elastic

Search
a new era

David Pilato | @dadoonet

BNP PARIBAS FORTIS

Elasticsearch

You Know, for Search

elastic

Elasticsearch

Lucene

elastic

“

These are not the droids you are looking for.

```
GET /_analyze
{
  "char_filter": [ "html_strip" ],
  "tokenizer": "standard",
  "filter": [ "lowercase", "stop", "snowball" ],
  "text": "These are <em>not</em> the droids
           you are looking for."
}
```

```
"char_filter": "html_strip"
```

```
These are <em>not</em> the droids you are
   looking for.
```

```
These are not the droids you are
   looking for.
```

```
"tokenizer": "standard"
```

```
These are not the droids you are looking for.
```

```
These
are
not
the
droids
you
are
looking
for
```

elastic

```
"filter": "lowercase"
```

```
These         these
are           are
not           not
the           the
droids   →    droids
you           you
are           are
looking       looking
for           for
```

```
"filter": "stop"
```

```
These           these           
are             are             
not             not             
the             the             
droids    →     droids    →     droids
you             you             you
are             are             
looking         looking         looking
for             for             
```

```
"filter": "snowball"
```

```
These      these
are        are
not        not
the        the
droids  →  droids  →  droids  →  droid
you        you        you        you
are        are
looking    looking    looking    look
for        for
```

These are <em>not</em> the **droids** **you** are **looking** for.

```
{ "tokens": [{
      "token": "droid",
      "start_offset": 27, "end_offset": 33,
      "type": "<ALPHANUM>", "position": 4
    },{
      "token": "you",
      "start_offset": 34, "end_offset": 37,
      "type": "<ALPHANUM>", "position": 5
    }, {
      "token": "look",
      "start_offset": 42, "end_offset": 49,
      "type": "<ALPHANUM>", "position": 7
    }]}
```

Semantic

search

≠

Literal

matches

Elasticsearch

You Know, for Search

Elasticsearch

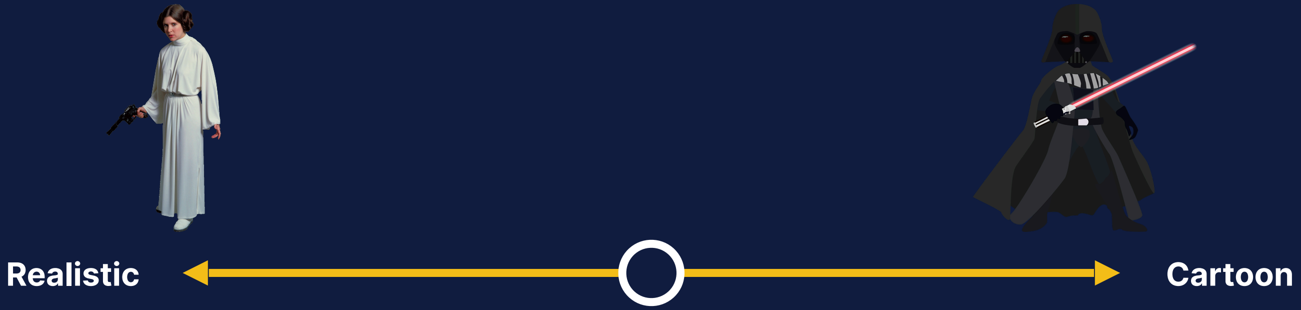
You Know, for Vector Search

elastic

What is a **Vector**?

