



Go-ing Gopher Hunting with Elasticsearch and Go

Carly Richmond
X@CarlyLRichmond

Elasticsearch: You Know, for Search



elasticsearch

lucene

Elasticsearch Go Client

2 Supported API Types:

- Low-level API
- Fully Typed API

Latest version: v8.9

Compatible with Go version 1.13+



Elasticsearch Go Client



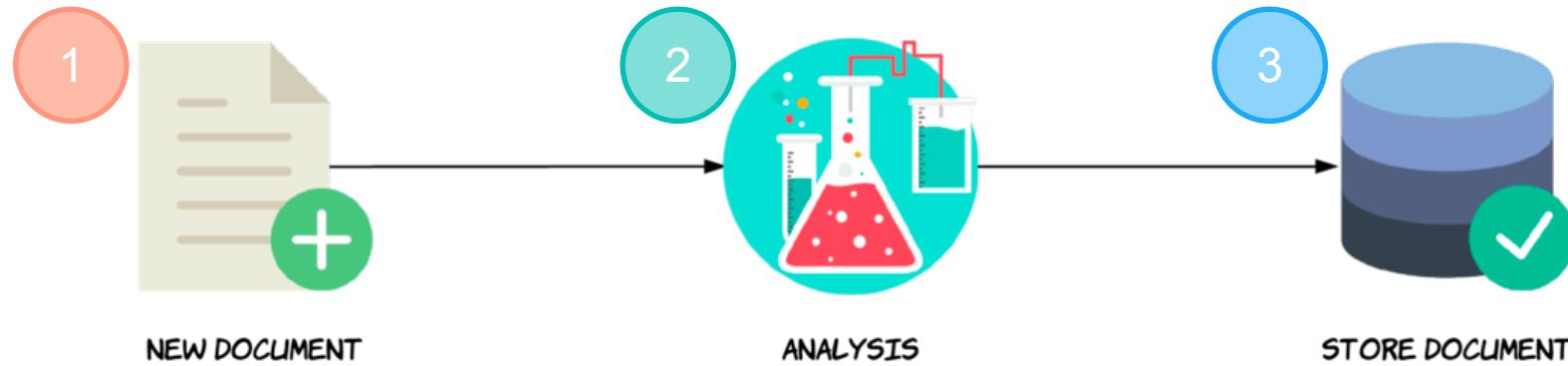
search.go

```
// Elasticsearch init
var cloudID = os.Getenv("ELASTIC_CLOUD_ID")
var apiKey = os.Getenv("ELASTIC_API_KEY")

var client, err =
    elasticsearch.NewTypedClient(elasticsearch.Config{
        CloudID: cloudID,
        APIKey: apiKey,
    })
```



Elasticsearch: You Know, for Keyword Search



[Credit: Understanding Analysis in Elasticsearch \(Analyzers\) Published on May 5, 2018 by Bo Andersen](#)



2

CHARACTER FILTERS

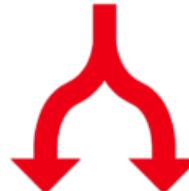
Two words!



Two words!

TOKENIZER

Two words!



[Two, words]

TOKEN FILTERS

[Two, words]



[two, words]

Credit: Understanding Analysis in Elasticsearch (Analyzers) Published on May 5, 2018 by Bo Andersen

Document



Contents [hide]
(Top)
Description
Behavior
Control
Classification
In popular culture
See also
References
External links

WIKIPEDIA
The Free Encyclopedia

Search Wikipedia Search

Create account Log in ...

Gopher

Article Talk

42 languages

From Wikipedia, the free encyclopedia

This article is about the rodent. For other uses, see [Gopher \(disambiguation\)](#).

Pocket gophers, commonly referred to simply as **gophers**, are **burrowing rodents** of the family **Geomysidae**.^[2] The roughly 41 species^[3] are all **endemic** to North and Central America.^[4] They are commonly known for their extensive tunneling activities and their ability to destroy farms and gardens.

