BeakPeek

Millennium
University Of Pretoria

Data Science Award

In Partnership With
AgileBridge

Group:
22

Mentor:
Ayaaz

2024/10/18

Contents

1 Bird Population Calculation For Endangered Birds **2**
- 1.1 Generating Modification Constant . . . 2
- 1.2 Detection Probability Problems . . . 2
- 1.3 Habitat Size . . . 2
- 1.4 Final Formula . . . 2
- 1.5 Conclusion . . . 2

2 Algorithms **2**
- 2.1 Data Science behind the HeatMap . . . 2
  - 2.1.1 Dynamic Pentad Data Loading . . . 3
  - 2.1.2 Data Visualization with Color Coding . . . 3
  - 2.1.3 Interactive Map Filtering . . . 3
  - 2.1.4 Efficient Data Rendering . . . 3
  - 2.1.5 Conclusion . . . 3
- 2.2 Data Science behind the BirdMap . . . 4
  - 2.2.1 Dynamic Location and KML Data Loading . . . 4
  - 2.2.2 Data Filtering by Province and Month . . . 4
  - 2.2.3 Polygon-Based Data Visualization . . . 4
  - 2.2.4 Real-Time Map Interaction . . . 4
  - 2.2.5 Conclusion . . . 5

3 Explanation of Algorithms in LifeList **5**
- 3.1 SQLite Tables . . . 5
  - 3.1.1 AllBirds Table . . . 5
  - 3.1.2 Bird Table . . . 5
  - 3.1.3 Provinces Table . . . 5
  - 3.1.4 Online Profile Update . . . 5
  - 3.1.5 Duplicate inserting . . . 6
  - 3.1.6 Bird Provinces . . . 6
  - 3.1.7 Achievement Progress . . . 6
  - 3.1.8 Overall Design . . . 6

1 Bird Population Calculation For Endangered Birds

1.1 Generating Modification Constant

Using information from : Estimating bird abundance: making methods work and Point Counts of Birds: What Are We Estimating? in combination with the few known bird populations in South Africa a constant, 9.01, was created. A massive issue that was encountered was that most bird population estimates are calculated for the bird species regardless of its habitat, this data clashes with our attempt to estimate the population of a bird species only for South Africa.

1.2 Detection Probability Problems

SABAP2, the database we use to get our raw data, relies on the sightings provided by South African birding professionals. This means that the viewing of birds is biased. From literature mentioned above and our own research it was established that the following skews the reporting rate of a bird: is the bird nocturnal, how popular is the location of their habitat, the rarity of a bird (the same bird may be reported by multiple birders a lot), a large group of birders going to a location only at a certain time of year meaning that migratory birds may be missed.

Towards solving this issue a standard reporting rate increase of 0.3 was added to all bird species. Since the goal of the algorithm is to estimate the population of endangered birds the issue of a reporting rate going above one does not need to be handled.

1.3 Habitat Size

The habitat size of a bird is calculated by taking the amount of pentads (9km x 9km area) they appear in and dividing it by 16 673 (The total number of pentads). This helps normalise the reporting rate between bird species.

1.4 Final Formula

At last all of this information comes together in this formula:

viewRate = (bird.fullProtocolRR+detectionProbability) * bird.fullProtocolNumber * pentads / 16673;

16673 all pentads ; 12497 pentads with more than 5 sightings ; 14288 pentads with atleast one sighting

population = (constant * viewRate).round();

Then all birds with a population of less than 20 000 individuals are considered endangered.

1.5 Conclusion

This formula accurately estimates the population of all South African birds so long as their report rate is above 1%, the formula also aims to always underestimate the population of birds by a small margin so that an endangered bird cannot falsely be reported as safe. We are currently working to try and further improve this formula and recently have started collaboration with Dr. Bernard Coetzee and his team towards this end. The biggest problem with the formula is that birds with a reporting rate of less than 1% have their population size grossly underestimated. In a perfect world a formula and constant as mentioned above would be created for each species of bird but we believed this to be out of the scope for this project.

2 Algorithms

2.1 Data Science behind the HeatMap

The HeatMap feature processes and visualizes a vast dataset, encompassing over **16,673 pentads** (geographic regions) and more than **1.4 million bird entries**. To manage such large-scale data, the app utilizes various algorithms to optimize data handling, visualization, and performance.

