勉強会プレゼン

Presents by Yoshino

2021/07/17

目次

Elasticsearch

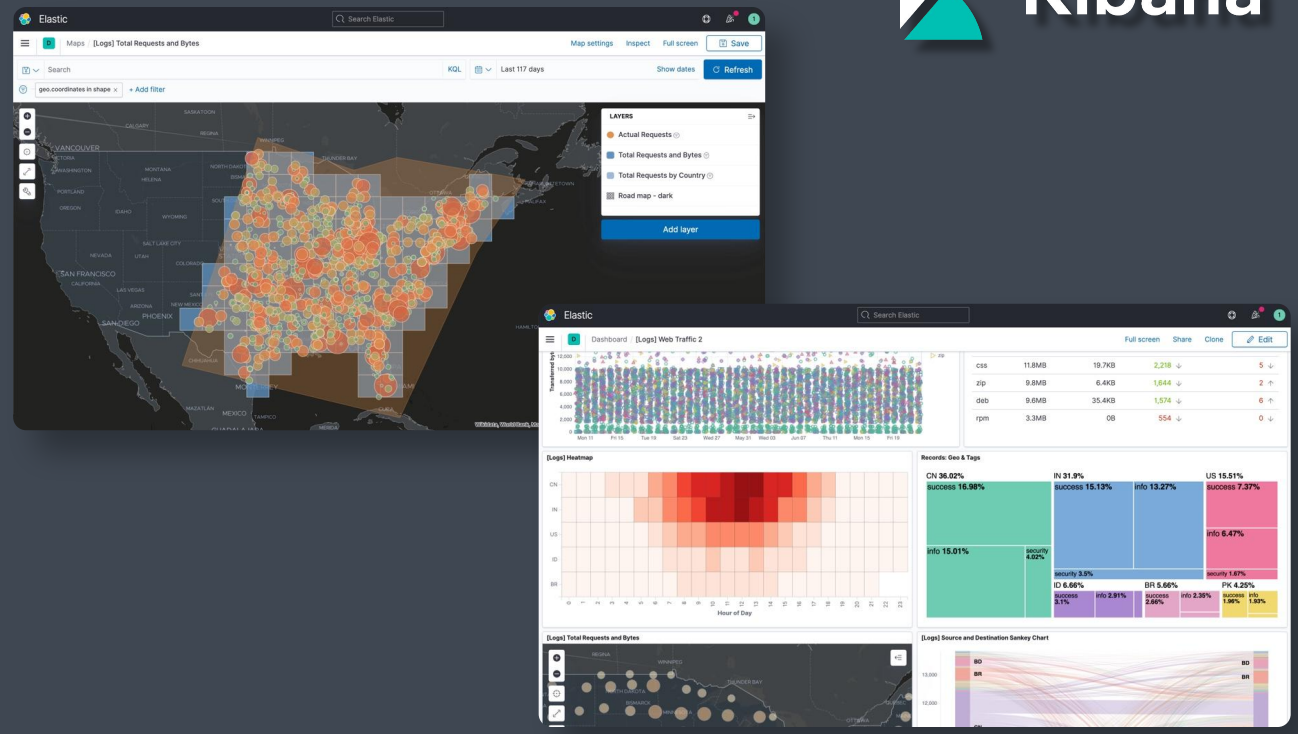
Elasticsearch

> 何ぞや？

- NoSQL
- 高速な検索エンジン
- OSS
- 後から何個も連結可能
- 障害に強い
- 可視化が簡単

Elasticsearch

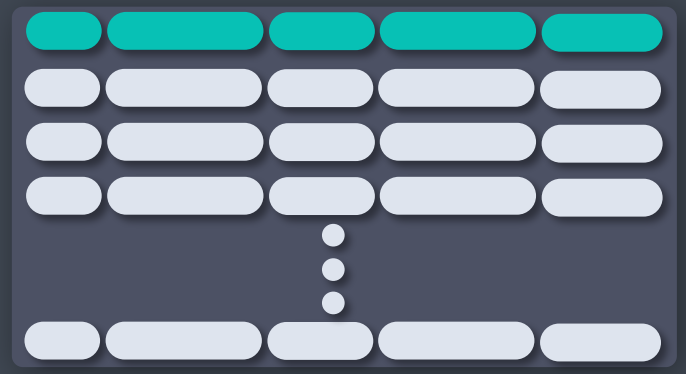
Kibana



Elasticsearch

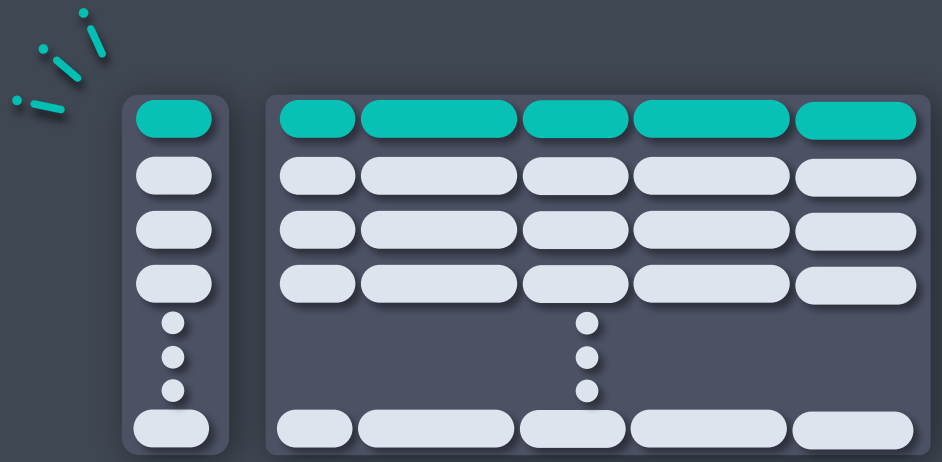
> 高速検索の仕組み

通常: LIKE検索



O(n)

Elasticsearch: インデックス検索



O(1)

Elasticsearch

> 高速検索の仕組み

Elasticsearch

> 高速検索の仕組み

例: 私はご飯を食べた。

Elasticsearch

> 高速検索の仕組み

例: 私はご飯を食べた。

Char Filtering

Tokenizing

Filtering

私ご飯食べた

Elasticsearch

> 高速検索の仕組み

例: 私はご飯を食べた。

私ご飯食べた

私/ご飯/食べた

Elasticsearch

> 高速検索の仕組み

例: 私はご飯を食べた。

Analyzing

Analyzing

> 自然言語処理 ~難易度~

英語の場合

例: I have a pen.

I/have/a/pen

日本語の場合

例: 私はペンを持っている。

私はペンを持っている

私 / ペン　持っている

> 単語の区切りが判別しにくい(日本語、中国語、韓国語)

Analyzing

> 自然言語処理 ~手法~

形態素解析

- 辞書を用いて分割
- 品詞分解

例: Mecab, kuromoji

- 辞書が必要
- 結果が辞書依存

N-gram

- N文字毎に分解
- 辞書がいらない
- 高速

- インデックスが大きくなる
- ノイズが入りやすい

Analyzing

> 自然言語処理 ~手法~

形態素解析

例: 鬼滅の刃

鬼/滅/の/刃

名詞/名詞/接続詞/名詞

× 「鬼滅」がヒットしない

N-gram

Uni-gram: [鬼、滅、の、刃]、

Bi-gram: [鬼滅、滅の、の刃]、

Tri-gram: [鬼滅の、滅の刃]、

Four-gram: [鬼滅の刃]

