Twitter Sentiment Analysis for Stock Market Prediction at Scale

Kirby Bloom, Ravi Ayyappan, Nishant Velagapudi, Brandon Cummings
University of California, Berkeley, W251 - Scaling Up, Really Big Data

Introduction

- Role of Social Media in capturing people’s sentiments
  - Warehouse of emotions
  - People share their happiness, sadness, frustrations and anger
  - Provides an excellent platform to understand and react to consumer attitudes
- Case Study - Stock market prediction

Goal

- Build a framework that obtains, analyzes and classifies sentiments of a stream of tweets
  - Scalable
- Proof of concept - tie tagged tweets to external data of interest
  - Case study - stock prices
    - Analytics
    - Prediction

Data Source & Elements

Live Twitter Stream

Filtering based on S&P 100 company names

~ 600,000 tweets per day

Case: Stock Data Collection

S&P 100 Companies, resulting in 102 stock ticker symbols

300,000 Per Minute Stock Changes Collected

Daily cron worker at 5:01pm to submit spark job

https://api.iextrading.com/1.0/stock/aapl/chart/1d?format=json

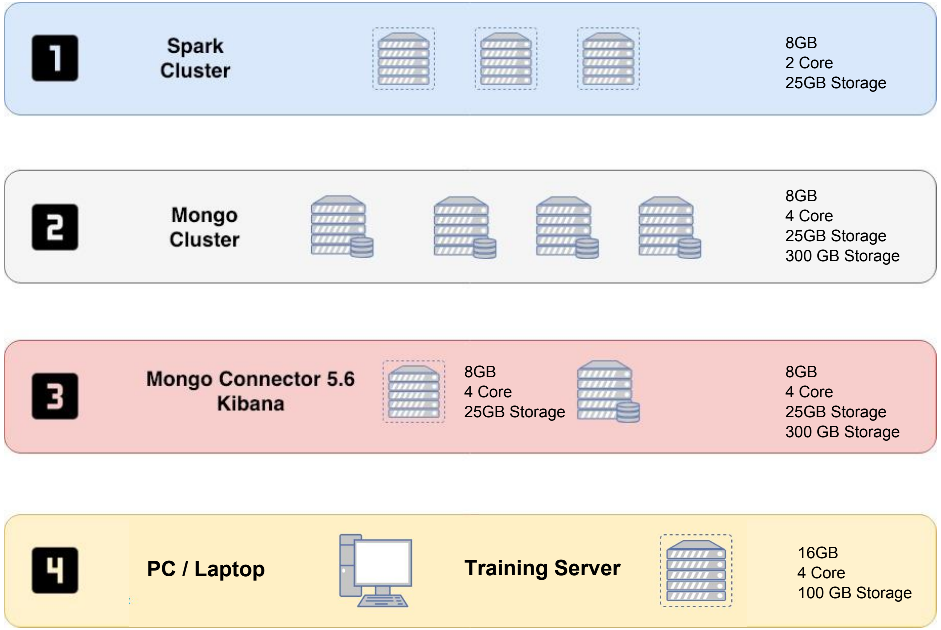
(per minute open json API)