The name "pocket gopher" on its own may refer to any of a number of **genera** within the **family Geomyidae**. These are the "true" gophers, but several **ground squirrels** in the distantly related family **Sciuridae** are often called "gophers", as well. The origin of the word "gopher" is uncertain; the French *gaufre*, meaning *waffle*, has been suggested, on account of the gopher tunnels resembling the *honeycomb*-like pattern of holes in a waffle;^[5] another suggestion is that the word is of *Muskogee* origin.^[6]

Description

Gophers weigh around 200 g ($\frac{1}{2}$ lb), and are about 15–20 cm (6–8 in) in body length, with a tail 2.5–5 cm (1–2 in) long. A few species reach weights approaching 1 kg (2.2 lb). Within any species, the males are larger than the females, and can be nearly double their weight.^[7]

Average lifespans are one to three years.^[8] The maximum lifespan for the pocket gopher is about five years.^[9] Some gophers, such as those in the genus *Geomys*, have lifespans that have been documented as up to seven years in the wild.^[10]



Botta's pocket gopher (*Thomomys bottae*)

gopher.json

```
{  
  "_index": "search-rodents",  
  "_id": "64f74ecd4acb3df024d91112",  
  "_score": 1.7641908,  
  "_ignored": ["body_content.enum"],  
  "_source": {  
    "last_crawled_at": "2023-09-13T18:25:06Z",  
    "additional_urls": ["https://en.wikipedia.org/wiki/Gopher"],  
    "body_content": "Pocket gophers, commonly referred to simply as gophers, are burrowing rodents of the family Geomyidae.",  
    "domains": ["https://en.wikipedia.org"],  
    "title": "Gopher - Wikipedia",  
    "url": "https://en.wikipedia.org/wiki/Gopher"  
  }  
}
```

[Source: Elastic Web Crawler](#)

[Alternative: Exercise: Web Crawler | Tour of Go](#)



Keyword Search

```
● ● ● keyword-search.go

// Traditional keyword search example
func KeywordSearch(term string) []Rodent {
    res, err := client.Search().
        Index("search-rodents").
        Query(&types.Query{
            Match: map[string]types.MatchQuery{
                "title": {Query: term},
            },
        }).
        From(0).
        Size(10).
        Do(context.Background())

    if err != nil {
        log.Fatal(err)
        return nil
    }

    return GetRodents(res.Hits.Hits)
}
```

Result Processing

```
rodent-processing.go

func GetRodents(hits []types.Hit) []Rodent {
    var rodents []Rodent

    for _, hit := range hits {
        var currentRodent Rodent
        err := json.Unmarshal(hit.Source_, &currentRodent)

        if err != nil {
            log.Fatal(err)
            return nil
        }

        currentRodent.ID = hit.Id_
        rodents = append(rodents, currentRodent)
    }

    return rodents
}
```

Semantic Search: Meaning, not literal matches

Elasticsearch: You Know, for Vector Search

Bag of Words

Pocket gophers, commonly referred to simply as gophers, are burrowing rodents of the family Geomyidae.

