ZurichWoof: Dog Recommendations

Kai Lo, Chong Wan

April 7, 2024

1 Github Link for Code Sample

`https://github.com/Chong-source/ZurichWoof-Dog_Recommendation`

2 Introduction : ZurichWoof

Problem Description : In today's digital age, personalized recommendations have become crucial for enhancing user experience and engagement. Our program provides recommendations for prospective dog owners to find the right type of dog for them based off of their demographics and preferences.

By analyzing key attributes such as age, gender, and location, our system internally matches the user with other users with similar profiles to determine common dog breeds with similar ownership experience. The data set of dog owners created by Open Zurich, the official open dataset that the Zurich city created to let the public analyze more data about their city.

By asking the user for their personal preferences for their prospective dog ownership (such as various personality aspects of the dog) we can further ensure a good match between pet and owner.

Context: The context of this problem is we came across this fun and unique data set (Dogs of Zürich) that intrigued us. We wanted to do computations on this dataset and see how must feature we can develop using just the dog's information and owner's information.

Motivation for the project: Research has shown that matching the lifestyle and personality of the owner and pet leads to a longer and healthier relationship between the pet and the owner [1]. Thus, we want the owner of the dog to make informed decisions when purchasing a dog in Zürich by choosing the suitable district to live in for them and their dogs. Moreover, matching their preferences with the correct breed.

Project question: What are the top 5 districts that has the highest proportion of a each dog breed in Zurich and how can we make various customized suggestions to potential dog owners using the dataset?

3 Data Sets

1. **Dogs of Zurich - www.kaggle.com/datasets/kmader/dogs-of-zurich**[2]
   (a) file name - `zurich_dog_data_2017.csv`.
   (b) This file contains the dog and dog owner's data in Zürich in 2017. It includes the dog's age, breed, and the gender of the owner, age of the owner, and the districts where the owners reside.
2. **Dog breeds - https://www.kaggle.com/datasets/sujaykapadnis/dog-breeds** [4]
   (a) file name - `breeds_traits.csv`
   (b) Other than analyzing the trends of dogs, we cleaned up this data set from the American Kennel Club to store more information about each dog and the dog breed to give a more customized suggestion for the potential dog owner.
   (c) This data set contains information about the dog's personality, barking level, openness to strangers ... etc.

3. **City Quarters - https://www.bfs.admin.ch/bfs/en/home/services/geostat/swiss-federal-statistics-geodata/administrative-boundaries/boundaries-urban-quarters-swiss-cities.html**[3]

   (a) file name - `district_quarters_2017.csv`.

   (b) We used the 2017 version because it matches the districts that were in place for the Dogs of Zürich dataset, and that data set was discontinued. So we wanted to use an accurate corresponding dataset to match the district information.

4. **City Quarter Distance - Generated Data using Google's Distance Matrix API based on the City Quarters' file**

   (a) file name - `district_closeness_2017.csv`

   (b) We used the Google map's matrix API to calculate the distance between 2 districts. Inputting the names of the districts to the API was enough for us to make a web request to the API and for it to return the relevant information to us in JSON format.

   (c) We used the requests package from pycharm to make and receive the requests to the Google API.

   (d) We also wrote our methods to generate the CSV file using these data, with each district's distance from each other separated by the vertical line character.

   (e) One problem we ran into during the data collection was that characters such as ü and ö for entries didn't show up correctly in the Google API searches, thus we manually replaced those characters.

   (f) Data Cleaning: We ran into both value error and key error because of specific names such as (Hard, and Zurich) that are too obscure for Google Maps API, so we manually searched up the location and input a more common name for the location.

   (g) We had 6 entries out of 1156 entries that had value error due to the Google Maps API unable to recognize the place name, so we manually calculated them.

5. **District latitude and longitude - Generated Data using Google's Geocode API based on the City Quarter's file**

   (a) file name - `district_lat_lng.csv`

   (b) We used Google's Geocode API to get the longitude and the latitude for each of the districts so we can display their geo-locations on an interactive map using tkinter after we have found the top 5 most similar districts according to the user's age, gender, and dog breed.

6. **Translated dog breed - Used googletrans package to translate dog breed's name in German to English**

   (a) file name - `translated_dog_breed.csv`

   (b) We used googletrans package to translate the dog names that were in German to English

   (c) For each row of the csv file, it follows the format [dog breed's name in German], [dog breed's name in English].

