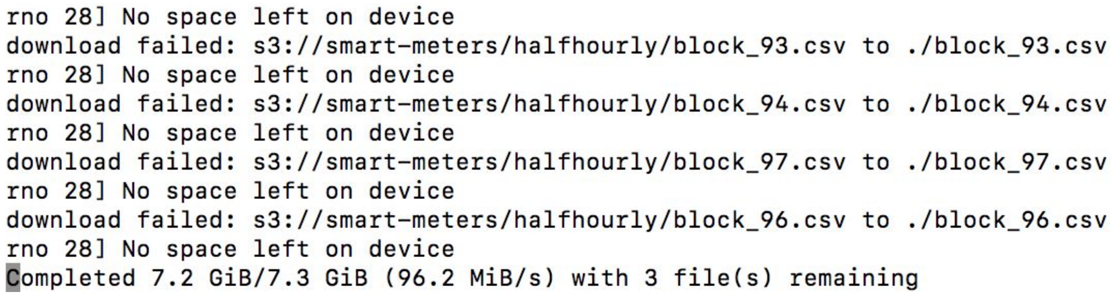
YARN: 1 Instance

- 15.7 min/model
- $0.44/model

Lesson Learned 🎁

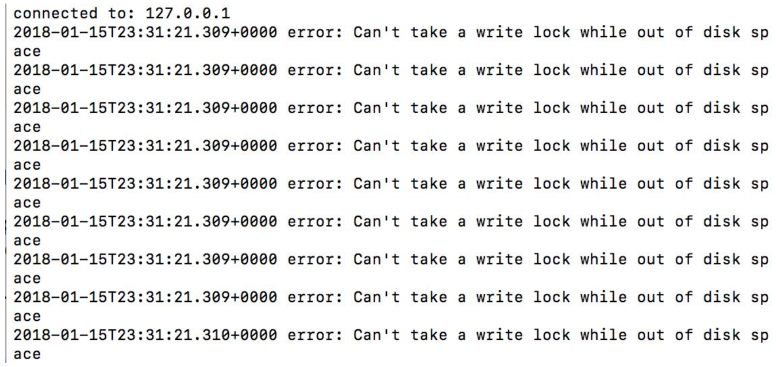
Memory & Storage; Errors

Not Enough Storage ~~8 GB, 16GB, 30GB, 100GB,~~ 300GB



```
rno 28] No space left on device
download failed: s3://smart-meters/halfhourly/block_93.csv to ./block_93.csv
rno 28] No space left on device
download failed: s3://smart-meters/halfhourly/block_94.csv to ./block_94.csv
rno 28] No space left on device
download failed: s3://smart-meters/halfhourly/block_97.csv to ./block_97.csv
rno 28] No space left on device
download failed: s3://smart-meters/halfhourly/block_96.csv to ./block_96.csv
rno 28] No space left on device
Completed 7.2 GiB/7.3 GiB (96.2 MiB/s) with 3 file(s) remaining
```

S3 to EC2



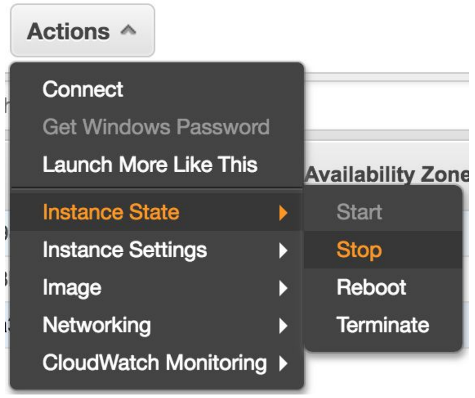
```
connected to: 127.0.0.1
2018-01-15T23:31:21.309+0000 error: Can't take a write lock while out of disk sp
ace
2018-01-15T23:31:21.309+0000 error: Can't take a write lock while out of disk sp
ace
2018-01-15T23:31:21.309+0000 error: Can't take a write lock while out of disk sp
ace
2018-01-15T23:31:21.309+0000 error: Can't take a write lock while out of disk sp
ace
2018-01-15T23:31:21.309+0000 error: Can't take a write lock while out of disk sp
ace
2018-01-15T23:31:21.309+0000 error: Can't take a write lock while out of disk sp
ace
2018-01-15T23:31:21.309+0000 error: Can't take a write lock while out of disk sp
ace
2018-01-15T23:31:21.309+0000 error: Can't take a write lock while out of disk sp
ace
2018-01-15T23:31:21.310+0000 error: Can't take a write lock while out of disk sp
ace
```

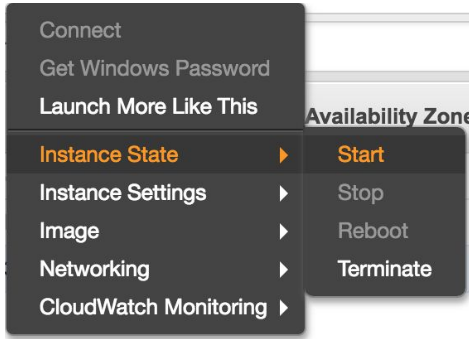
MongoDB on EC2:

- 7.87 GB -> 49.3 GB

Instance Dying

- Port 22 “Operation Timed Out”
  - Solution: Start and stop instance with New IP
  - Cause: All of us using same instance at same time
  - How to avoid: Run model on separate instances





Thanks!