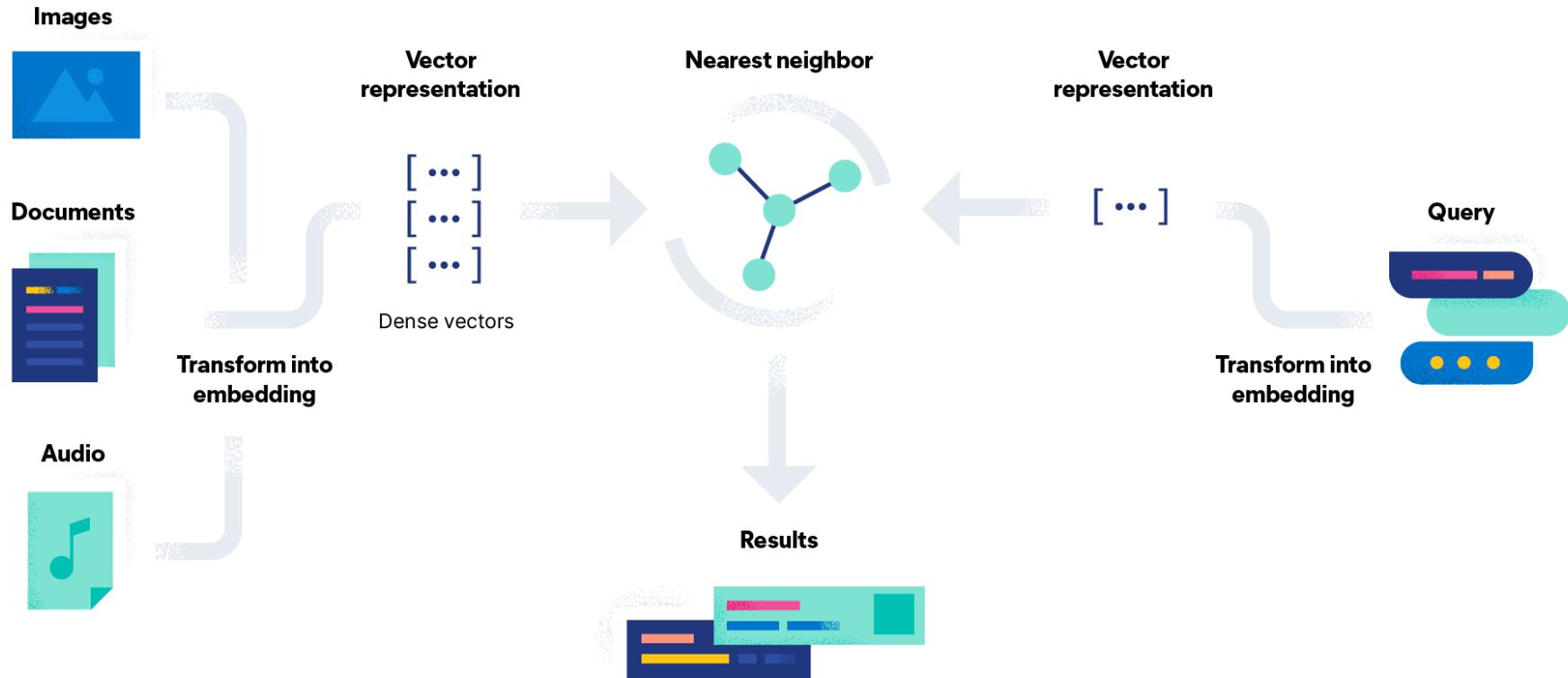
The capybara, or greater capybara, is a giant cavy rodent.

pocket: 1, **gopher: 2**, commonly: 1, referred: 1, simply: 1, burrowing: 1, **rodent: 2**, family: 1, geomyidae: 1, capybara: 2, greater: 1, giant: 1, cavy: 1, native: 1

[pocket, **gopher**, commonly, referred, simply, burrowing, **rodent**, family, geomyidae, capybara, greater, giant, cavy, native]

[1, **2**, 1, 1, 1, 1, **1**, 1, 1, 0, 0, 0, 0, 0]
[0, **0**, 0, 0, 0, 0, **1**, 0, 0, 2, 1, 1, 1, 1]

Vector Search



How vectors are indexed for search: graphs

Indexing documents with dense vectors



Hierarchical Navigable Small Worlds:

a layered approach that simplifies access to the nearest neighbor



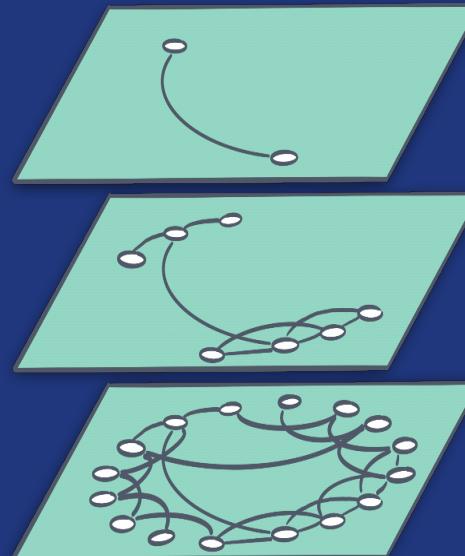
Tiered: from coarse to fine approximation over a few steps



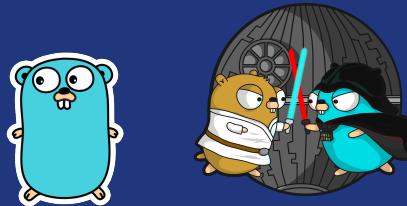
Balance: Bartering a little accuracy for a lot of scalability



