

# Elasticsearch & Lucene for Apache Spark and MLlib

Costin Leau (@costinl)



SPARK SUMMIT 2016  
DATA SCIENCE AND ENGINEERING AT SCALE  
JUNE 6-8, 2016 SAN FRANCISCO



---

**Mirror, mirror on the wall,  
what's the happiest team of  
us all ?**

---

*Briita Weber*

*- Rough translation from German by yours truly -*

# Purpose of the talk

Improve ML pipelines through IR

Text processing

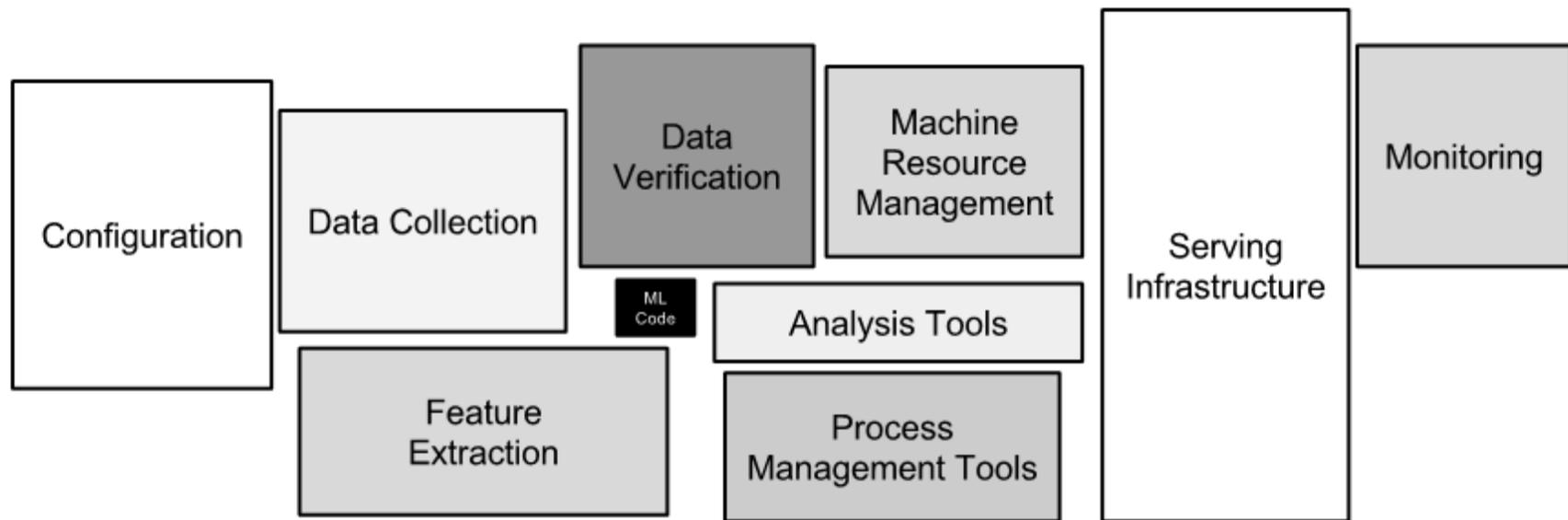
- Analysis
- Featurize/Vectorize \*

\* In research / poc / WIP / Experimental phase



SPARK SUMMIT 2016

# Technical Debt



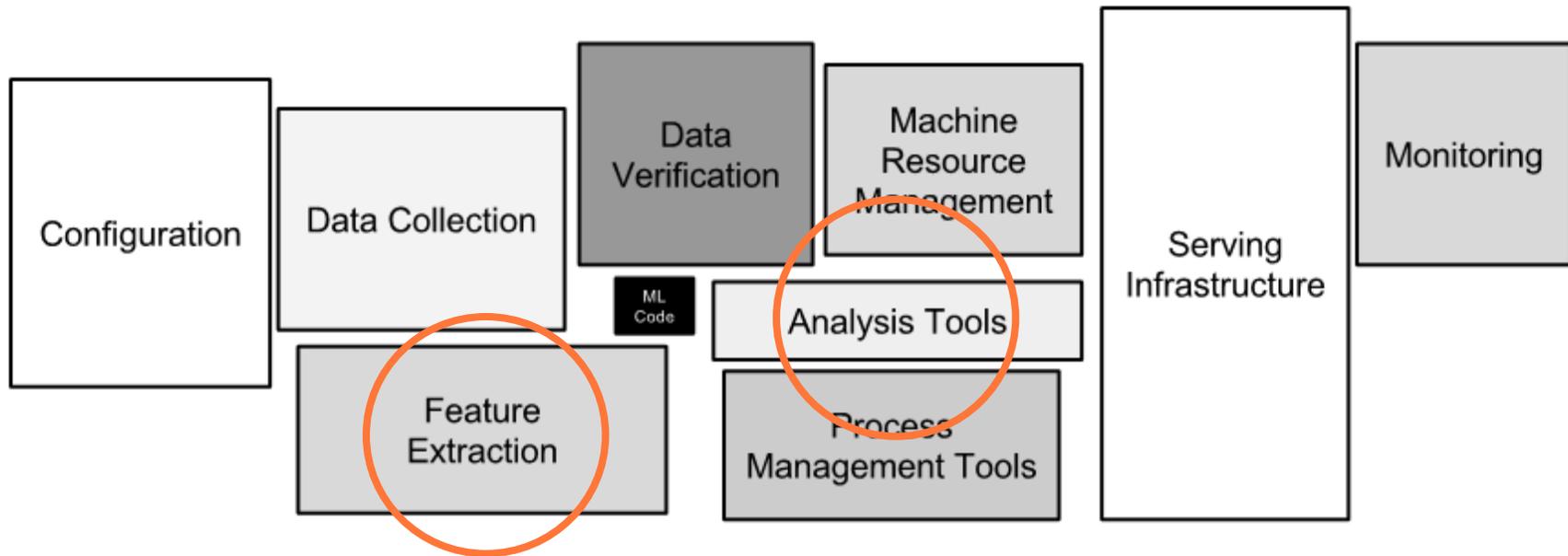
Machine Learning: The High Interest Credit Card of Technical Debt”, Sculley et al

<http://research.google.com/pubs/pub43146.html>



SPARK SUMMIT 2016

# Technical Debt



Machine Learning: The High Interest Credit Card of Technical Debt”, Sculley et al

