Elasticsearch Features

An Overview

ENTERPRISE PRODUCTIVITY

Contents

Introduction

Elasticsearch is an open source search product which is a scalable search product with great features. Elasticsearch cluster can be setup including number of servers or machines where search indices can be distributed. Most of search features of the Elasticsearch is free. There are features like encryption, machine learning which come under a package called XPack which require license. This document covers some interesting features provided by Elasticsearch.

Location Based Search

In today's world where people use various mobile devices and want information based on their current location, location specific search has great relevance. In general search is performed based on keyword provided by user. But if search result can be provided based on current location of the user then it adds more value and relevance. Elasticsearch provides feature which can return search result relevant to user's location. When data is indexed in Elasticsearch, the data need to be indexed with metadata like latitude and longitude. Below is an example of indexing data with latitude and longitude:

```
PUT conferences/conference/1
{
  "text": "Conference on AI enhanced sports","country":"US","category":"sports",
  "location": {
    "lat": 41.12,
    "lon": -71.34
  }
}
```

While performing the search query, user's current latitude and longitude need to be obtained and then passed in the search query. Modern browsers like Chrome have feature where they can provide user's latitude and longitude. Indexed data can be searched based on user's latitude, longitude within a particular radius of the user current location.

Example search query with co-ordinates:

```
        {
```

```
        "filtered" : {
            "query" : {
                "field" : { "text" : "restaurant" }
            },
            "filter" : {
                "geo_distance" : {
                    "distance" : "12km",
                    "pin.location" : {
                        "lat" : 40,
                        "lon" : -70
                    }
                }
            }
        }
}
```

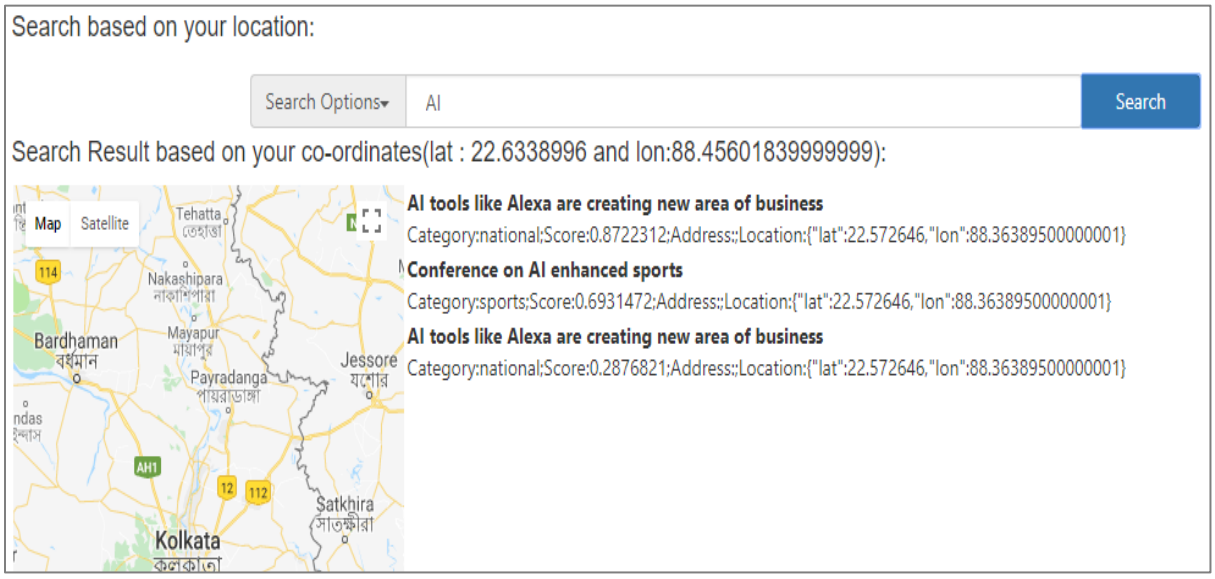

Figure 1:Search result based on location

Search Social Media(Twitter) data from Elasticsearch

Elasticsearch stack consists of Elasticsearch, Kibana and Logstash. Kibana is front end for Elasticsearch and Logstash is an open source, server-side data processing pipeline that ingests data from a multitude of sources simultaneously, transforms it, and then sends it to your favourite "stash" such as Elasticsearch. Logstash has number of plugins to pull data from various sources like PostgreSQL, MySQL, Twitter etc. For reading data from Twitter, an app need to be registered in Twitter. Then consumer key, consumer secret, oauth token, oauth token secret of the app can be used by Logstash to read data from Twitter. Logstach can continuously read data from Twitter based on a keyword and index the data to Elasticsearch. User can search indexed Twitter data from Elasticsearch.