Embeddings represent your data

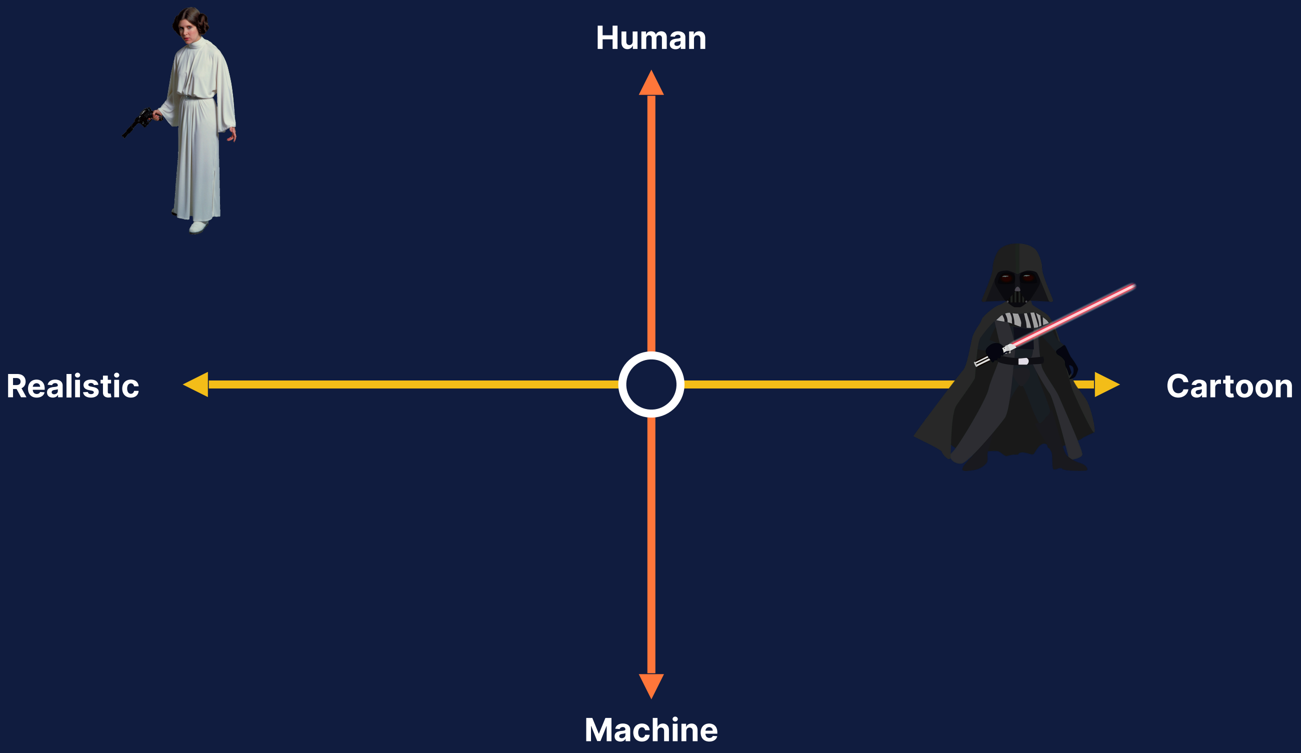
Example: 1-dimensional vector



Character	Vector
	[-1]
	[1]

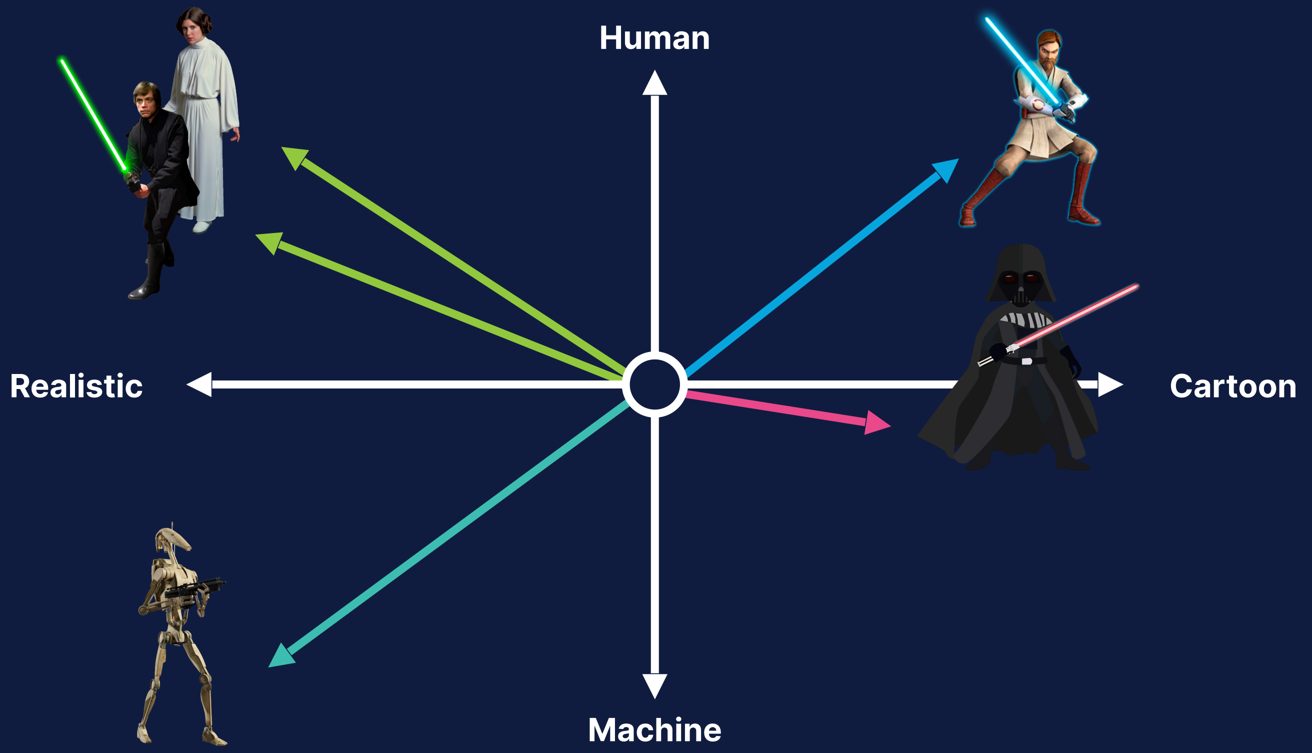
Multiple dimensions

represent different data aspects



Character	Vector
	[-1, 1]
	[1, 0]