<http://research.google.com/pubs/pub43146.html>



SPARK SUMMIT 2016

# Challenge



**SPARK SUMMIT 2016**  
DATA SCIENCE AND ENGINEERING AT SCALE  
JUNE 6-8, 2016 SAN FRANCISCO

# Challenge: What team at Elastic is most happy?

Data: Hipchat messages

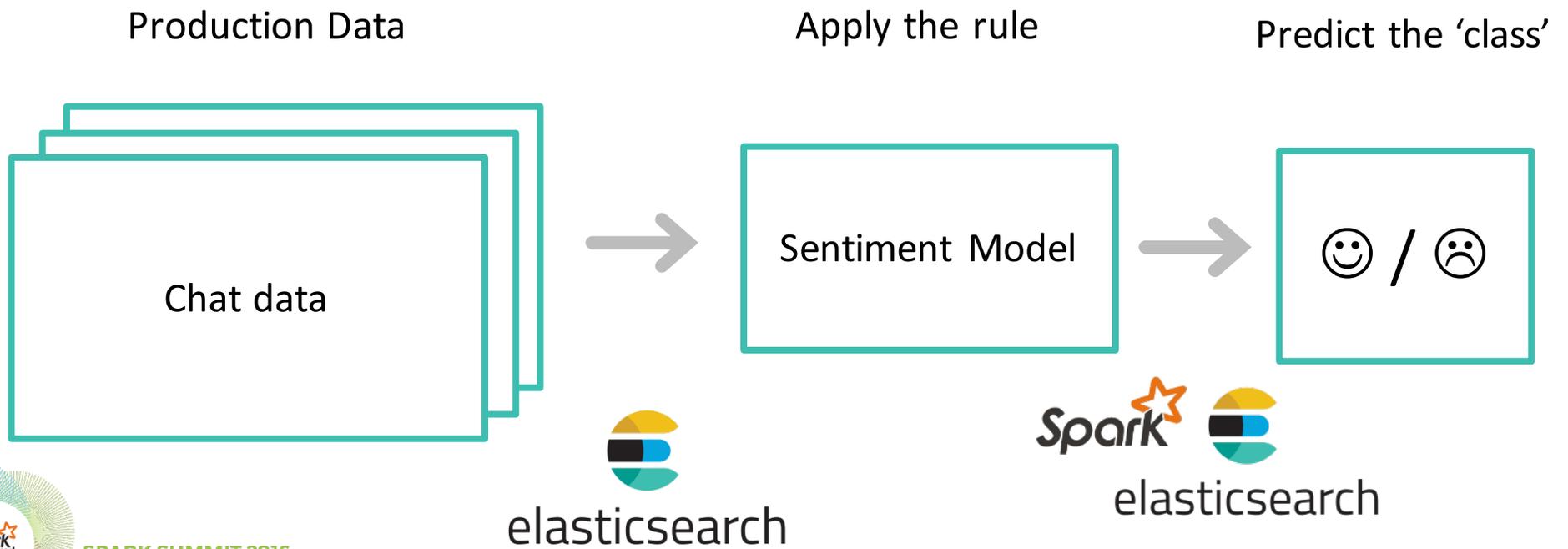
Training / Test data: <http://www.sentiment140.com>

Result: Kibana dashboard



SPARK SUMMIT 2016

# ML Pipeline



SPARK SUMMIT 2016

# Data is King



**SPARK SUMMIT 2016**  
DATA SCIENCE AND ENGINEERING AT SCALE  
JUNE 6-8, 2016 SAN FRANCISCO

## Example: Word2Vec

Input snippet

```
it was introduced into mathematics in the book  
disquisitiones arithmeticae by carl friedrich gauss in  
one eight zero one ever since however modulo has gained  
many meanings some exact and some imprecise
```

<http://spark.apache.org/docs/latest/mllib-feature-extraction.html#example>



SPARK SUMMIT 2016

# Real data is *messy*

originally looked like this:

```
It was introduced into <a
href="https://en.wikipedia.org/wiki/Mathematics"
title="Mathematics">mathematics</a> in the book <i><a
href="https://en.wikipedia.org/wiki/Disquisitiones_Arithmeticae"
title="Disquisitiones Arithmeticae">Disquisitiones Arithmeticae</a></i>
by <a href="https://en.wikipedia.org/wiki/Carl_Friedrich_Gauss"
title="Carl Friedrich Gauss">Carl Friedrich Gauss</a> in 1801. Ever
since, however, "modulo" has gained many meanings, some exact and some
imprecise.
```



SPARK SUMMIT 2016

[https://en.wikipedia.org/wiki/Modulo\\_\(jargon\)](https://en.wikipedia.org/wiki/Modulo_(jargon))

## ~~Feature extraction~~ Cleaning up data

```
"huuuuuuunnnnnnngrrrryyy",  
"aaaaamaaazinggggg",  
"aaaaamazing",  
"aaaaammm",  
"aaaaammmazzzingggg",  
"aaaaamy",  
"aaaaan",  
"aaaaand",  
"aaaaannnnnnddd",  
"aaaaanyways"
```

*Does it help to clean that up?*

see “Twitter Sentiment Classification using Distant Supervision”, Go et al.

<http://www-cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf>



## Language matters

读书须用意，一字值千金



SPARK SUMMIT 2016

# Lucene to the rescue!



High-performance, full-featured text search library

15 years of experience

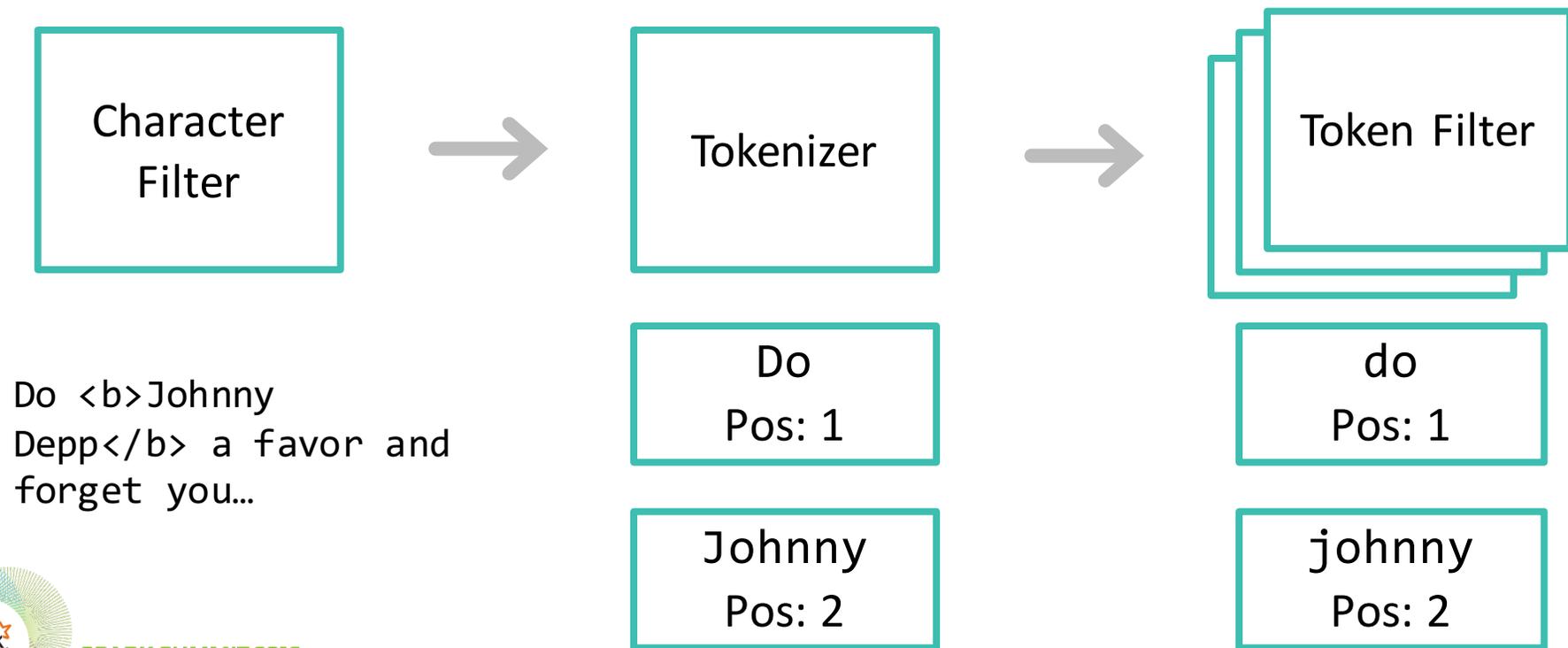
Widely recognized for its utility

- It's a primary test bed for new JVM versions



SPARK SUMMIT 2016

# Text processing



# Lucene for text analysis

state of the art text processing

many extensions available for different languages, use cases,...