Infrastructure

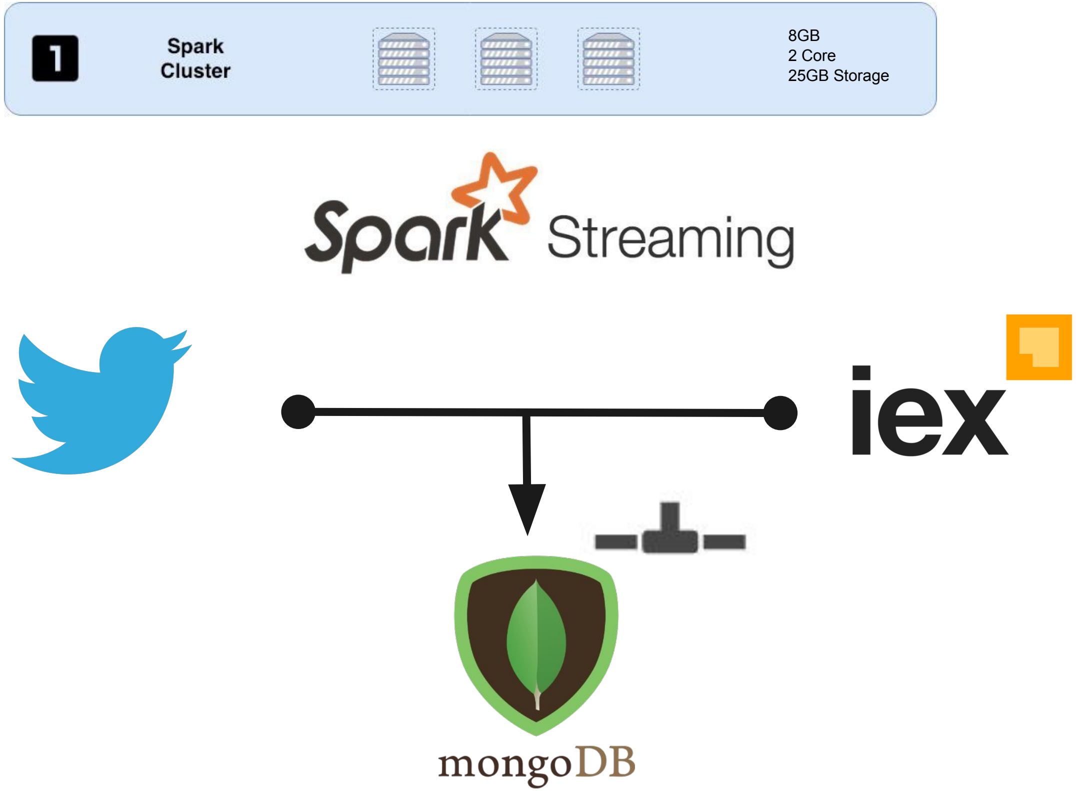
Approach

1. Gather
2. Store
3. Optimize
4. Analyze

Topology
(SJC)