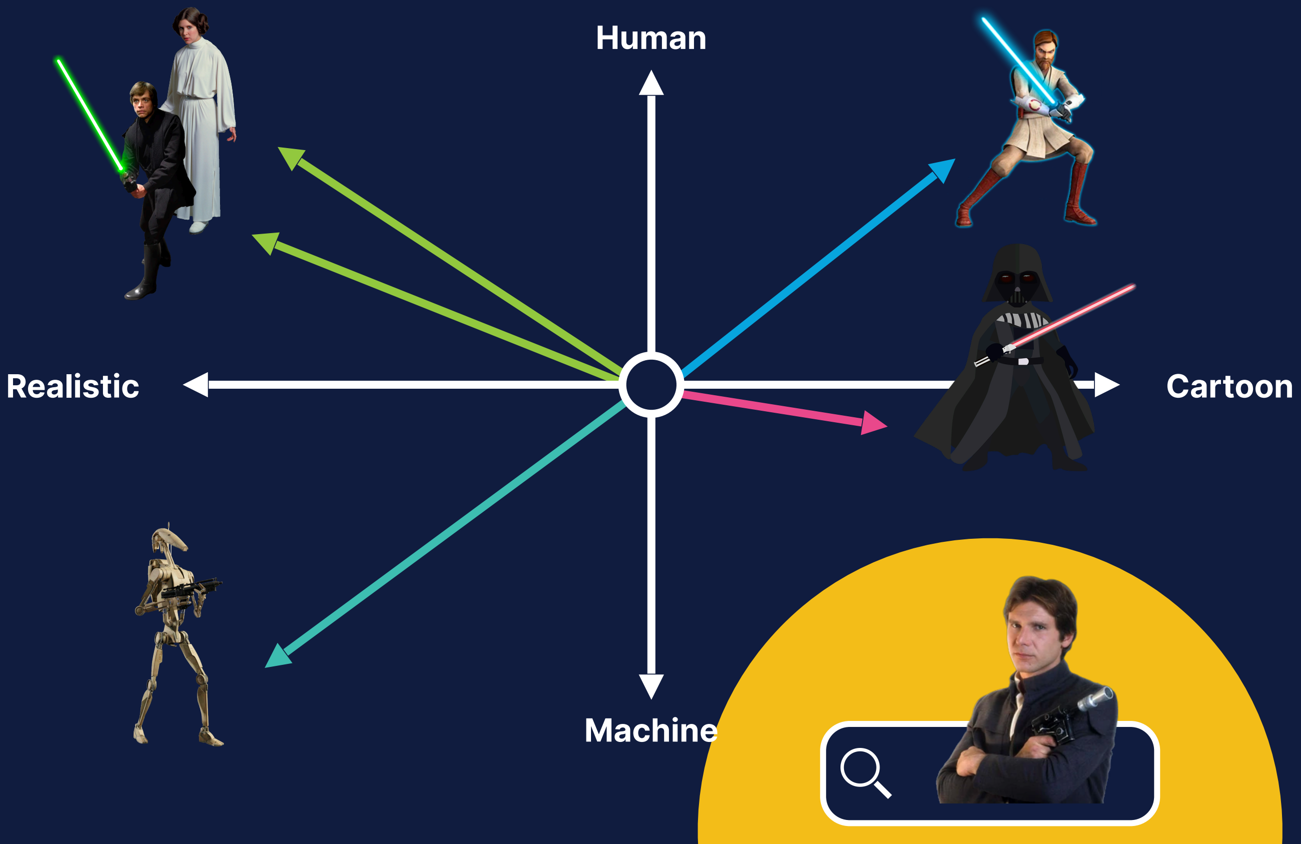
Similar data
is grouped together



Character	Vector
	[-1.0, 1.0]
	[1.0, 0.0]
	[-1.0, 0.8]
	[1.0, 1.0]
	[-1.0, -1.0]

elastic

Vector search ranks objects
by similarity (~relevance) to the query

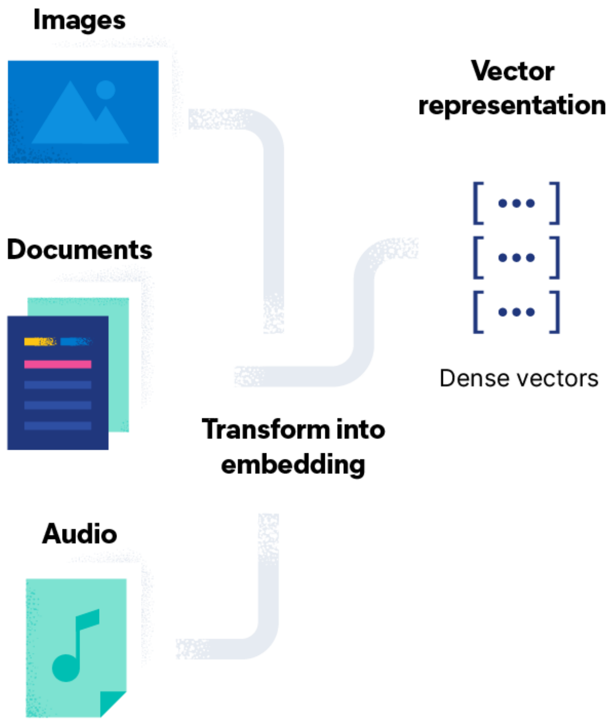


Rank	Result
Query	
1	
2	
3	
4	
5	

How do you

index vectors?