however...



```

...
import org.apache.lucene.analysis...
...

Analyzer a = new Analyzer() {
    @Override
    protected TokenStreamComponents createComponents(String fieldName) {
        Tokenizer tokenizer = new StandardTokenizer();
        return new TokenStreamComponents(tokenizer, tokenizer);
    }

    @Override
    protected Reader initReader(String fieldName, Reader reader) {
        return new HTMLStripCharFilter(reader);
    }
};

TokenStream stream = a.tokenStream(null, "<a href=...>some text</a>");
CharTermAttribute term = stream.addAttribute(CharTermAttribute.class);
PositionIncrementAttribute posIncrement = stream.addAttribute(PositionIncrementAttribute.class);
stream.reset();
int pos = 0;
while (stream.incrementToken()) {
    pos += posIncrement.getPositionIncrement();
    System.out.println(term.toString() + " " + pos);
}

```

```

> some 1
> text 2

```

SPARK SUMMIT 2016



```

...
import org.apache.lucene.analysis...
...

Analyzer a = new Analyzer() {
    @Override
    protected TokenStreamComponents createComponents(String fieldName) {
        Tokenizer tokenizer = new StandardTokenizer();
        return new TokenStreamComponents(tokenizer, tokenizer);
    }

    @Override
    protected Reader initReader(String fieldName, Reader reader) {
        return new HTMLStripCharFilter(reader);
    }
};

TokenStream stream = a.tokenStream(null, "<a href=...>some text</a>");
CharTermAttribute term = stream.addAttribute(CharTermAttribute.class);
PositionIncrementAttribute posIncrement = stream.addAttribute(PositionIncrementAttribute.class);
stream.reset();
int pos = 0;
while (stream.incrementToken()) {
    pos += posIncrement.getPositionIncrement();
    System.out.println(term.toString() + " " + pos);
}

```

```

> some 1
> text 2

```

SPARK SUMMIT 2016

## How about a declarative approach?



# LITTLE DEMO



NABUMA RUBBERBAND



SPARK SUMMIT 2016

# Very quick intro to Elasticsearch



**SPARK SUMMIT 2016**  
DATA SCIENCE AND ENGINEERING AT SCALE  
JUNE 6-8, 2016 SAN FRANCISCO

# Elasticsearch in 5 3'



Scalable, real-time search and analytics engine

Data distribution, cluster management

REST APIs

JVM based, uses Apache Lucene internally

Open-source (on Github, Apache 2 License)



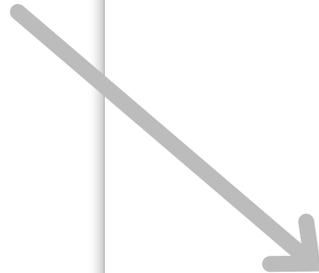
SPARK SUMMIT 2016

# Elasticsearch in 3'



elasticsearch

Unstructured  
search



The screenshot shows the Wikipedia homepage with the following language editions and article counts:

Language	Article Count
English	5 077 000+
Spanish	1 233 000+
German	1 907 000+
French	1 723 000+
Portuguese	909 000+
Polish	1 154 000+
Japanese	1 001 000+
Russian	1 289 000+
Italian	1 252 000+
Chinese	863 000+

The search bar at the bottom contains the text "elasticsearch" and shows a dropdown result for "Elasticsearch" with the description: "A distributed, scalable, and highly available real-time search platform with a RESTful API."



SPARK SUMMIT 2016

# Elasticsearch in 3'



elasticsearch

## Sorting / Scoring

The screenshot shows a Stack Overflow search interface. At the top left is the Stack Overflow logo. On the right, there are tabs for 'Questions' and 'Jobs'. Below the logo is a search bar containing the text 'elasticsearch' and a blue 'search' button. Below the search bar, it says '35,840 results'. To the right of the results count are sorting options: 'relevance', 'newest', 'votes' (which is selected and highlighted with an orange underline), and 'active'. A large grey arrow points from the text 'Sorting / Scoring' on the left towards the 'votes' sorting option. Below the sorting options, there are two search results. The first result has a green box on the left containing '574 votes' and a blue link for the question: 'A: ElasticSearch, Sphinx, Lucene, Solr, Xapian. Which fits for which usage?'. The answer text follows, starting with 'As the creator of **ElasticSearch**, maybe I can give you some reasoning on why I went ahead and created it in the first place :). Using pure Lucene is challenging. There are many things that you need ... This is why I went ahead and created **ElasticSearch**. It has a very advanced distributed model, speaks JSON natively, and exposes many advanced search features, all seamlessly expressed through JSON DSL ...'. The answer is attributed to 'answered Feb 18 '10 by kimchy'. The second result has a green box on the left containing '530 votes' and a blue link for the question: 'A: Shards and replicas in Elasticsearch'. The answer text follows, starting with 'I'll try to explain with a real example, since the answer and replies you got don't seem to help you. When you download **elasticsearch** and start it up you create an **elasticsearch** node which tries ... it will have the default number of shards: 5 primaries. What does it mean? It means that **elasticsearch** will create 5 primary shards that will contain your data ...'. The answer is attributed to 'answered Mar 29 '13 by javanna'.



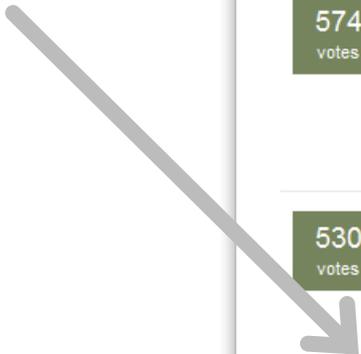
SPARK SUMMIT 2016

# Elasticsearch in 3'



elasticsearch

Pagination



The screenshot shows a Stack Overflow search results page. At the top left is the Stack Overflow logo. To the right are tabs for 'Questions' and 'Jobs'. Below the logo is a search bar containing the text 'elasticsearch' and a blue 'search' button. Underneath the search bar, it says '35,840 results' followed by sorting options: 'relevance', 'newest', 'votes' (which is selected), and 'active'. The first result is an answer with 574 votes, titled 'A: ElasticSearch, Sphinx, Lucene, Solr, Xapian. Which fits for which usage?'. The second result is an answer with 530 votes, titled 'A: Shards and replicas in Elasticsearch'. The text of the second answer is partially visible, starting with 'I'll try to explain with a real example, since the answer and replies you got don't seem to help you. When you download elasticsearch and start it up you create an elasticsearch node which tries ... it will have the default number of shards: 5 primaries. What does it mean? It means that elasticsearch will create 5 primary shards that will contain your data ...'.