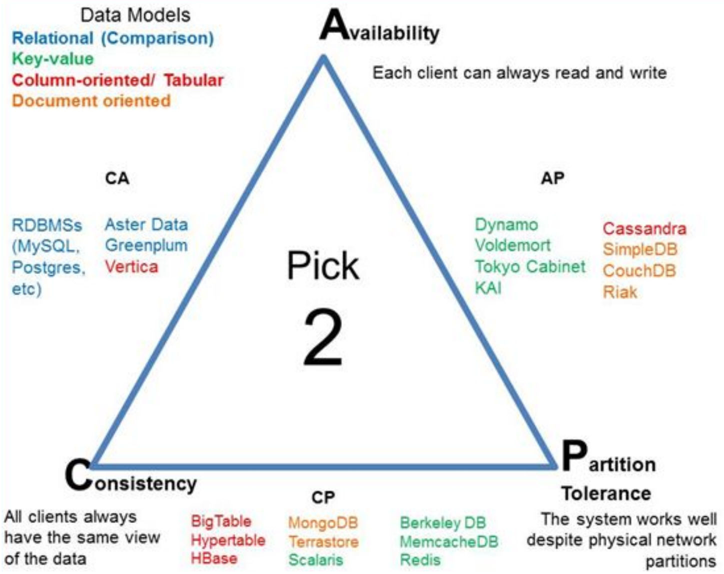
Gather


1
Spark
Cluster
8GB
2 Core
25GB Storage
Spark Streaming
iex
mongoDB

Store

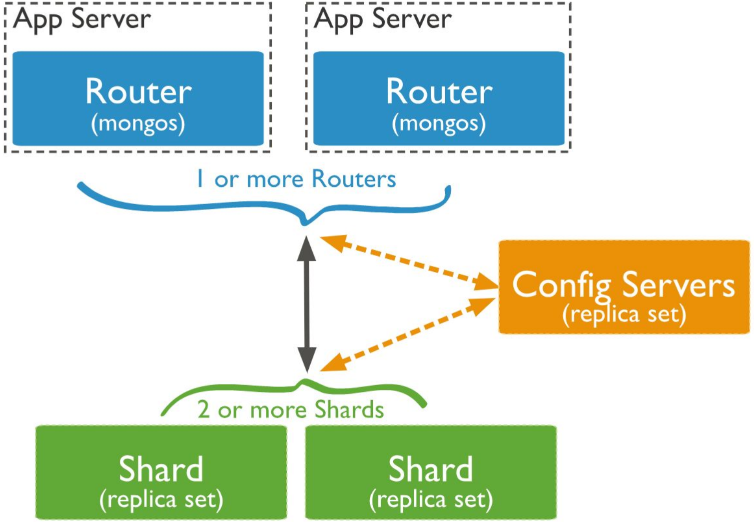
2 Mongo Cluster

8GB
4 Core
25GB Storage
300 GB Storage



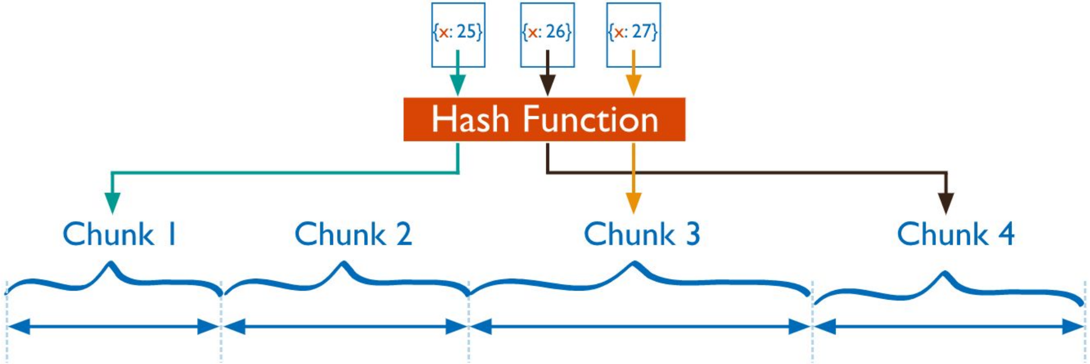
Store


2
Mongo
Cluster


App Server
Router
(mongos)
App Server
Router
(mongos)
1 or more Routers
Config Servers
(replica set)
2 or more Shards
Shard
(replica set)
Shard
(replica set)

Store


2
Mongo
Cluster


{x: 25}
{x: 26}
{x: 27}
Hash Function
Chunk 1
Chunk 2
Chunk 3
Chunk 4

Store


```
sharding version: {
        "_id" : 1,
        "minCompatibleVersion" : 5,
        "currentVersion" : 6,
        "clusterId" : ObjectId("5bfe35e53d216d3bf77d1c17")
}
shards:
        {  "_id" : "stock-tweet-2",  "host" : "stock-tweet-2/169.53.133.188:27022",  "state" : 1 }
        {  "_id" : "stock-tweet-3",  "host" : "stock-tweet-3/169.53.133.187:27022",  "state" : 1 }
        {  "_id" : "stock-tweet-4",  "host" : "stock-tweet-4/169.53.133.190:27022",  "state" : 1 }
        {  "_id" : "stock-tweet-5",  "host" : "stock-tweet-5/169.53.133.180:27022",  "state" : 1 }
active mongoses:
        "4.0.4" : 1
autosplit:
        Currently enabled: yes
balancer:
        Currently enabled:  yes
        Currently running:  no
        Failed balancer rounds in last 5 attempts:  0
        Migration Results for the last 24 hours:
                No recent migrations
```

Store

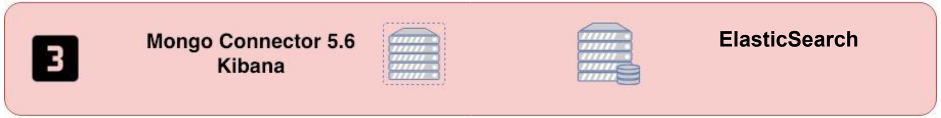
Qualitative
(Tweets)

```
qualitative_stock_db.tweets
        shard key: { "tweet_id" : "hashed" }
        unique: false
        balancing: true
        chunks:
                stock-tweet-2   21
                stock-tweet-3   22
                stock-tweet-4   21
                stock-tweet-5   22
```