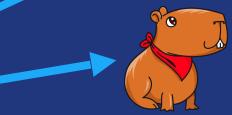
Speed: Excellent query latency on large scale indices



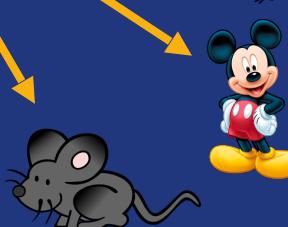
GOPHER



REALISTIC



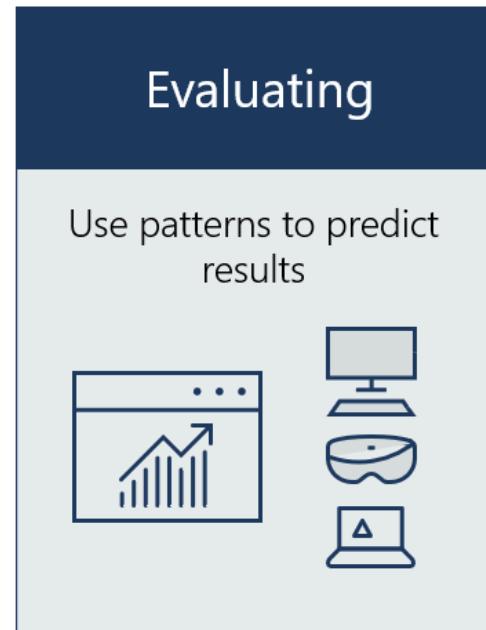
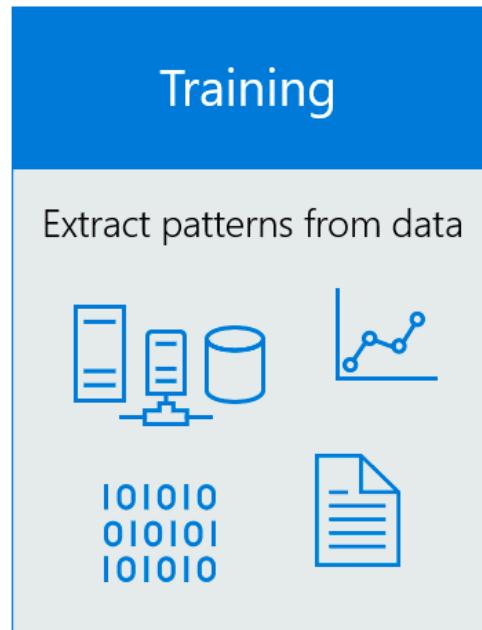
CARTOON



MOUSE



What is a model?



Free vs Paid (Platinum)

kNN with HNSW

1. Bring your own embeddings
2. Enough memory for vectors to fit in off-heap RAM

Inference in Elasticsearch

1. Generate embeddings
(like HuggingFace transformers)
2. ELSER

Setting Up the Embedding Model

The screenshot shows the Hugging Face Model Hub search results for the query "sentence-transformers/msmarco-MiniLM-L-12-v3". The results page includes filters for Tasks (e.g., Question Answering, Text Classification), Datasets (e.g., SQuAD, TriviaQ), and Languages (e.g., English, French). The main search results list includes "sentence-transformers/msmarco-MiniLM-L-12-v3" at the top, followed by other models like "distilbert-base-uncased", "roberta-base", and "t5-base". Each model entry provides details such as the number of downloads and the last update date.

```
$ eland_import_hub_model  
--cloud-id $ELASTIC_CLOUD_ID \  
--es-api-key $ELASTIC_API_KEY \  
--hub-model-id sentence-  
transformers/msmarco-MiniLM-  
L-12-v3 \  
--task-type text_embedding \  
--start
```

The screenshot shows the eland UI's "Trained Models" section. It lists the trained model "sentence-transformers/msmarco-MiniLM-L-12-v3" along with its configuration parameters: "Encoder", "Input text", and "Model type: sentence-transformers/msmarco-MiniLM-L-12-v3". Below this, the "Deployment stats" table shows the instance ID, state, node name, size, and address. The "Model stats" table provides detailed information about the model's size and memory usage.