SPARK SU

The pagination controls are located at the bottom of the page. It features a row of buttons: '1' (highlighted in orange), '2', '3', '4', '5', an ellipsis '...', '2390', and 'next'. To the right of these buttons are three more buttons: '15' (highlighted in orange), '30', and '50', followed by the text 'per page'.

# Elasticsearch in 3'



Enrichment



Search

**Repositories** 8,033

**Code** 798,062

**Issues** 57,377

**Answers** 16

**Languages**

Java	1,349
JavaScript	1,015
Shell	828
Python	804
Ruby	783
PHP	405
Go	187
C#	174
Scala	165
HTML	99

**We've found 8,033 repository results**

**elastic/elasticsearch**  
Open Source, Distributed, RESTful Search Engine  
Updated 8 hours ago

**dockerfile/elasticsearch**  
**Elastic Search** Dockerfile for trusted automated Docker builds.  
Updated on Jan 8

**mesos/elasticsearch**  
**Elasticsearch** on Mesos  
Updated 5 days ago



SPARK SUMMIT 2016

# Elasticsearch in 3'



Structured search



Search

We've found 8,033 repository results

Category	Count
Repositories	8,033
Code	798,062
Issues	57,377
Users	16

**Languages**

Java	1,349
JavaScript	1,015
Shell	828
Python	804
Ruby	783
PHP	405
Go	187
C#	174
Scala	165
HTML	99

**elastic/elasticsearch**  
Open Source, Distributed, RESTful Search Engine  
Updated 8 hours ago

**dockerfile/elasticsearch**  
**Elastic Search** Dockerfile for trusted automated Docker builds.  
Updated on Jan 8

**mesos/elasticsearch**  
**Elasticsearch** on Mesos  
Updated 5 days ago

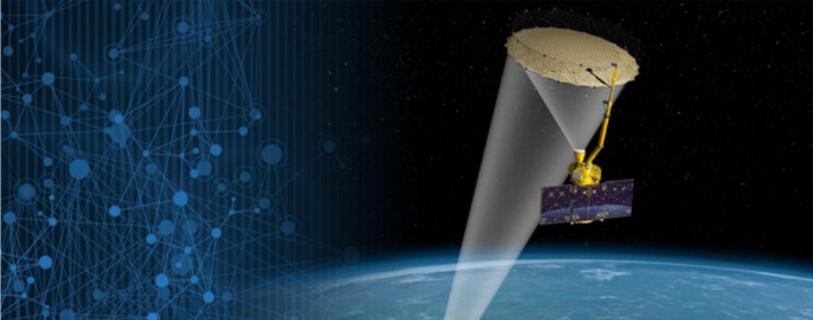
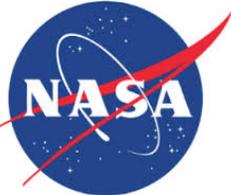


SPARK SUMMIT 2016

# Elasticsearch in 3'



elasticsearch



`_search?q=life:universe`



The slide features a central white rectangular area. On the left side of this area is the NASA logo. To its right is a photograph of a satellite in space, with a blue network of nodes and lines overlaid on the background. Below the satellite image is the Elasticsearch search query: `_search?q=life:universe`. At the bottom of the white area is a photograph of a Mars rover on the red planet's surface, with yellow network lines overlaid on the left side.



SPARK SUMMIT 2016

<https://www.elastic.co/elasticon/2015/sf/unlocking-interplanetary-datasets-with-real-time-search>

# Machine Learning and Elasticsearch



**SPARK SUMMIT 2016**  
DATA SCIENCE AND ENGINEERING AT SCALE  
JUNE 6-8, 2016 SAN FRANCISCO

# Machine Learning and Elasticsearch



SPARK SUMMIT 2016

# Machine Learning and Elasticsearch

Term Analysis (tf, idf, bm25)

Graph Analysis

Co-occurrence of Terms (significant terms)

- ChiSquare

Pearson correlation (#16817)

Regression (#17154)

What about classification/clustering/ etc... ?



SPARK SUMMIT 2016

**It's not the matching data,  
but the meta that lead to it**



---

# How to use Elasticsearch from Spark ?

---

*Somebody on Stackoverflow*

# Elasticsearch for Apache Hadoop™

## elasticsearch-hadoop

Java ★ 670 📌 362

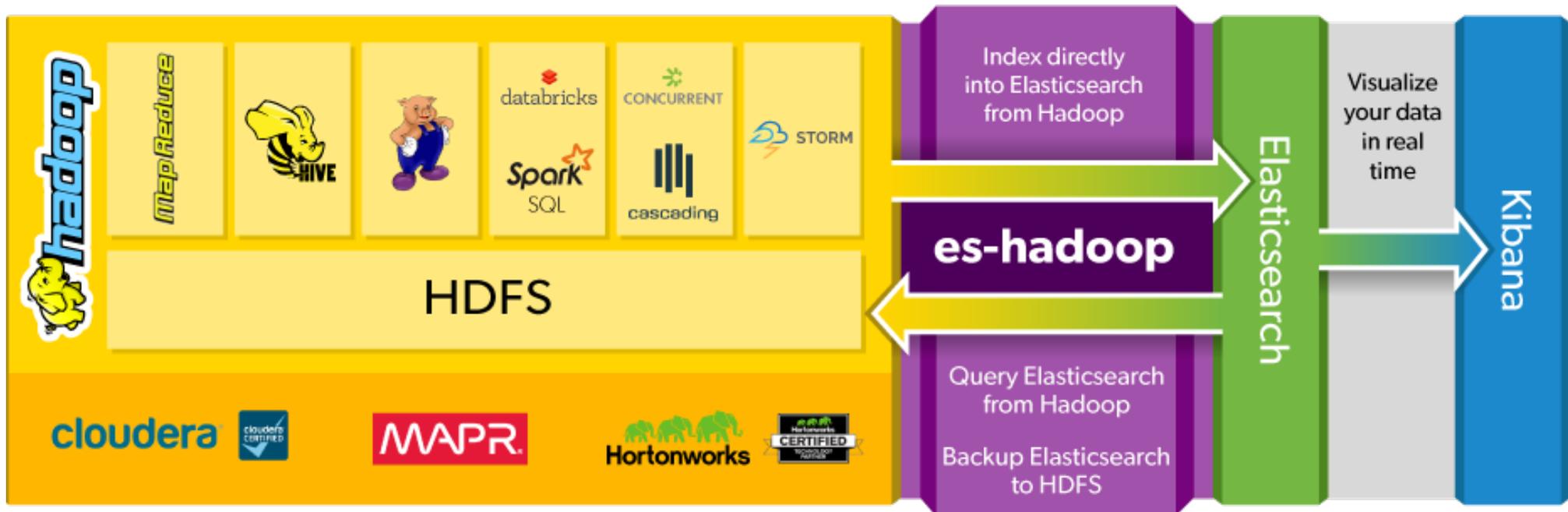
Elasticsearch real-time search and analytics natively integrated with Hadoop