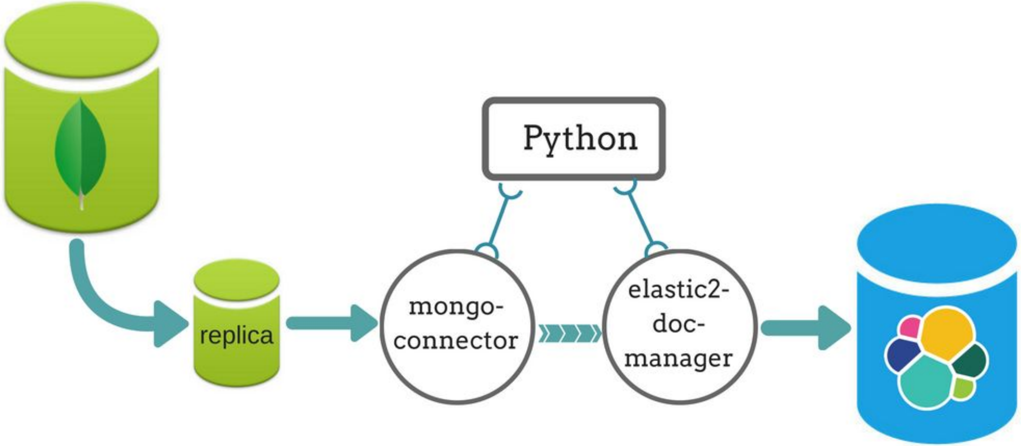
Quantitative
(Stocks)

```
quantitative_stock_db.ticker_scrapes
        shard key: { "ticker_scrap_id" : "hashed" }
        unique: false
        balancing: true
        chunks:
                stock-tweet-2   2
                stock-tweet-3   2
                stock-tweet-4   2
                stock-tweet-5   2
```

Optimize

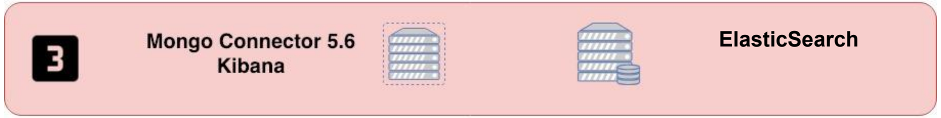


3
Mongo Connector 5.6
Kibana
ElasticSearch



Python
replica
mongo-connector
elastic2-doc-manager

Optimize




```
{
  "namespaces": {
    "db.included_collection": true,
    "db.filtered_collection1": {
      "includeFields": ["included_field", "included.nested.field"]
    },
    "db.filtered_collection2": {
      "excludeFields": ["excluded_field", "excluded.nested.field"]
    },
    "filtered_database.*": {
      "includeFields": ["included_field", "included.nested.field"]
    },
    "filtered_renamed_database.*": {
      "rename": "new_filtered_database.*",
      "includeFields": ["included_field", "included.nested.field"]
    }
  }
}
```

```
root@search1:~# curl "169.53.133.184:9200/_cat/indices"
yellow open quantitative_stock_db SHrR0gx5TvCunxDirJbwsQ 5 1  358020   15365  65.2mb  65.2mb
yellow open qual_copy             8chgQDmrSi2hnIyWltidgg 5 1 3384227  762647   4.8gb   4.8gb
yellow open mongodb_meta          d9GsGQl5S5WWKu357lK4oQ 5 1 4321678 1478649 230.6mb 230.6mb
yellow open .kibana               Za3HvpyGRcmRATt9nPyZcg 1 1      25       1  45.3kb  45.3kb
yellow open cleanedstocks         mtzdmPBwQdi_z3Dm27bndw 5 1   35802    6302     4mb     4mb
yellow open qual_conv_stock_db    QEyP5592ToG8P3TboH54zQ 5 1 3665277 1022863   5.5gb   5.5gb
yellow open qualitative_stock_db  Oy9zedxbQhiDb4vQUdLrPA 5 1 3673550 1544783   6.3gb   6.3gb
yellow open config                8jzeYFHzTieq5xcfqeFYrA 5 1      74      70 235.4kb 235.4kb
yellow open testdb                pENPlfDTRvCSnu3DorwWNA 5 1  210972   40668 325.6mb 325.6mb
```

