DejaVu Sans

Visual Analytics Assignment 2 Report

Zitian

April 28, 2025

1 Section 1: Data Indexing

1.1 Data Cleaning

The raw CSV data was first cleaned using a Python script to ensure all fields are well-typed, missing values are handled, and city names are standardized. The main steps include:

- Converting numeric fields to the correct type (e.g., ratings, votes, cost).
- Filling missing values with defaults.
- Standardizing date and location formats.
- Removing duplicates.
- Outputting a cleaned CSV file for indexing.

Key code snippet:

Listing 1: Key data cleaning code

```
df['AggregateRating'] = pd.to_numeric(df['AggregateRating'],
    errors='coerce')
df['Votes'] = pd.to_numeric(df['Votes'], errors='coerce')
df['AverageCostForTwo'] = pd.to_numeric(df['
    AverageCostForTwo'], errors='coerce')
df['RestaurantName'] = df['RestaurantName'].fillna('Unknown
    Restaurant')
df['Date'] = df['Date'].apply(validate_date)
df.to_csv('Assignment2/restaurants_cleaned.csv', sep=';',
    index=False)
```

1.2 Indexing with Grok Pipeline

To ingest the cleaned data into Elasticsearch, we initially used a Grok pipeline but later switched to a more reliable direct JSON document approach. Here's our journey and lessons learned:

1.2.1 Initial Approach: Grok Pipeline

The initial approach used a Grok pipeline to parse raw CSV lines:

- Converting each row to a single raw string
- Using Grok patterns to parse fields
- Converting field types after parsing

However, this approach had limitations:

- Complex string parsing could lead to data loss
- Field type conversion was error-prone
- Debugging was difficult when fields were missing

1.2.2 Improved Approach: Direct JSON Documents

We improved the indexing process by directly constructing JSON documents:

- Each field is explicitly typed and mapped
- No intermediate string parsing required
- Better control over data integrity

Key code snippet of the improved approach:

Listing 2: Direct JSON document construction

```
def prepare_bulk_data(csv_file):
    df = pd.read_csv(csv_file, sep=';')
    bulk_data = []
    for _, row in df.iterrows():
        doc = {
            "SerialNumber": int(row['SerialNumber']),
            "RestaurantName": str(row['RestaurantName']),
            "AverageCostForTwo": int(row['AverageCostForTwo'
                ]),
            "AggregateRating": float(row['AggregateRating'])
                ,
            "RatingText": str(row['RatingText']),
            "Votes": float(row['Votes']),
            "Date": str(row['Date']),
            "Coordinates": str(row['Coordinates']),
            "City": str(row['City']),
            "Country": str(row['Country']),
            "Continent": str(row['Continent']),
            "City/Country/Continent": f"{str(row['City'])}/{
                str(row['Country'])}/{str(row['Continent'])}"
        }
```

```
        bulk_data.append(json.dumps({"index": {"_index":
            INDEX_NAME}}))
        bulk_data.append(json.dumps(doc))
    return '\n'.join(bulk_data) + '\n'
```

1.2.3 Key Learnings

Through this process, we learned several important lessons:

- Direct data mapping is more reliable than string parsing
- Explicit type conversion prevents data loss
- Simpler processing pipelines are easier to debug
- Maintaining data integrity requires careful handling at each step

Result: All cleaned restaurant data is now correctly indexed in Elasticsearch as structured JSON documents, with all fields properly preserved and typed.

1.3 Query 1: Find restaurants with 'bar' in the name but not 'barbecue(s)' or 'barbeque(s)'

The goal is to retrieve all restaurants whose names contain the substring `bar`, but do not contain `barbecue`, `barbeque`, `barbecues`, or `barbeques`. Only the restaurant name, city/country/continent, and number of votes are returned.

Listing 3: Elasticsearch query for 'bar' but not 'barbecue(s)' or 'barbeque(s)'

```
GET restaurants/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "wildcard": {
            "RestaurantName": "*bar*"
          }
        }
      ],
      "must_not": [
        {
          "wildcard": {
            "RestaurantName": "*barbecue*"
          }
        },
        {
          "wildcard": {
            "RestaurantName": "*barbeque*"
          }
```

```
        },
        {
          "wildcard": {
            "RestaurantName": "*barbecues*"
          }
        },
        {
          "wildcard": {
            "RestaurantName": "*barbeques*"
          }
        },
        {
          "match": {
            "RestaurantName": "bar"
          }
        }
      ]
    }
  },
  "_source": ["RestaurantName", "City/Country/Continent", "
     Votes"]
}
```

Query Result Summary

The query returned **89** restaurants whose names contain "bar" but do not contain "barbecue(s)" or "barbeque(s)".