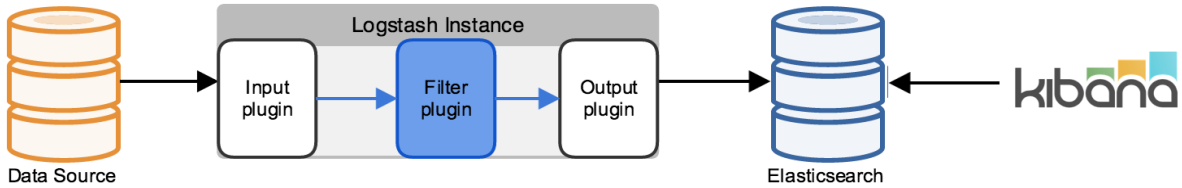

Figure 2: Integrating various data sources to Elasticsearch

Query Boosting in Elasticsearch

Search result can be promoted in Elasticsearch based on condition. The search query can be defined in such a way to promote results matching certain condition.

Examples of query boosting:

```
GET /_search
        {
           "query": {
              "bool": {
                 "must": {
                    "match": {
                       "content": {
                          "query":    "full text search",
                          "operator": "and"
```

```
            }
        }
          },
          "should": [
            { "match": { "content": "Elasticsearch" }},
            { "match": { "content": "Lucene"       }}
          ]
        }
      }
    }
```

In the above example the search query is looking for data containing text ‘full text search’. Again in the ‘should’ clause of the query there is text ‘Elasticsearch’ and ‘Lucene’. If the document contains additional text ‘Elasticsearch’ and ‘Lucene’ then that document will be promoted in the search result.

Examples of query boosting

```
GET /_search
{
   "query": {
      "bool": {
         "must": {
            "match": {
               "content": {
                  "query":    "full text search",
                  "operator": "and"
               }
            }
         },
         "should": [
            { "match": {
```

```
            "content": {
              "query": "Elasticsearch",
              "boost": 3
            }
          }},
          { "match": {
            "content": {
              "query": "Lucene",
              "boost": 2
            }
          }}
        ]
      }
    }
}
```

In the above query, data containing 'Elasticsearch' and 'Lucene' will be boosted by weight of 3 and 2 which will promote any data containing these texts. The query boosting can be used to implement relevance and additional promoting factor while displaying search result to end user.

Machine Learning in Elasticsearch

Elasticsearch offers Machine Learning feature as part of XPack Subscription. Machine Learning jobs can be created on the data indexed in Elasticsearch. These jobs can provide insight and help in anomaly detection by analyzing large volume of data over a period of time. It has forecasting feature for selected future time period. This feature is useful for analyzing anomalies in log data and their root causes. This feature can help analyzing non time series data to detect anomalies. The below diagram shows how data indexed over a period of can be used to 'learn' and then predict future behavior by Machine Learning jobs. Once any anomaly is detected then notifications can be setup to inform relevant people about the anomaly.