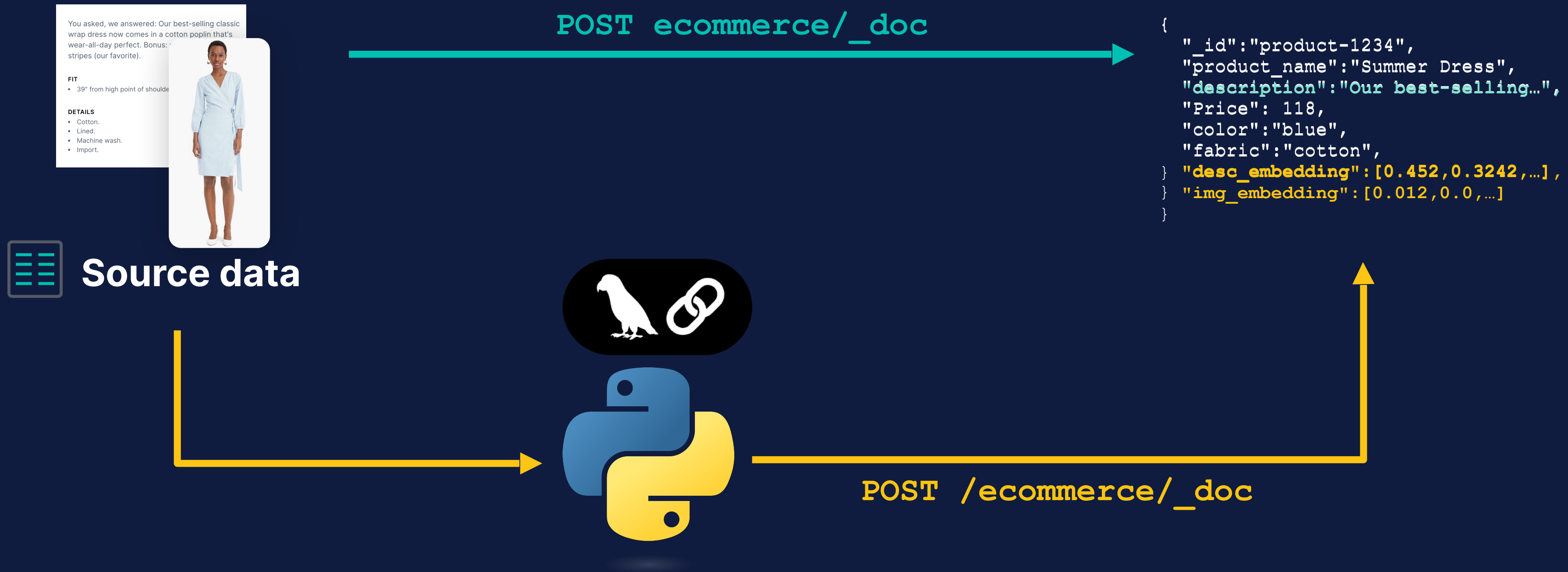
Architecture of Vector Search



`dense_vector` field type

```
PUT ecommerce
{
  "mappings": {
    "properties": {
      "description": {
        "type": "text"
      }
      "desc_embedding": {
        "type": "dense_vector"
      }
    }
  }
}
```

Data Ingestion and Embedding Generation



You asked, we answered: Our best-selling classic wrap dress now comes in a cotton poplin that's wear-all-day perfect. Bonus: stripes (our favorite).

FIT

- 39" from high point of shoulde

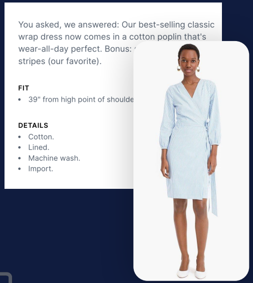
DETAILS

- Cotton.
- Lined.
- Machine wash.
- Import.

```
{
  "_id":"product-1234",
  "product_name":"Summer Dress",
  "description":"Our best-selling…",
  "Price": 118,
  "color":"blue",
  "fabric":"cotton",
} "desc_embedding":[0.452,0.3242,…],
} "img_embedding":[0.012,0.0,…]
}
```

elastic

With Elastic ML



You asked, we answered: Our best-selling classic wrap dress now comes in a cotton poplin that's wear-all-day perfect. Bonus: stripes (our favorite).

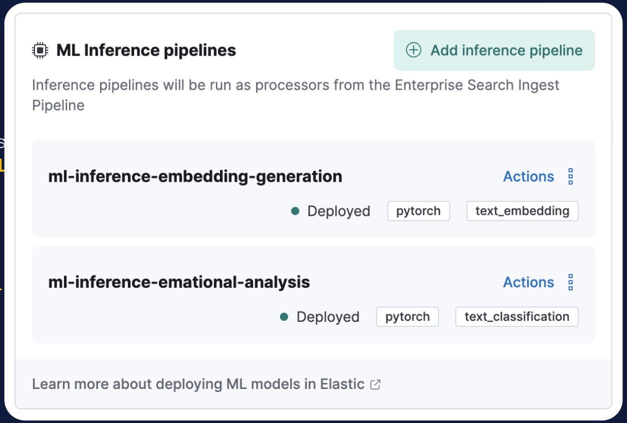
FIT
- 39" from high point of shoulde

DETAILS
- Cotton.
- Lined.
- Machine wash.
- Import.