2.1.1 Dynamic Pentad Data Loading

Given the scale of the data, the `loadPentadData` function dynamically loads bird sighting data only as needed. Instead of attempting to load all 16,673 pentads and millions of bird entries simultaneously, the algorithm fetches data relevant to the user's selections, such as species and month, and updates the map accordingly. It parses the kml file and calculates the coordinates based on the data. The coordinates are then made into a polygon which is linked with the reporting rate of the bird.

Why it's effective:

- **Data Efficiency:** By fetching only necessary data based on user input, the app minimizes memory usage and processing time, allowing it to handle a database with over 1.4 million entries efficiently.
- **Batch Processing:** The large dataset is processed in small increments (batches), preventing performance degradation and ensuring smooth interaction even with vast amounts of data.

2.1.2 Data Visualization with Color Coding

With a database of this size, data visualization is key to transforming raw data into actionable insights. The `getColorForReportingRate` function assigns colors to polygons based on the reporting rate (i.e., bird sighting percentages in each pentad). This color coding makes it easier to interpret patterns across thousands of pentads.

Why it's effective:

- **Data-Driven Visualization:** Transforming over a million bird entries into an intuitive color-coded map allows users to quickly understand the distribution and frequency of sightings.
- **Scalable Visualization:** As users filter the data, the map dynamically recolors the polygons to reflect updated sighting patterns, providing immediate visual feedback.

2.1.3 Interactive Map Filtering

With 16,673 pentads in the dataset, filtering is essential for managing data. The app allows users to filter bird sighting data by month, significantly reducing the dataset to a manageable size while maintaining interactivity.

Why it's effective:

- **Time-Based Filtering:** Birds migrate seasonally, and sightings vary by month. The app allows users to focus on specific months, making the data more relevant and easier to interpret.
- **Interactive Scalability:** Filtering by time and species reduces the complexity of rendering the data, allowing the system to handle large datasets efficiently and update the map in real-time.

2.1.4 Efficient Data Rendering

Rendering polygons for 16,673 pentads and over a million bird entries is computationally intensive. The app uses the Google Map widget to render polygons dynamically, processing bird sighting data in small batches to avoid performance bottlenecks.

Why it's effective:

- **Real-Time Data Processing:** The app only processes and renders polygons relevant to the current view or filter, keeping the performance optimized.
- **Performance Optimization:** By processing and rendering polygons in batches, the app remains responsive, even when dealing with large datasets.

2.1.5 Conclusion

These algorithms enable the HeatMap feature to efficiently manage and visualize over 1.4 million bird sightings across 16,673 pentads. By dynamically loading, filtering, and rendering data, the app delivers a responsive and scalable user experience, leveraging data science to provide actionable insights on bird populations and conservation.

2.2 Data Science behind the BirdMap

The BirdMap feature handles a large dataset consisting of **16,673 pentads** (geographic regions) and over **1.4 million bird entries**. The following algorithms ensure efficient data management, visualization, and interactivity on the map.

2.2.1 Dynamic Location and KML Data Loading

The `_getCurrentLocation` function dynamically centers the map based on the user's current location, while the `_loadKmlData` function loads KML data that represents polygon shapes for provinces. This approach allows the app to manage large datasets and visualize geographic information in real-time.

Why it's effective:

- **Data Efficiency:** By dynamically loading polygon data for specific provinces, the app minimizes the amount of data that needs to be processed, ensuring that only the relevant information is displayed.
- **Scalable Processing:** Instead of attempting to load all 16,673 pentads or 1.4 million bird entries at once, the app fetches data on demand, optimizing performance and memory usage.

2.2.2 Data Filtering by Province and Month

The map provides filtering options based on province and month, allowing users to narrow down the data to specific geographic areas and time frames. This significantly reduces the amount of data the app needs to process and display.

Why it's effective:

- **Interactive Data Reduction:** Filtering helps manage the large dataset by focusing on a specific province or month, making the data more relevant and easier to handle.
- **Efficient Querying:** By querying the backend for bird sightings based on the selected filters, the app reduces the need to handle unnecessary data, improving both performance and responsiveness.

2.2.3 Polygon-Based Data Visualization

The polygons representing each pentad are color-coded based on bird sighting reporting rates. This visual approach transforms the large dataset into an easy-to-understand map, where users can quickly identify areas with high or low bird sightings.