Throughput

QuantCount

QuantCount

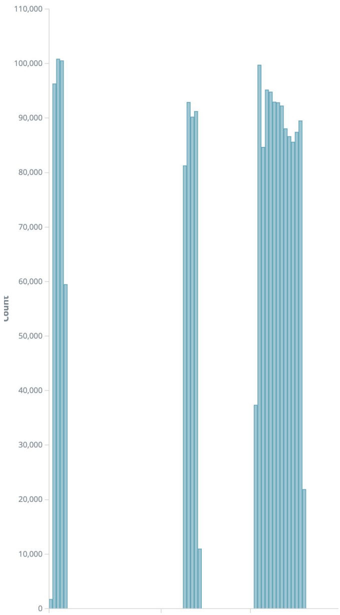
Count

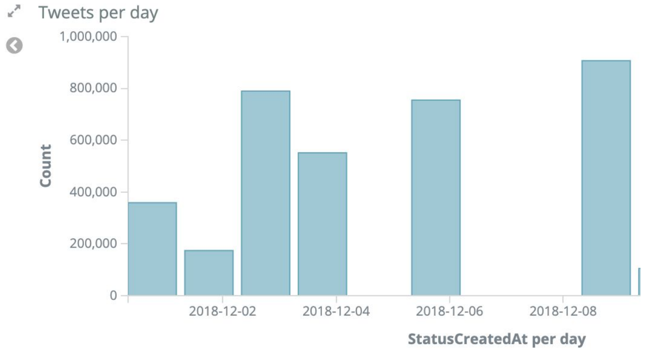
35,802

Tweet Count

Count

3,665,277





Reflection

- Pros
  - Fault tolerant
    - Node failure
    - Replay / restructure
  - Very flexible
    - Lot of configuration points
    - Various secondary data sources
  - Horizontally scalable
    - Built as cluster(s)
- Cons
  - Longer setup time
  - Larger footprint to maintain
    - Nodes
    - Daemons
  - Cost
    - Overbuilt in spots

Analysis

Sentiment

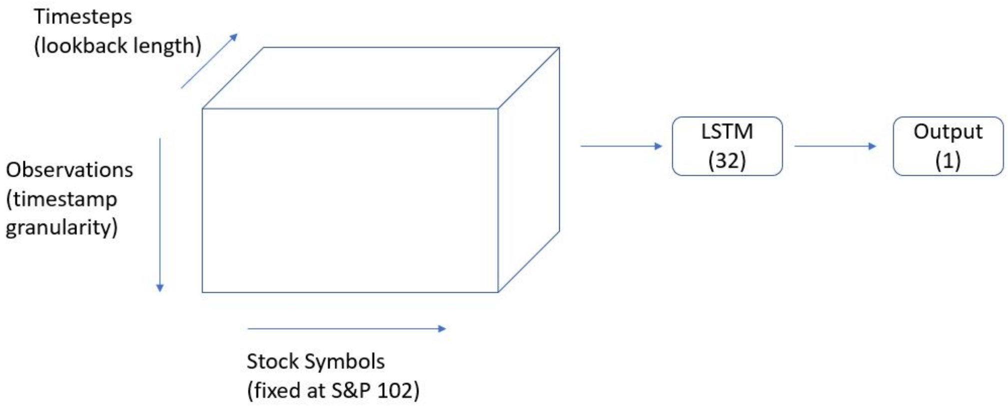
Sentiment was calculated using the Stanford CoreNLP pipeline

- Base sentence represented as a tree
- Sentiment of each node is calculated using a pre-trained RNN for continuous scores
  - Sub-tree elements aggregated into a document-level score
- Outputs range from 1-5
  - Unusual to find 4 and 5 scores
- Concerns about CoreNLP Sentiment Annotation RNN suitability for twitter

Analysis - Prediction

RNN for stock price change prediction:

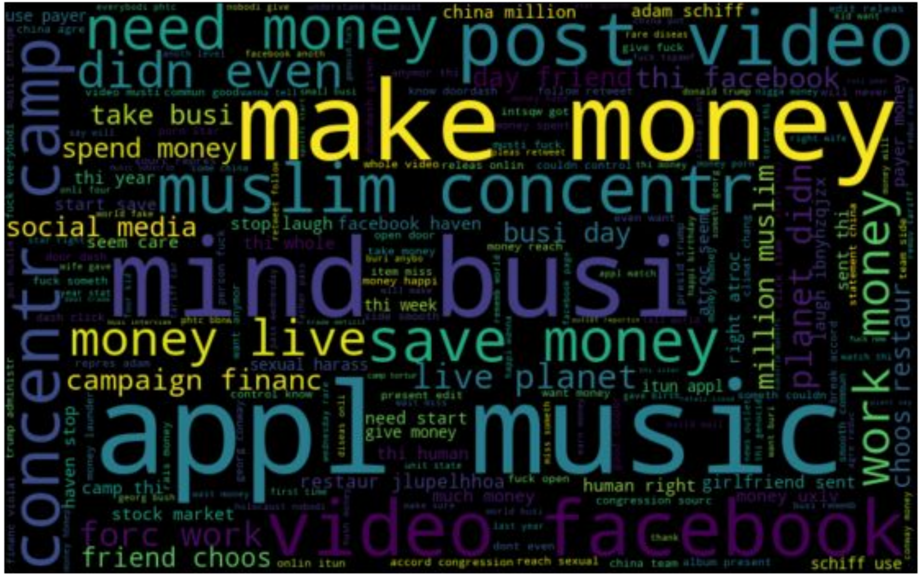
- Aggregate data into arbitrary granularity
- Single-layer LSTM architecture
- Predict price of one ticker symbol
  - Uses variable amount of data
- Stock only - Poor performance
  - 9 days of data for train+eval
  - Test time periods show behavior unseen in training
- Improved performance with Sentiment