Source data

```
{
  "_id":"product-1234",
  "product_name":"Summer Dres
  "description":"Our best-sel
  "Price": 118,
  "color":"blue",
  "fabric":"cotton",
}
```

POST /ecommerce/_doc



ML Inference pipelines

Add inference pipeline

Inference pipelines will be run as processors from the Enterprise Search Ingest Pipeline

ml-inference-embedding-generation Actions
● Deployed pytorch text_embedding

ml-inference-emational-analysis Actions
● Deployed pytorch text_classification

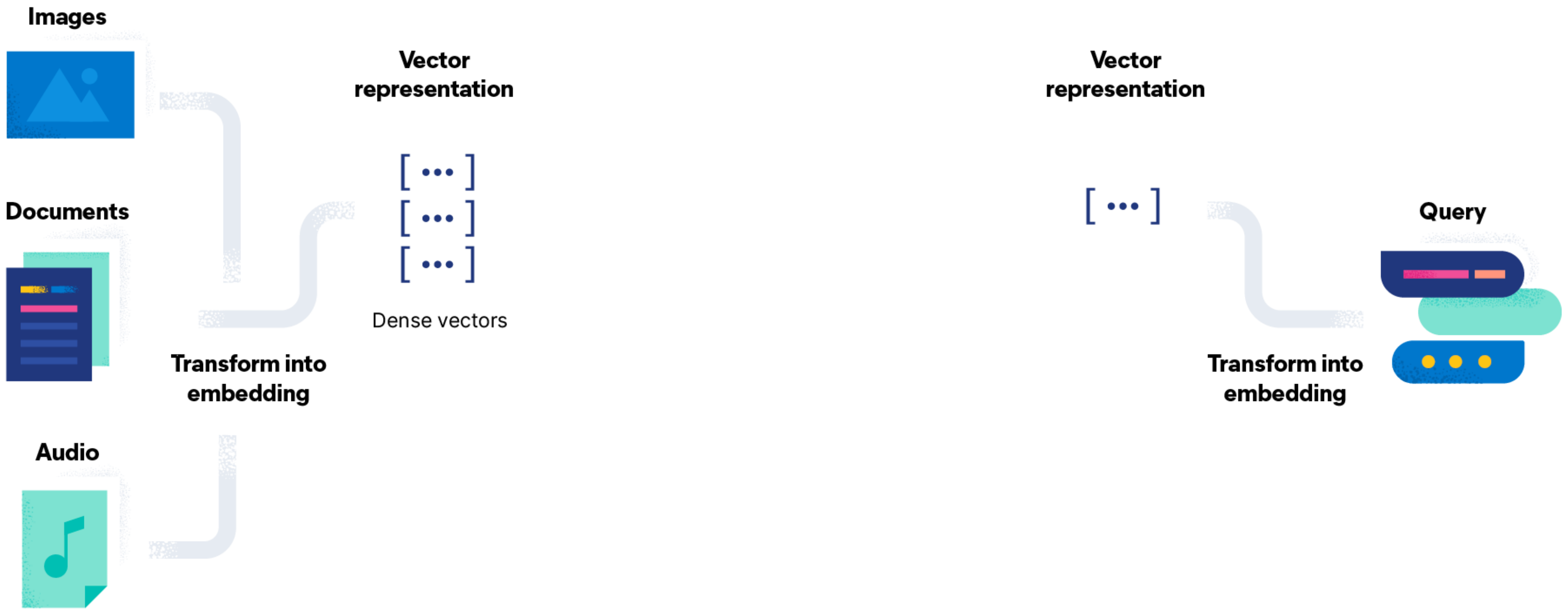
Learn more about deploying ML models in Elastic

```
{
  "_id":"product-1234",
  "product_name":"Summer Dress",
  "description":"Our best-selling…",
  "Price": 118,
  "color":"blue",
  "fabric":"cotton",
  "desc_embedding":[0.452,0.3242,…]
}
```

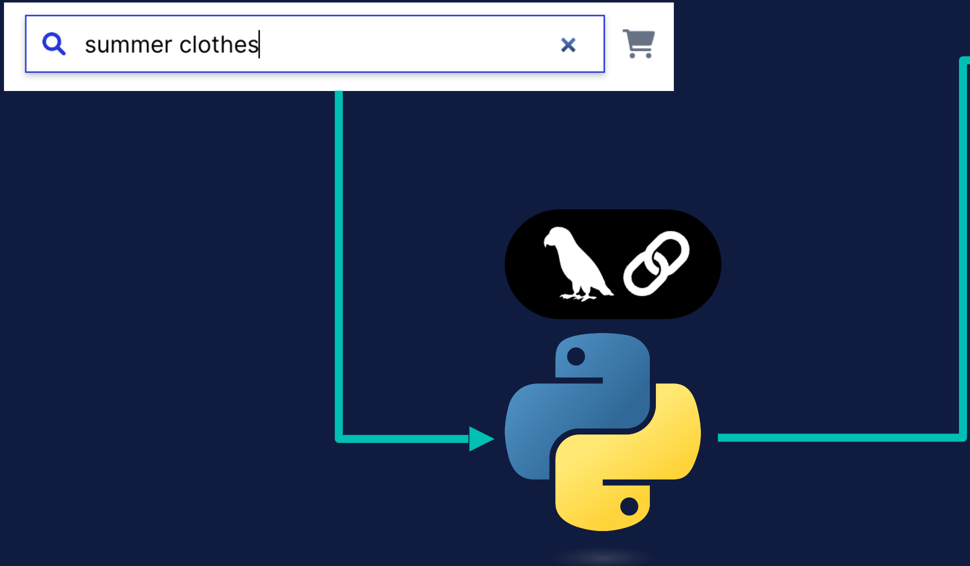
elastic

How do you search vectors?