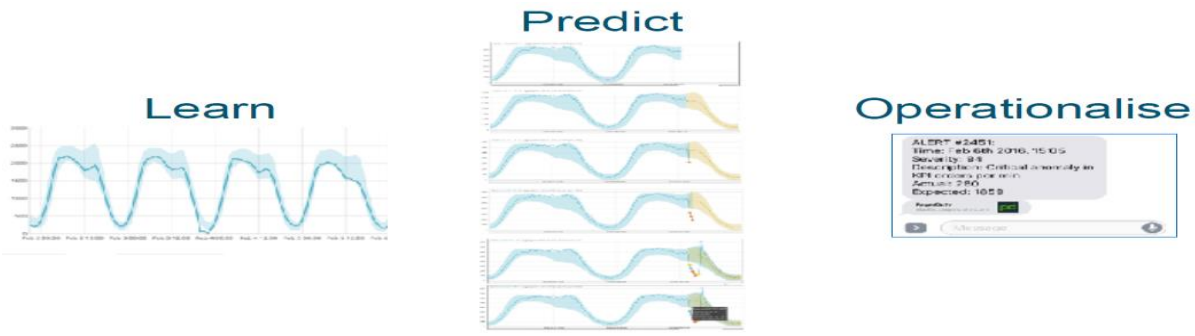

Figure 3: Machine Learning jobs

The below screenshot is an example of the Machine Learning job detecting anomaly in Kibana UI.

Figure 4:Anomaly Detection in ML job

The below screenshot is an example of the Machine Learning job doing forecasting for a future period in Kibana UI.

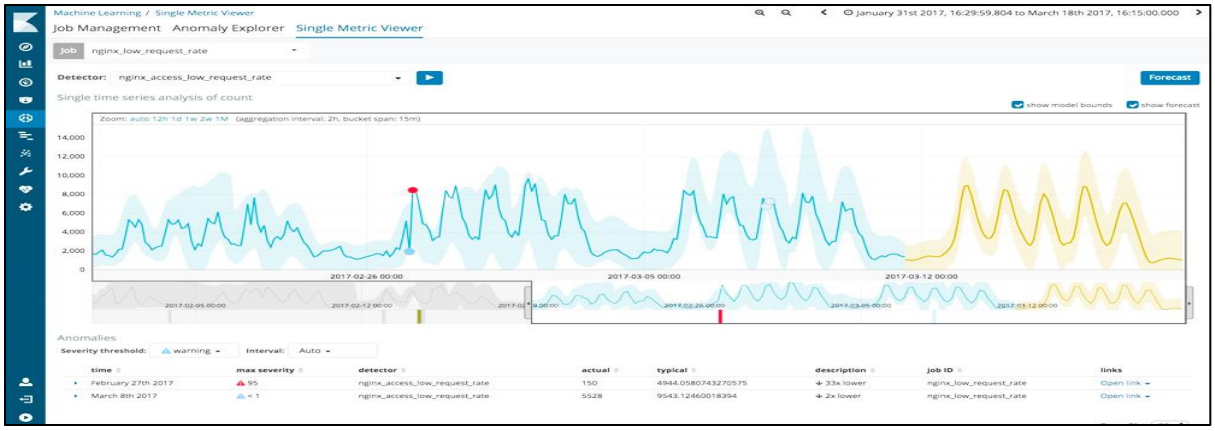

Figure 5:Forecasting in ML job

Possible Use cases

Location based search: There can be many use cases for this feature. For example, an user looking for a Food store or Hospital or Shopping Mall near his location or a parent looking for a school for his child near his residence.

Machine Learning: If daily production logs of a web server is indexed in Elasticsearch then Machine Learning jobs can point to any anomaly in server resource usage or end user behavior. For example, more server resources are being used during peak hours. For a Shopping portal, during 'New Year Sale' there may be more user hits. Machine Learning jobs can point to any such anomaly and forecast future behavior.

Query Boosting: This feature can be used to promote search result based on user preferences such as user whose interest include sports will always get search results related to sports higher in order in search result.

Social Media Search: This feature can be used to get combined search result for a user from within the application. The content source of search can be both data indexed in Elasticsearch and data from social media.

Alternative Solutions

There are other open source search products like Apache Solr, Lucene, Sphinx which can be explored as alternative solution.

Next Steps

Elasticsearch provides cloud based infrastructure (SAAS) and services which can be explored.

Conclusion

As an open source search product, Elasticsearch offers great features. Organizations not willing to spend too much for a search product can definite explore Elasticsearch which is scalable and fast. A large number organizations do use Elasticsearch already. Elasticsearch have been adding new features like Machine Learning to improve their offerings. Also, products like Logstash allow users to write their own plugins to connect to any data source which already do not have any plugin available. On the whole, the Elastic stack including Elasticsearch, Kibana and Logstash give lot of features and overall coverage for any organization looking for implementing search and analysis.