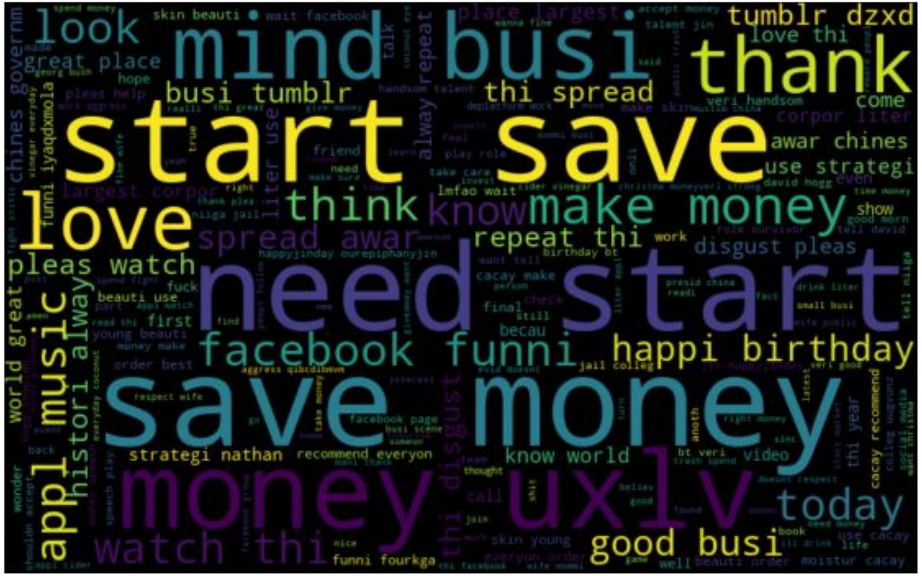
RNN Results

	Amazon		Facebook		Apple	
Minutes	Avg_shift	RMSE	Avg_shift	RMSE	Avg_shift	RMSE
1	0.0166	0.012	0.0105	0.61	0.0023	0.031
3	0.0494	0.0287	0.0317	0.104	0.00684	0.124
5	0.0833	0.13	0.0525	0.134	0.0122	0.0224
10	0.166	0.114	0.106	0.109	0.022	0.258
30 + Sentiment	**0.352**	**0.309**	**0.325**	**0.171**	**0.386**	**0.783**

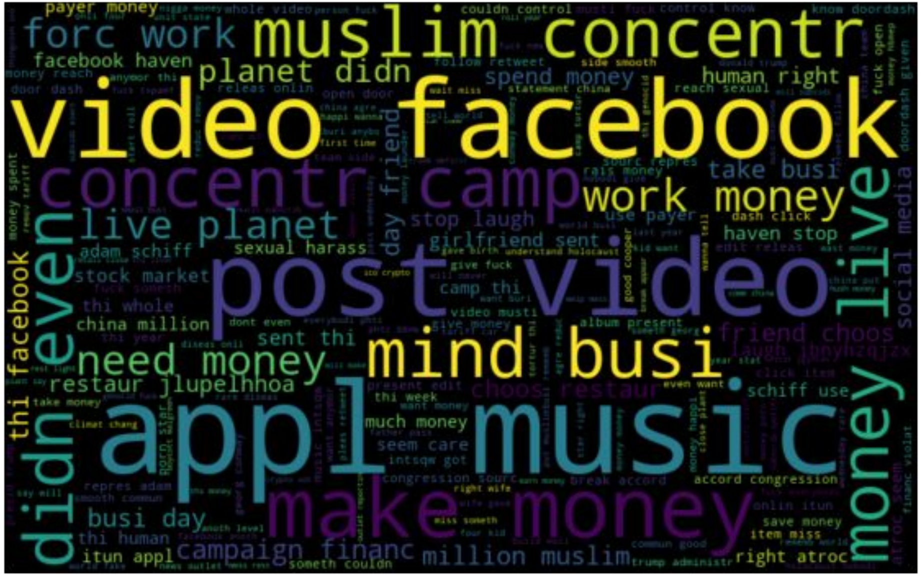
Visualizations - Wordcloud (All words)