Architecture of Vector Search



knn query



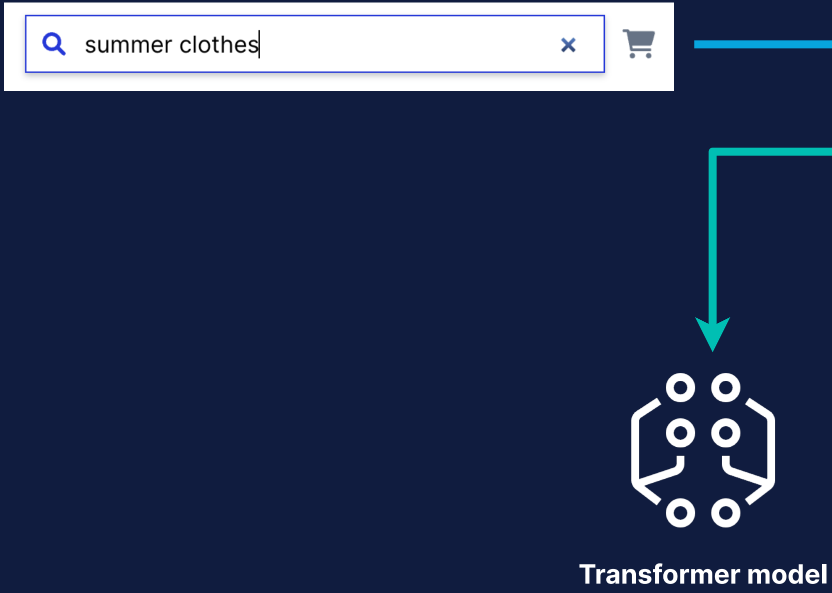
```
GET ecommerce/_search
{
 "query" : {
    "bool": {
      "must": [{
         "knn": {
           "field": "desc_embbeding",
           "query_vector": [0.123, 0.244,...]



         }
      }],
      "filter": {
        "term": {
          "department": "women"
        }
      }
    }
  },
  "size": 10
}
```

elastic

`knn` query (with Elastic ML)



```
GET ecommerce/_search
{
 "query" : {
    "bool": {
      "must": [{
         "knn": {
           "field": "desc_embbeding",
           "query_vector_builder": {
             "text_embedding": {
               "model_text": "summer clothes",
               "model_id": <text-embedding-model>
             }
           }
         }
      }],
      "filter": {
        "term": {
          "department": "women"
        }
      }
    }
  },
  "size": 10
}
```