Updated 3 hours ago



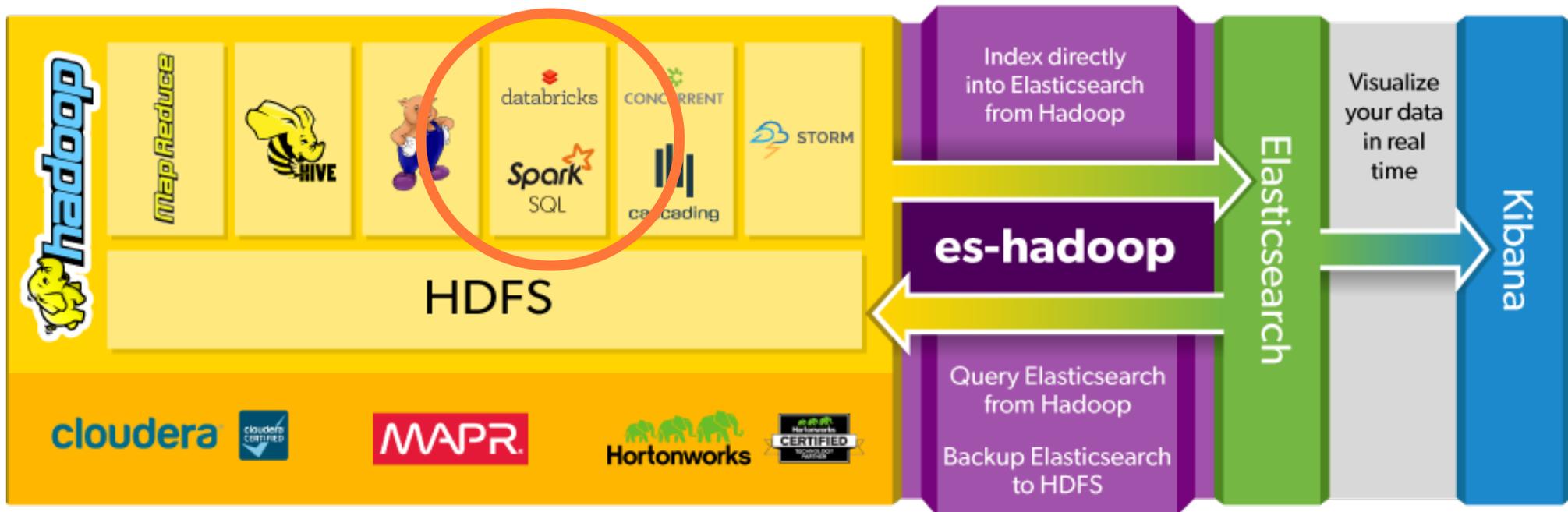
SPARK SUMMIT 2016

# Elasticsearch for Apache Hadoop™



SPARK SUMMIT 2016

# Elasticsearch for Apache Hadoop™



SPARK SUMMIT 2016

# Elasticsearch Spark – Native integration



Scala & Java API

Understands Scala & Java types

- Case classes
- Java Beans

Available as Spark package

Supports Spark Core & SQL

all 1.x version (1.0-1.6)

Available for Scala 2.10 and 2.11

SPARK SUMMIT 2016



## elasticsearch-hadoop [\(homepage\)](#)

Official integration between Apache Spark and Elasticsearch real-time search and analytics

@elastic / ★★★★★ (3)

Native Java/Scala API for Elasticsearch in Spark. Read/write RDDs and DataFrames from/to Elasticsearch.

Reference documentation available at <https://www.elastic.co/guide/en/elasticsearch/hadoop/current/spark.html>

Note that artifacts for Scala 2.11 are also available - simply use the `_2.11` suffix instead in the artifact id.

### Your rating

★★★★★ Very Good

### Tags

1 sql +- 1 elasticsearch +- 1 search +- 1 analytics +- 1 realtime +- 1 core +- 1 data source +-

### How to [+]

Include this package in your Spark Applications using:

spark-shell, pyspark, or spark-submit

```
> $SPARK_HOME/bin/spark-shell --packages org.elasticsearch:elasticsearch-spark_2.10:5.0.0-alpha3
```

### Releases

Version: **5.0.0-alpha3** ( [5e5a7d](#) | [zip](#) | [jar](#) ) / Date: 2016-06-01 / License: [Apache-2.0](#) / Scala version: 2.10

Version: **2.3.2** ( [cae8b7](#) | [zip](#) | [jar](#) ) / Date: 2016-05-22 / License: [Apache-2.0](#) / Scala version: 2.10

# Elasticsearch as RDD / Dataset\*

```
import org.elasticsearch.spark._

val sc = new SparkContext(new SparkConf())
val rdd = sc.esRDD("buckethead/albums", "?q=piques")
```

```
import org.elasticsearch.spark._

case class Artist(name: String, albums: Int)

val u2 = Artist("U2", 13)
val bh = Map("name" -> "Buckethead", "albums" -> 255, "age" -> 46)

sc.makeRDD(Seq(u2, bh)).saveToEs("radio/artists")
```



# Elasticsearch as a DataFrame

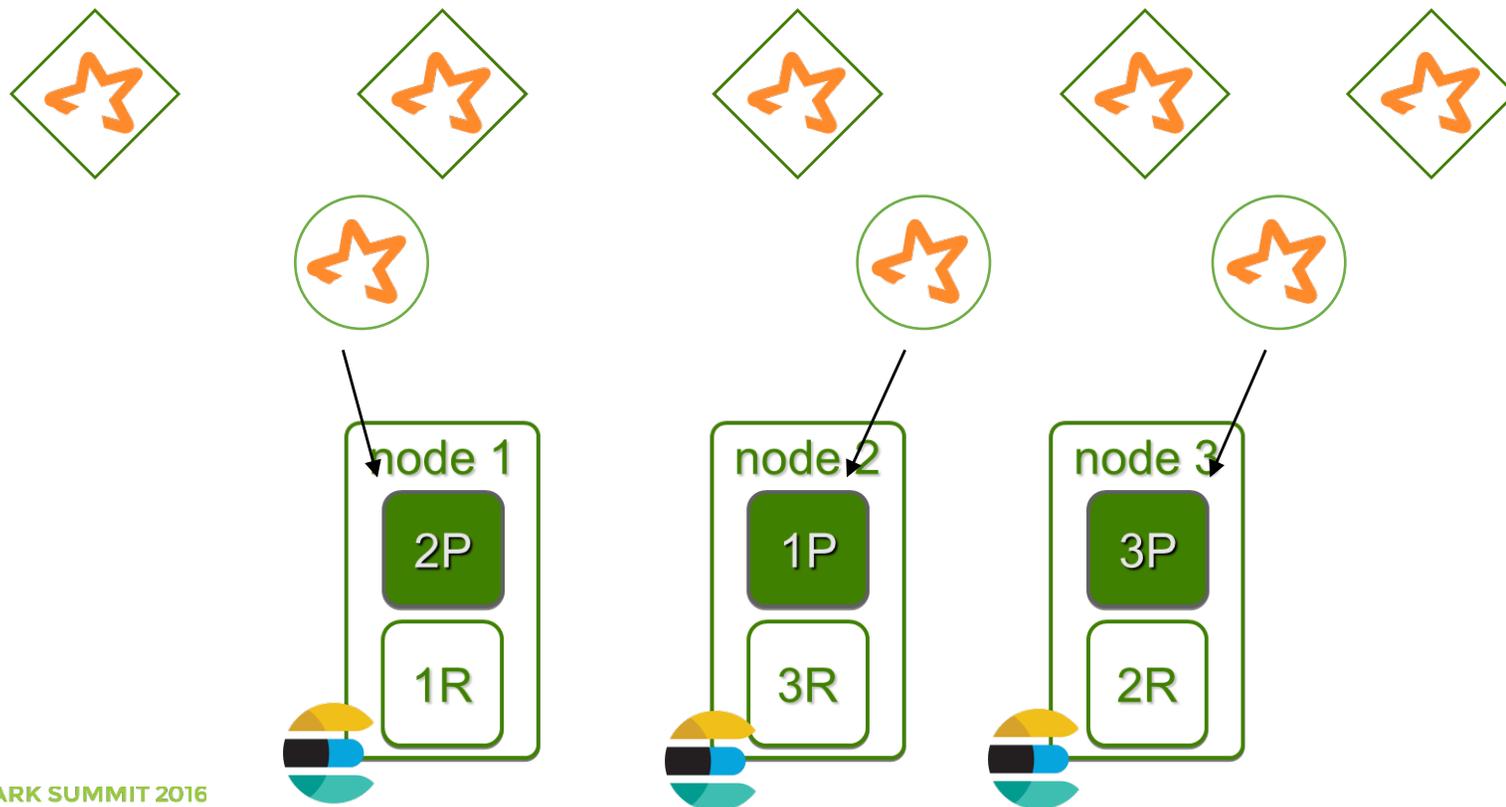
```
val df = sql.read.format("es").load("buckethead/albums")  
df.filter(df("category").equalTo("pikes").and(df("year").geq(2015)))
```



```
{ "query" :  
  { "bool" : { "must" : [  
    "match" : { "category" : "pikes" }  
  ],  
    "filter" : [  
      { "range" : { "year" : { "gte" : "2015" }}}  
    ]  
  }  
}}
```