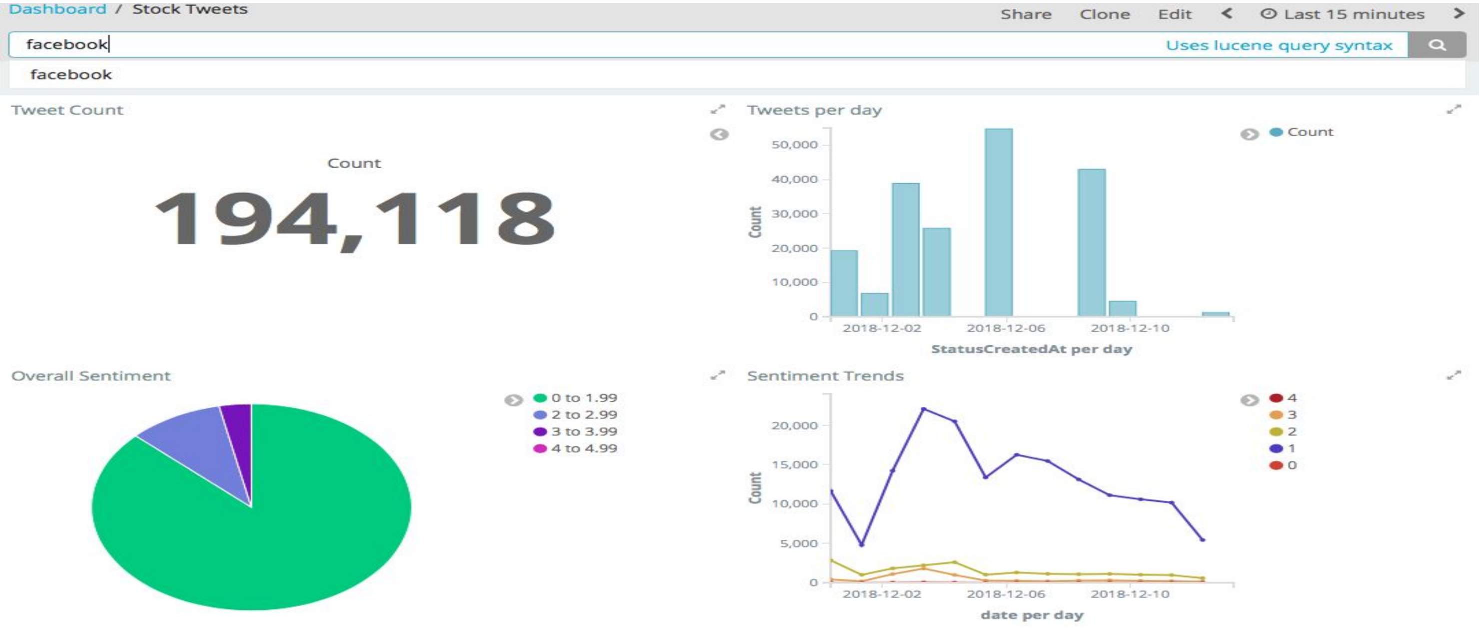
Visualizations - Wordcloud (Positive Sentiment)



Visualizations - Wordcloud (Negative Sentiment)



Visualizations



Conclusion

- Created scalable architecture for tweet collection, scoring, analysis
- Collected ~3.6m tweets (~360k/day)
- Calculated sentiment for incoming tweets using pretrained RNN
- Developed Kibana dashboards for exploratory analysis on sentiment & stock prices
- Trained RNN for stock price prediction
    - Using only historical stock price - poor performance
    - Using historical prices + related tweet sentiments - better performance
- Future work: challenge the infrastructure with more data

Thanks