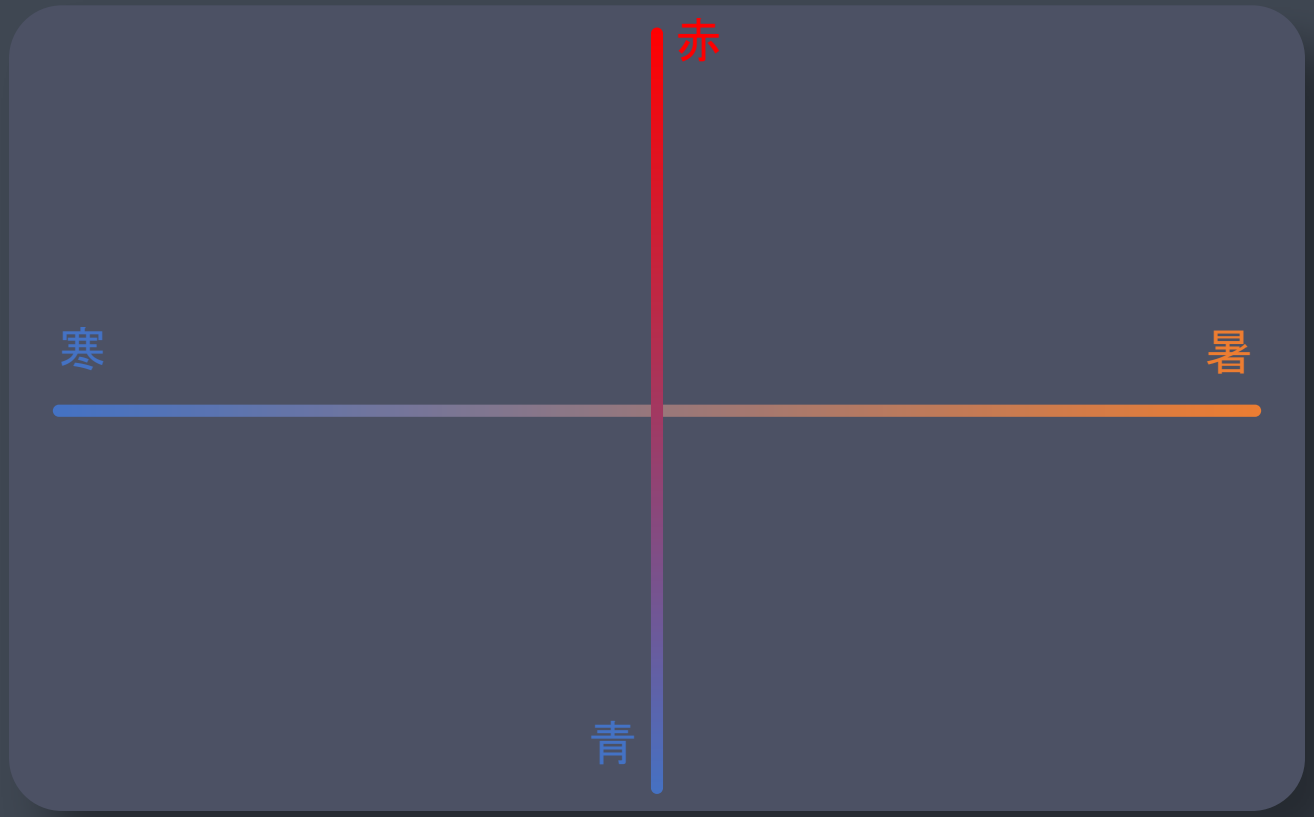
× 生成情報が大きい

Word-embedding

Word-embedding (単語埋め込み)