Full results are provided in the file: `section1QueryResult/query1_results.json` (see submission folder).

Below are some sample results:

```
{
  "RestaurantName": "Paribar",
  "City/Country/Continent": "Sao Paulo/Brazil/",
  "Votes": 46
},
{
  "RestaurantName": "Bardenay",
  "City/Country/Continent": "Boise/United States/",
  "Votes": 879
},
{
  "RestaurantName": "Barbacoa Restaurant",
  "City/Country/Continent": "Boise/United States/",
  "Votes": 538
}
```

A Full Query 1 Results

The complete results for Query 1 are provided below:

Listing 4: All results for Query 1


```
{
    "took": 34,
    "timed_out": false,
    "_shards": {
      "total": 1,
      "successful": 1,
      "skipped": 0,
      "failed": 0
    },
    "hits": {
      "total": {
        "value": 89,
        "relation": "eq"
      },
      "max_score": 1,
      "hits": [
        {
          "_index": "restaurants",
          "_id": "QE5hfZYBQKwotppyNThx",
          "_score": 1,
          "_source": {
            "RestaurantName": "Paribar",
            "City/Country/Continent": "S    O  Paulo/Brazil/"
                ,
            "Votes": 46
          }
        },
        {
          "_index": "restaurants",
          "_id": "hk5hfZYBQKwotppyNThx",
          "_score": 1,
          "_source": {
            "RestaurantName": "Bardenay",
            "City/Country/Continent": "Boise/United States/"
                ,
            "Votes": 879
          }
        },
        {
          "_index": "restaurants",
          "_id": "i05hfZYBQKwotppyNThx",
          "_score": 1,
          "_source": {
            "RestaurantName": "Barbacoa Restaurant",
```

```
            "City/Country/Continent": "Boise/United States/",
            "Votes": 538
          }
        },
        {
          "_index": "restaurants",
          "_id": "ik5hfZYBQKwotppyNTly",
          "_score": 1,
          "_source": {
            "RestaurantName": "Barrett Junction Cafe",
            "City/Country/Continent": "Potrero/United States/",
            "Votes": 9
          }
        },
        {
          "_index": "restaurants",
          "_id": "uU5hfZYBQKwotppyNTly",
          "_score": 1,
          "_source": {
            "RestaurantName": "Rhubarb Le Restaurant",
            "City/Country/Continent": "Singapore/Singapore/",
            "Votes": 33
          }
        },
        {
          "_index": "restaurants",
          "_id": "f05hfZYBQKwotppyNTpy",
          "_score": 1,
          "_source": {
            "RestaurantName": "The Cafe Baraco",
            "City/Country/Continent": "Ahmedabad/India/",
            "Votes": 317
          }
        },
        {
          "_index": "restaurants",
          "_id": "Rk5hfZYBQKwotppyNTty",
          "_score": 1,
          "_source": {
            "RestaurantName": "Doon Darbar",
            "City/Country/Continent": "Dehradun/India/",
            "Votes": 121
          }
        },
        {
          "_index": "restaurants",
          "_id": "J05hfZYBQKwotppyNTzo",
```

```
            "_score": 1,
            "_source": {
              "RestaurantName": "The Grill Darbar",
              "City/Country/Continent": "Faridabad/India/",
              "Votes": 17
            }
          },
          {
            "_index": "restaurants",
            "_id": "TU5hfZYBQKwotppyNTzo",
            "_score": 1,
            "_source": {
              "RestaurantName": "Barista",
              "City/Country/Continent": "Ghaziabad/India/",
              "Votes": 33
            }
          },
          {
            "_index": "restaurants",
            "_id": "dk5hfZYBQKwotppyNTzo",
            "_score": 1,
            "_source": {
              "RestaurantName": "Zambar",
              "City/Country/Continent": "Gurgaon/India/",
              "Votes": 802
            }
          }
        ]
      }
    }
```