# Partition to Partition Architecture

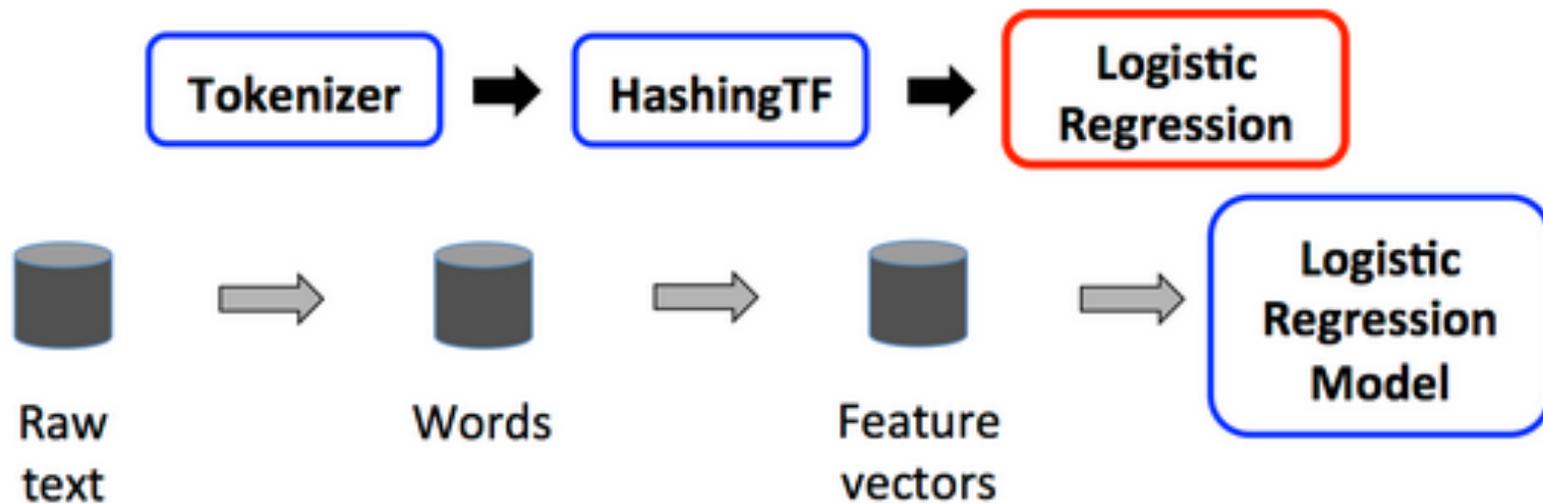


# Putting the pieces together

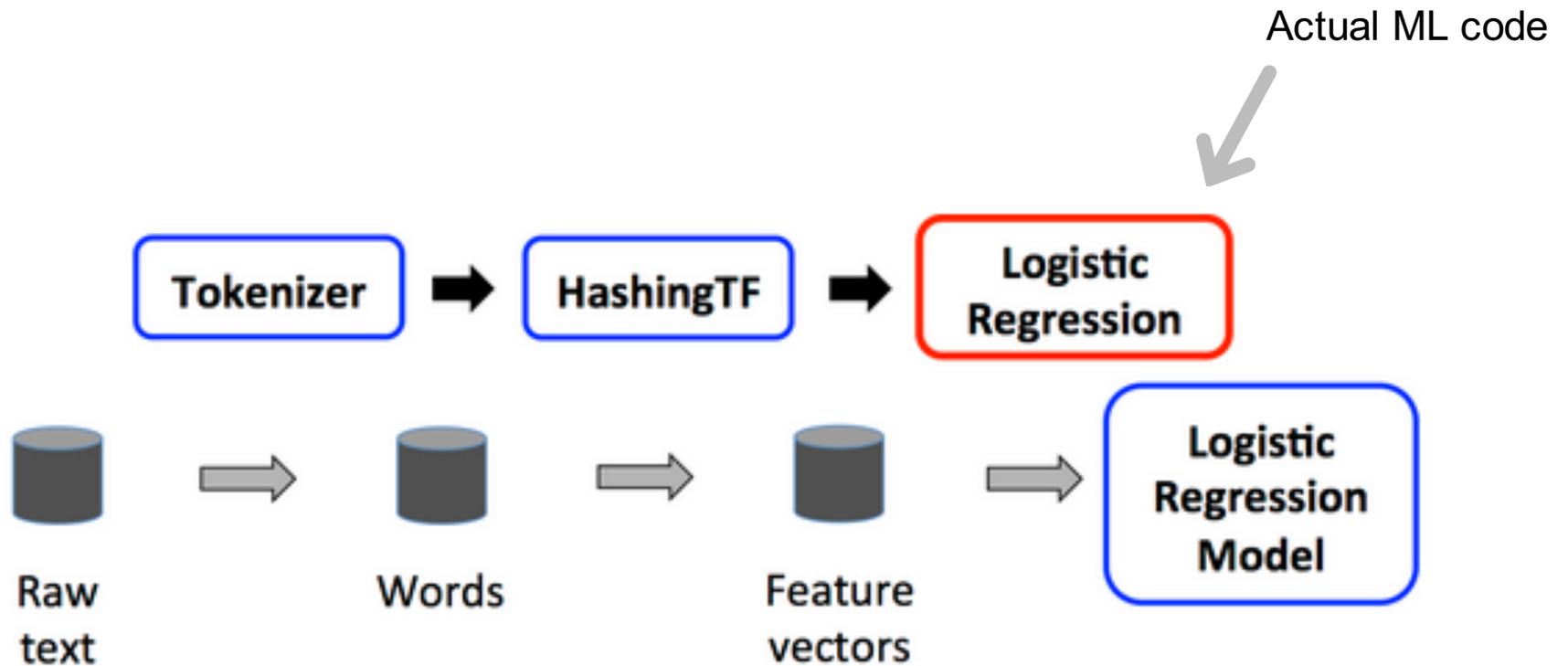


**SPARK SUMMIT 2016**  
DATA SCIENCE AND ENGINEERING AT SCALE  
JUNE 6-8, 2016 SAN FRANCISCO

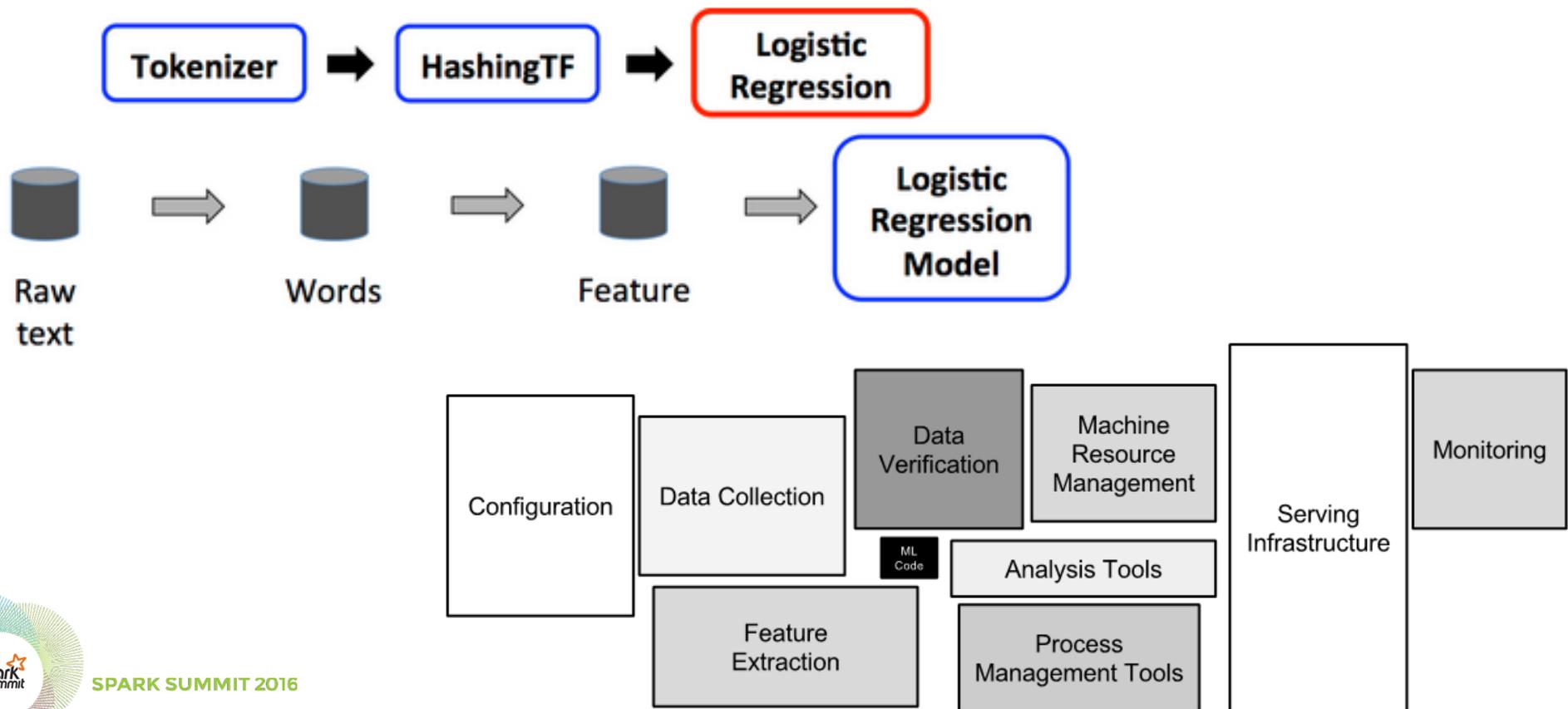
# Typical ML pipeline for text



# Typical ML pipeline for text



# Typical ML pipeline for text



# Pure Spark MLlib

```
val training = movieReviewsDataTrainingData

val tokenizer = new Tokenizer()
    .setInputCol("text")
    .setOutputCol("words")
val hashingTF = new HashingTF()
    .setNumFeatures(1000)
    .setInputCol(tokenizer.getOutputCol)
    .setOutputCol("features")
val lr = new LogisticRegression()
    .setMaxIter(10)
    .setRegParam(0.001)
val pipeline = new Pipeline()
    .setStages(Array(tokenizer, hashingTF, lr))

val model = pipeline.fit(training)
```



# Pure Spark MLlib

```
val tokenizer = new Tokenizer()  
    .setInputCol("text")  
    .setOutputCol("words")  
val hashingTF = new HashingTF()  
    .setNumFeatures(1000)  
    .setInputCol(tokenizer.getOutputCol)  
    .setOutputCol("features")  
val lr = new LogisticRegression()  
    .setMaxIter(10)  
    .setRegParam(0.001)
```



# Pure Spark MLlib

```
val tokenizer = new Tokenizer()  
    .setInputCol("text")  
    .setOutputCol("words")  
val hashingTF = new HashingTF()  
    .setNumFeatures(1000)  
    .setInputCol(tokenizer.getOutputCol)  
    .setOutputCol("features")  
val lr = new LogisticRegression()  
    .setMaxIter(10)  
    .setRegParam(0.001)
```



# Pure Spark MLlib

```
val analyzer = new ESAnalyzer()  
    .setInputCol("text")  
    .setOutputCol("words")  
val hashingTF = new HashingTF()  
    .setNumFeatures(1000)  
    .setInputCol(tokenizer.getOutputCol)  
    .setOutputCol("features")  
val lr = new LogisticRegression()  
    .setMaxIter(10)  
    .setRegParam(0.001)
```