semantic_text field type

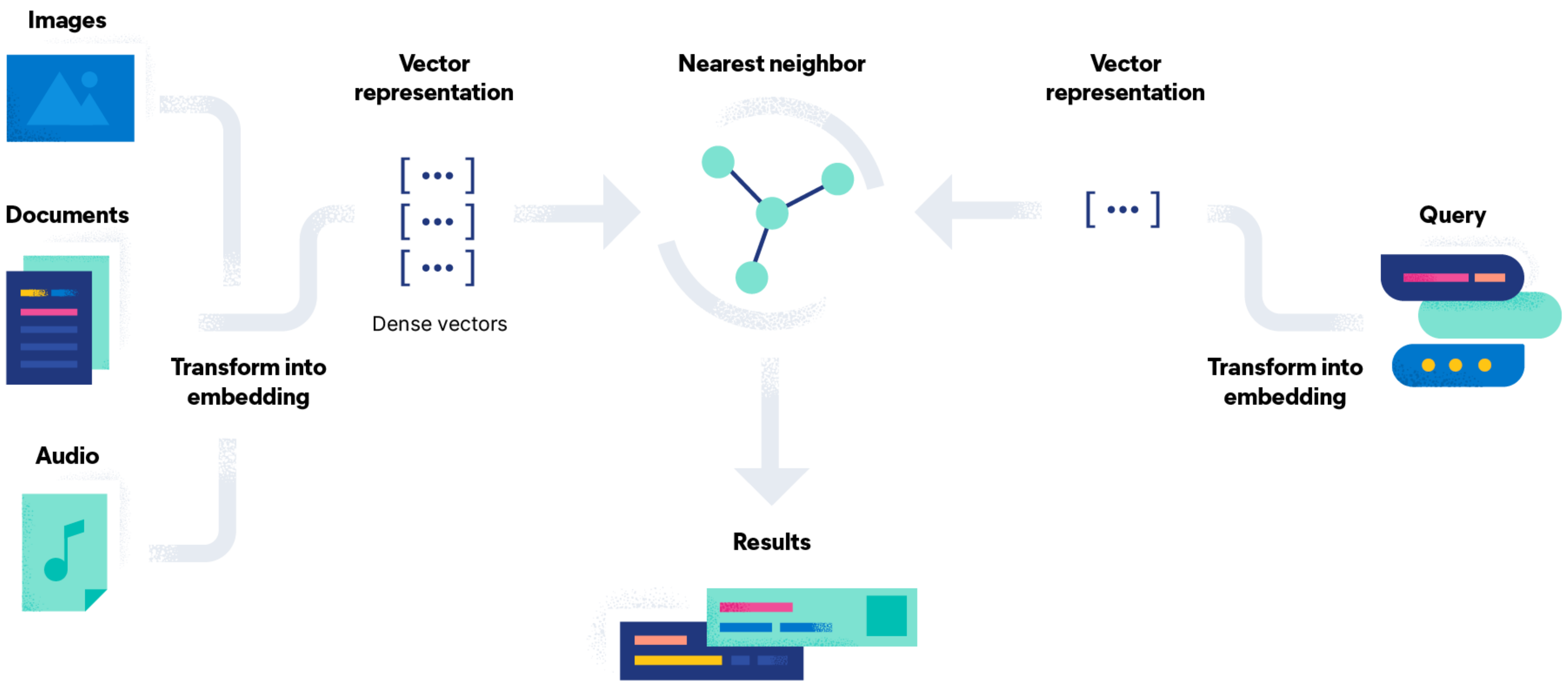
```
PUT /_inference/text_embedding/e5-small-multilingual
{
    "service": "elasticsearch",
    "service_settings": {
       "num_allocations": 1,
       "num_threads": 1,
       "model_id": ".multilingual-e5-small_linux-x86_64"
    }
}
```

```
PUT ecommerce
{
  "mappings": {
    "properties": {
      "description": {
        "type": "text",
        "copy_to": [ "desc_embedding" ]
      }
      "desc_embedding": {
        "type": "semantic_text",
        "inference_id": "e5-small-multilingual"
      }
    }
  }
}
```

```
POST ecommerce/_doc
{
  "description": "Our best-selling…"
}
```

```
GET ecommerce/_search
{
  "query": {
    "semantic": {
      "field": "desc_embedding"
      "query" : "I'm looking for a red dress for a DJ party"
}}}
```

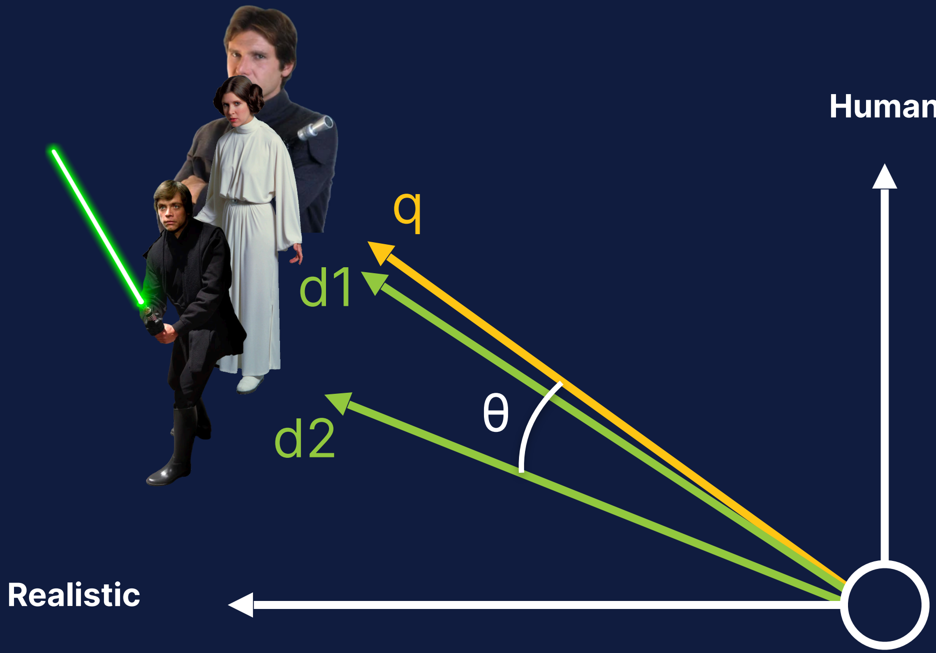
Architecture of Vector Search



But how does it

really work?