Select the appropriate model



Load the model to the cluster



Manage models



Vector Document

The screenshot shows the 'Ingest Pipelines' section of the Elastic Stack Management interface. It displays a list of pipelines, with one pipeline named 'search-rodents-vector-embedding-pipeline' selected. The pipeline configuration is shown in JSON format:

```
{
  "processors": [
    {
      "inference": {
        "model_id": "sentence-transformers__msmarco-minilm-l1-12-v3",
        "target_field": "text_embedding",
        "field_map": {
          "body_content": "text_field"
        }
      }
    }
  ]
}
```

The screenshot shows the Dev Tools Console in the Elastic Stack Management interface. A POST _reindex request is being typed into the console:

```
POST _reindex
{
  "source": {
    "index": "search-rodents"
  },
  "dest": {
    "index": "vector-search-rodents",
    "pipeline": "search-rodents-vector-embedding-pipeline"
  }
}
```

The screenshot shows the search results in the Dev Tools Console. A document about pocket gophers is returned:

```
{
  "_index": "vector-search-rodents",
  "_id": "64f174cc4cb3df024d91112",
  "_score": 2.262317,
  "_ignored": ["body_content.keyword"],
  "source": {
    "last_crawled_at": "2023-09-13T18:25:06Z",
    "text_embedding": {
      "predicted_value": [
        -0.1321354481220245, -0.3257530927658081, 0.06877975910902023,
        0.05385938659310341, -0.21258828043937683, -0.053170233964920044
      ],
      "ts_truncated": true,
      "model_id": "sentence-transformers__msmarco-minilm-l1-12-v3"
    },
    "body_content": "Pocket gophers, commonly referred to simply as gophers, are burrowing rodents of the family Geomyidae."
  },
  "domains": ["https://en.wikipedia.org"],
  "title": "Gopher - Wikipedia",
  "url": "https://en.wikipedia.org/wiki/Gopher"
}
```



PyTorch

Enrich data using
inference
processor

[Ingest pipeline Inference processor | Elastic](#)



Reindex data
using ingest
pipeline



Generate
embeddings

elastic

Vector Search

```
vector-search-filtered.go

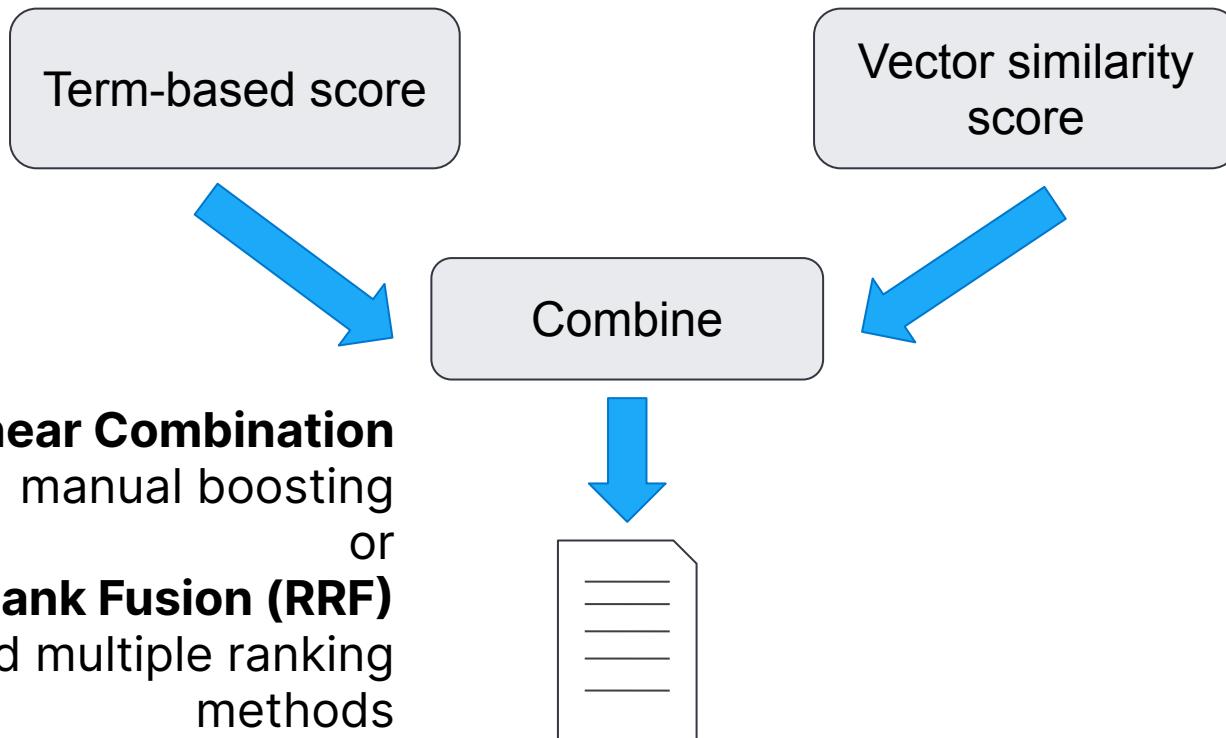
// Vector search example with filter
func VectorSearchWithFilter(term string) []Rodent {
    res, err := client.Search().
        Index("vector-search-rodents").
        Knn(types.KnnQuery{
            Field: "text_embedding.predicted_value",
            K:      10,
            Filter: []types.Query{
                {
                    Match: map[string]types.MatchQuery{
                        "body_content": {Query: "rodent"},
                    },
                },
            },
            NumCandidates: 10,
            QueryVectorBuilder: &types.QueryVectorBuilder{
                TextEmbedding: &types.TextEmbedding{
                    ModelId:   "sentence-transformers--msmarco-minilm-l-12-v3",
                    ModelText: term,
                },
            }).Do(context.Background())

    if err != nil {
        log.Fatal(err)
        return nil
    }

    return GetRodents(res.Hits.Hits)
}
```