Why it's effective:

- **Visual Representation of Data:** Color-coded polygons provide an intuitive way to understand bird sighting distribution across thousands of pentads, transforming over 1.4 million entries into a simple visual format.
- **Scalable Visualization:** As the user zooms in or filters the data, polygons are updated in real-time, ensuring the map stays relevant and responsive.

2.2.4 Real-Time Map Interaction

The app leverages Google Maps for real-time interaction, allowing users to zoom, pan, and click on polygons to get more information about bird sightings in specific regions. Each polygon represents a pentad, and tapping on it provides detailed data about birds in that area.

Why it's effective:

- **Real-Time Data Updates:** Users can interact with the map to view specific bird data, with the polygons and information updating in real-time based on user actions.
- **Efficient Event Handling:** The app only updates the relevant portions of the map (such as polygons) based on user input, maintaining high performance even with large datasets.

2.2.5 Conclusion

The BirdMap feature efficiently handles over 1.4 million bird entries and 16,673 pentads using dynamic data loading, filtering, and polygon-based visualization. These algorithms ensure that the app remains responsive, interactive, and scalable, providing users with real-time insights into bird populations and sightings.

3 Explanation of Algorithms in LifeList

The LifeList's algorithms are geared towards performance ensuring that the user always has access to the data with minimal delays, ensuring that it the App remains efficient and responsive.The SQLite database that is used has also been designed to be scalable whilst still maintaining efficiency.

3.1 SQLite Tables

The LifeList database is comprised of 3 tables Birds, allBirds and Provinces

3.1.1 AllBirds Table

The allBirds table stores all the data for the 800+ birds in South Africa.Images are also store as a Blob of base64. The user also has the ability to request for this table to be updated giving the latest data when they require it.

Why it's effective:

- **Offline access:** By keeping this table the user is able to access all information about all the birds in South Africa at anytime regardless of internet access.
- **Better response time:** Having no need to constantly make request to the online API means that the information is always readily available and able to be displayed.
- **Lower storage requirements:** By storing images as a string of base64 we drastically decease the size of the database
- **Redundancy:** All birds also stores the online URL for the images this means that should the image fail to load the online image is retrieved and stored in the table for next usage.

3.1.2 Bird Table

The `Bird` table stores rudimentary bird data for birds that have been seen.

Why it's effective:

- **Saves storage:** By keeping this table small we minimize storage requirements.
- **Seamless Integration:** As the `Bird` stores minimal information it make it easier to upload to the users online account allowing for the transfer of data to be quicker and more cost effective.

3.1.3 Provinces Table

The `Provinces` table stores the Bird is and a list of provinces that the bird is found in.

Why it's effective:

- **Efficiency :** This table allows for 1 query to be made to find number of birds in a province and which birds are found in which provinces.
- **Storage Efficiency:** This allows the `AllBirds` table to be made smaller and faster.

3.1.4 Online Profile Update

The `fetchUserLifelistString` retrieves the necessary data from the `Birds` table that is need for online profile storage.

Why it's effective:

- **Efficiency :** This allows for efficient retrieval of the life list to be stored online .
- **Storage Efficiency:** This allows the online storage to be smaller and more manageable.

3.1.5 Duplicate inserting

The `isDuplicate` checks if the bird to be inserted is in the life list already.

Why it's effective:

- **Efficiency :** This prevents duplicate data from even being attempted to be inserted and it allows the front end to identify if a bird is in the Life List without being queried.

3.1.6 Bird Provinces

The `getBirdProvinces` return all the provices that a bird is in.

Why it's effective:

- **Efficiency :** Allow for 1 query that retrieves all provinces that a bird is seen in.

3.1.7 Achievement Progress

The `precentLifeListBirds` returns the percentage of the current achievement. By querying `Provinces` and `Birds` tables we are able to calculate the percentage of the achievements.This query is called upon inserting a bird into life list and the value is stored in the user model.

Why it's effective:

- **Efficiency :** Allow for 1 query that retrieves the current percentage.
- **Seamless Integration :** By storing this data in the user model it is always available when need so loading time is minimized.

3.1.8 Overall Design

These tables and algorithms ensure that the LifeList as well as all bird information is available at all times. By loading the LifeList as the app starts we ensure that the data is always readily available to the user anytime the user needs it. These algorithms are also geared to minimize loading times and save storage space by breaking the data down into smaller and more manageable tables.