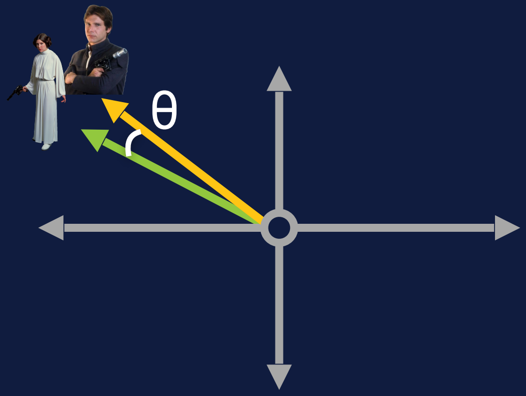
Similarity



$$cos(\theta) = \frac{\vec{q} \times \vec{d}}{|\vec{q}| \times |\vec{d}|}$$

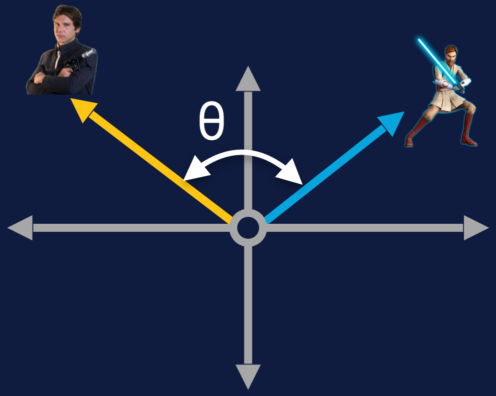
$$\_score = \frac{1 + cos(\theta)}{2}$$

Similarity: cosine (cosine)



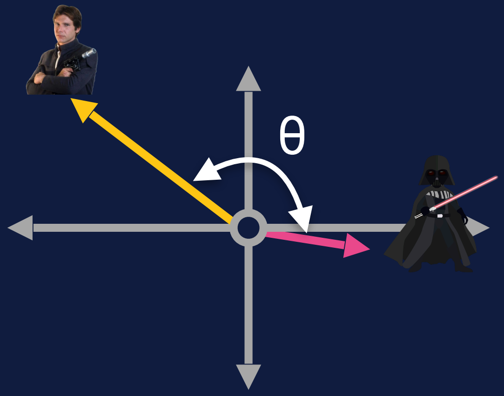
Similar vectors
θ close to 0
cos(θ) close to **1**

$$\_score = \frac{1+1}{2} = 1$$



Orthogonal vectors
θ close to 90°
cos(θ) close to **0**

$$\_score = \frac{1+0}{2} = 0.5$$



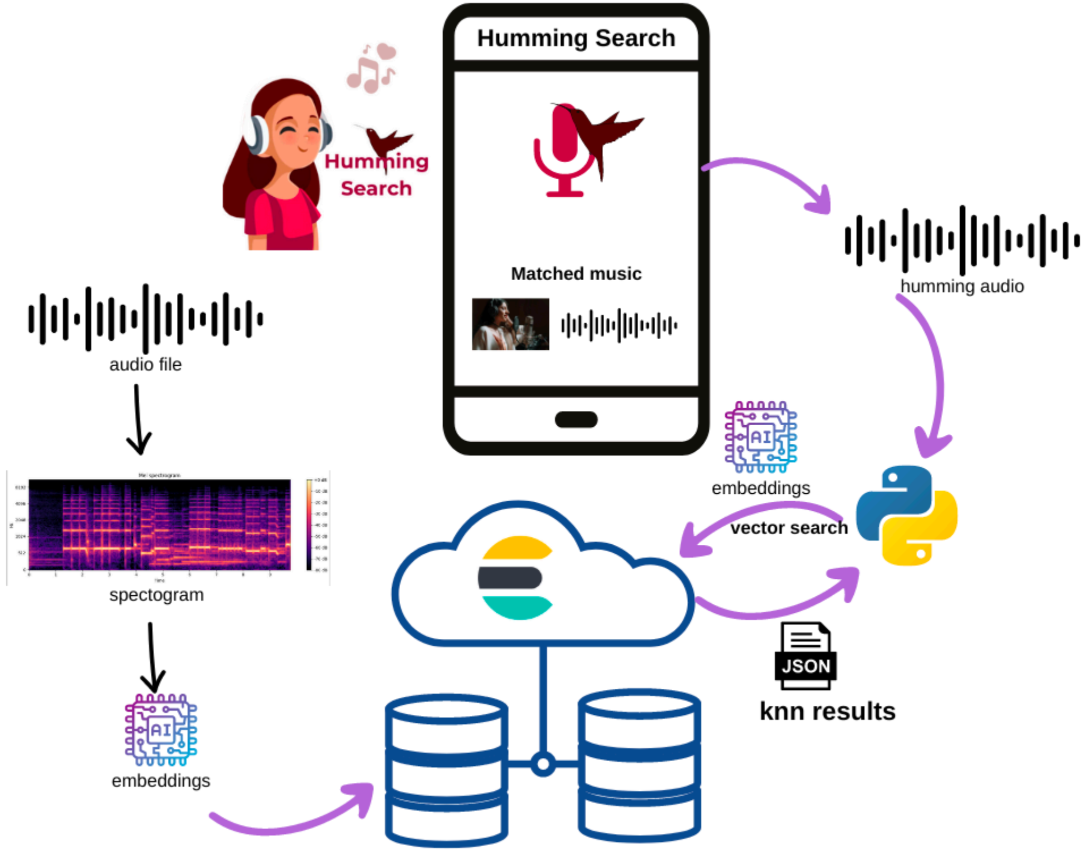
Opposite vectors
θ close to 180°
cos(θ) close to **-1**

$$\_score = \frac{1-1}{2} = 0$$

elastic

https://djdadoo.pilato.fr/

elastic



https://github.com/dadoonet/music-search/

elastic

elastic

Search

a new era

David Pilato | @dadoonet

BNP PARIBAS FORTIS