Elasticsearch: You Know, for Hybrid Search

Hybrid Scoring



Manual Boosting

```
hybrid-search-manual.go

// Hybrid search example
func HybridSearchWithBoost(term string) []Rodent {
    var knnBoost float32 = 0.2

    res, err := client.Search().
        Index("vector-search-rodents").
        Knn(types.KnnQuery{
            Field:           "text_embedding.predicted_value",
            Boost:          &knnBoost,
            K:              10,
            NumCandidates: 10,
            QueryVectorBuilder: &types.QueryVectorBuilder{
                TextEmbedding: &types.TextEmbedding{
                    ModelID:   "sentence-transformers__msmarco-minilm-l-12-v3",
                    ModelText: term,
                },
            },
        }).
        Query(&types.Query{
            Match: map[string]types.MatchQuery{
                "title": {Query: term},
            },
        }).
        From(0).
        Size(2).
        Do(context.Background())

    if err != nil {
        log.Fatal(err)
        return nil
    }

    return GetRodents(res.Hits.Hits)
}
```

Reciprocal Rank Fusion



rrf.py

```
score = 0.0
for q in queries:
    if d in result(q):
        score += 1.0 / ( k + rank( result(q), d ) )
return score

# where
# k is a ranking constant
# q is a query in the set of queries
# d is a document in the result set of q
# result(q) is the result set of q
# rank( result(q), d ) is d's rank within the result(q)
# starting from 1
```

Reciprocal Rank Fusion

```
hybrid-search-rrf.go

// Hybrid search with RRF
func HybridSearchWithRRF(term string) []Rodent {
    var windowSize int64 = 10 // min required
    var rankConstant int64 = 42

    res, err := client.Search()
        Index("vector-search-rodents").
        Knn(types.KnnQuery{
            Field: "text_embedding.predicted_value",
            K: 10,
            NumCandidates: 10,
            QueryVectorBuilder: &types.QueryVectorBuilder{
                TextEmbedding: &types.TextEmbedding{
                    ModelId: "sentence-transformers--msmarco-minilm-l-12-v3",
                    ModelText: term,
                },
            }).
            Query(&types.Query{
                Match: map[string]types.MatchQuery{
                    "title": {Query: term},
                },
            }).
            Rank(&types.RankContainer{
                Rrf: &types.RrfRank{
                    WindowSize: &windowSize,
                    RankConstant: &rankConstant,
                },
            }).
            Do(context.Background())
}

if err != nil {
    log.Fatal(err)
    return nil
}

return GetRodents(res.Hits.Hits)
}
```

Thank you!



Carly Richmond
X@CarlyLRichmond