> なんやそれ？

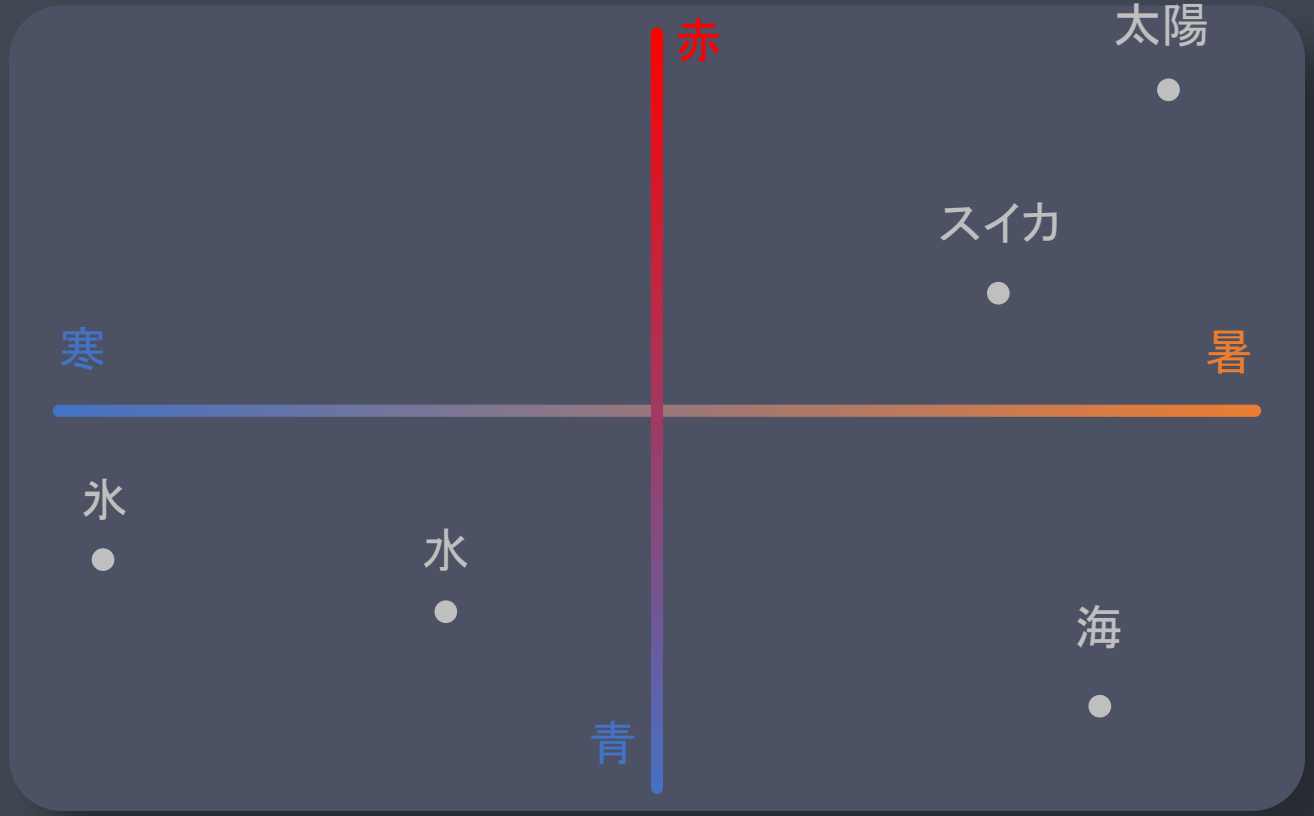
- 単語をベクトル化する
- 深層学習を用いる



Word-embedding (単語埋め込み)

> なんやそれ？

- 単語をベクトル化する
- 深層学習を用いる



→ ベクトル演算が可能になる！

Word-embedding (単語埋め込み)

> 手法

シソーラス(Thesaurus)

- 単語を関係で表現
- 人力

例:車

[自動、自動車、乗用車]

[移動手段、バス、人]

カウントベース

- 単語の出現回数を使用
- 周りの単語で予測する
- 行列で管理

例:お金降ってこないかな

「降る」↓

[お金:1,来る:1,かな:1]

推論ベース

- 深層学習で単語を推測
- 周りの単語で予測する
- 再学習が早い

例:お金_こないかな

降る、落ちる

Word2vec,fastText

Word-embedding (単語埋め込み)

> 推論ベース

CBOW

(Continuous Bag of Words)

- 周辺から単語を予測

例:お金_こないかな

降る、落ちる

早い

Skip-gram

- 単語から周辺の単語を予測

例:_降って_かな

(雨、くる)

(お金、くる)

高精度

まとめ

- Elasticsearch
  > ええで。早い。NoSQL。

- 自然言語処理
  > 日本語難しい(´･ω･`)

- 単語埋め込み
  > ベクトル演算面白そう。色んな手法あり。

Fin!