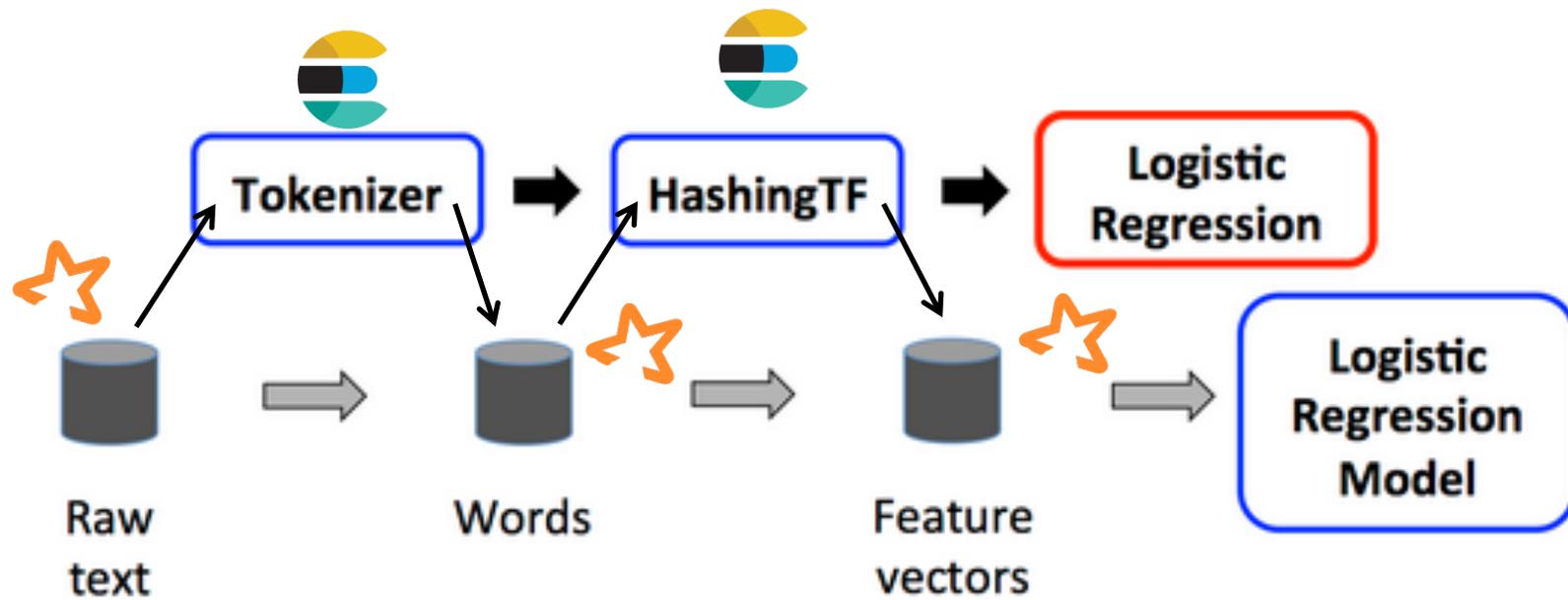


# Pure Spark MLlib

```
val analyzer = new ESAnalyzer()  
    .setInputCol("text")  
    .setOutputCol("words")  
val hashingTF = new HashingTF()  
    .setNumFeatures(1000)  
    .setInputCol(tokenizer.getOutputCol)  
    .setOutputCol("features")  
val lr = new LogisticRegression()  
    .setMaxIter(10)  
    .setRegParam(0.001)
```



# Data movement



## Work once – reuse multiple times

```
// index / analyze the data  
training.saveToEs("movies/reviews")
```



## Work once – reuse multiple times

```
// prepare the spec for vectorize – fast and lightweight

val spec = s""""{ "features" : [{
  | "field": "text",
  | "type" : "string",
  | "tokens" : "all_terms",
  | "number" : "occurrence",
  | "min_doc_freq" : 2000
  | }],
  | "sparse" : "true"}""".stripMargin

ML.prepareSpec(spec, "my-spec")
```



## Access the vector directly

```
// get the features - just another query

val payload = s"""{"script_fields" : { "vector" :
  | { "script" : { "id" : "my-spec", "lang" : "doc_to_vector" } }
  | }}""".stripMargin

// index the data
vectorRDD = sparkCtx.esRDD("ml/data", payload)

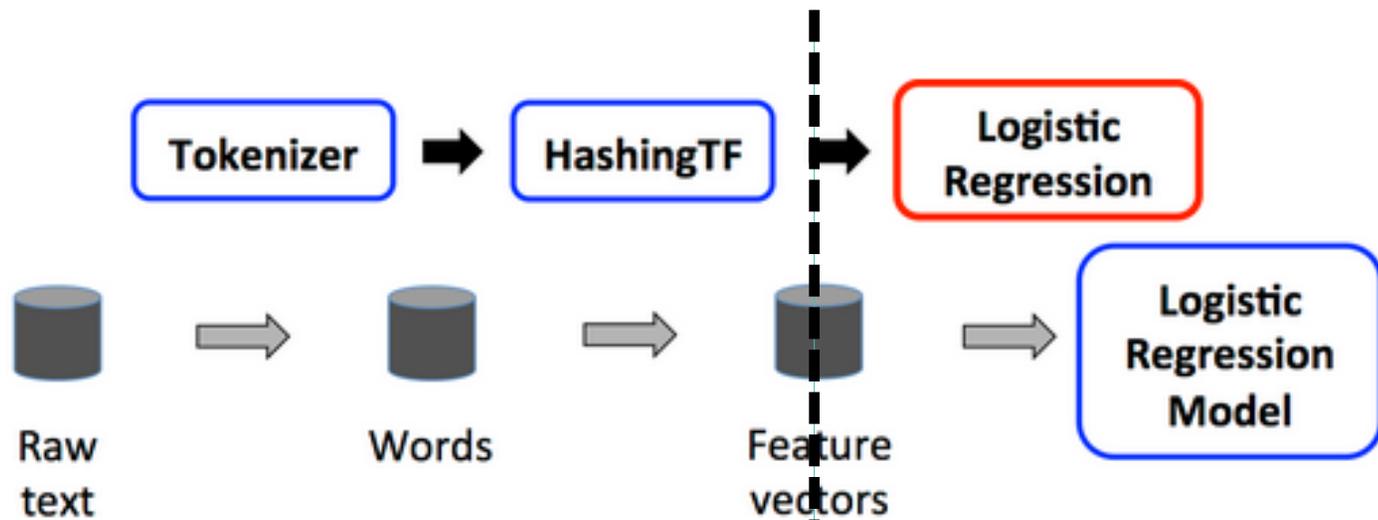
// feed the vector to the pipeline
val vectorized = vectorRDD.map ( x =>
  // get indices, the vector and length
  (if (x._1 == "negative") 0.0d else 1.0d, ML.getVectorFrom(x._2))
).toDF("label", "features")
```

## Revised ML pipeline

```
val vectorized = vectorRDD.map...  
  
val lr = new LogisticRegression()  
    .setMaxIter(10)  
    .setRegParam(0.001)  
  
val model = lr.fit(vectorized)
```



# Simplify ML pipeline



Once per dataset,  
regardless of # of  
pipelines



Raw data is not  
required any more



## Need to adjust the model? Change the spec

```
val spec = s""${ "features" : [{  
  | "field": "text",  
  | "type" : "string",  
  | "tokens" : "given",  
  | "number" : "tf",  
  | "terms": ["term1", "term2", ...]  
  | }],  
  | "sparse" : "true"}""}.stripMargin
```

```
ML.prepareSpec(spec)
```

# LITTLE DEMO



NABUMA RUBBERBAND



SPARK SUMMIT 2016

# All this is WIP

Not all features available (currently dictionary, vectors)

Works with data outside or inside Elasticsearch (latter is **much** faster)

Bind vectors to queries

Other topics WIP:

Focused on document / text classification – numeric support is next

Model importing / exporting – Spark 2.0 ML persistence

Feedback highly sought - Is this useful?



SPARK SUMMIT 2016

# THANK YOU.

[j.mp/spark-summit-west-16](http://j.mp/spark-summit-west-16)  
[elastic.co/hadoop](http://elastic.co/hadoop)  
[github.com/elastic](https://github.com/elastic) | [costin](#) | [brwe](#)  
[discuss.elastic.co](http://discuss.elastic.co)  
[@costinl](#)



**SPARK SUMMIT 2016**  
DATA SCIENCE AND ENGINEERING AT SCALE  
JUNE 6-8, 2016 SAN FRANCISCO