7. **Dog breed images - used google's custom search API to find image links to dog breeds**

   (a) file name - `dog_images.csv`

   (b) We used google's custom search API to webscrape for the first image that pops up when we search for the dog breed.

   (c) For each row of the csv file, the file is in the format [dog breed's name in english], [the link to the first search result on google images]

4 Computational Overview

Our project is split into two main parts: Demographic recommendations where we ask the user personal questions such as their age, gender, and address city district) to find similar users and reference their dogs, and preference recommendations where we ask the user for their personal preferences regarding the dog's attributes and personality to match them with the corollary dog breeds.

4.1 Demographic Recommendations

This recommendation system is based off of the Open Zurich dog owner data set: a large list of users within Zurich and the dogs they own. We are concerned with the following information: The age of each owner, the gender of each owner, which city district the owner lives in, and the breed of their dog. These information are taken from the file `zürich_dog_data_2017.csv`.

Then, we construct a graph with vertices for each dog owner and vertices for each dog breed, with edges between them representing ownership. More specifically, each owner is represented by a custom User class that also contains information about their demographics. We then use the User class's `compare` method to which owners are most similar to our user, and then traverse the graph to find the most popular dog breeds by user similarity.

A second (weighted) graph is also constructed with vertices for each city district and dog breed, with weighted edges between them representing the number of dogs of that breed type in the given district. This is used to find the districts with the highest proportion of a specific dog breed when recommending the user potential places to go to find similar dogs to their own. Combined with the user demographic data, the main function for this calculation is `get_demographic_recommendations`.

Graphs play an important role in this project because we used them to make predictions such as the relevant age of a dog owner and what types of breed might be good for a person with a similar age. We also connected the compatible breeds, representing the dogs as vertices and the compatibility of dogs with other dogs if the compatibility score is greater to a certain extent.

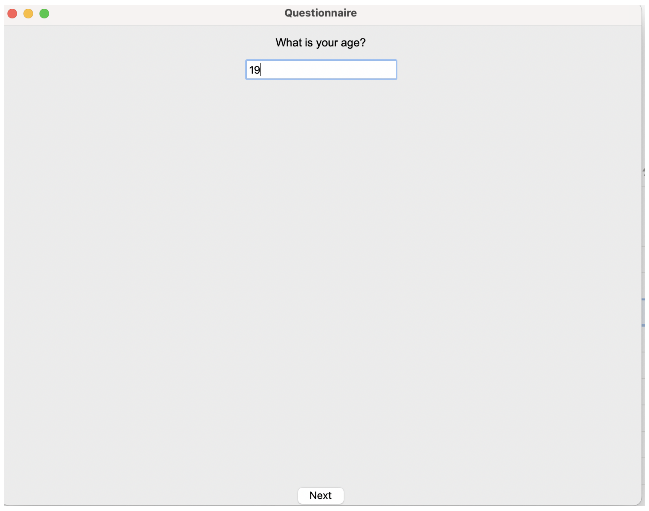
4.2 Preference Recommendations

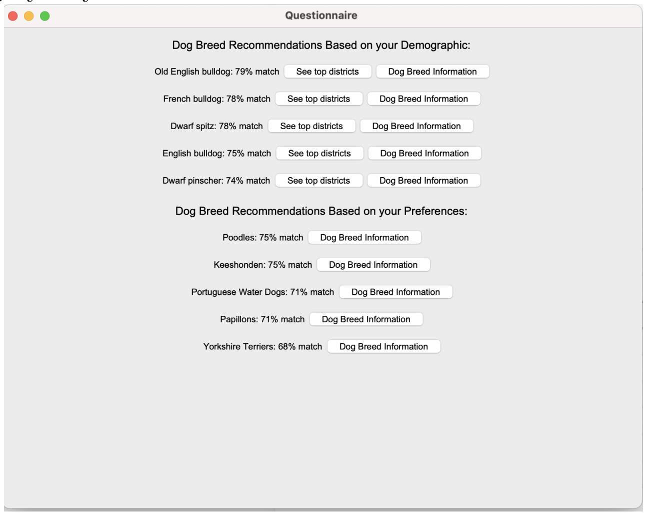
Our data for the preference recommendations is based off the American Kennel club's descriptions of dog breeds. For each dog breed, it contains the information about the dog's energy level, shedding level, mental stimulation needs, protective nature... etc. Each of these characteristics are ranked in a score out of 5. We're using the data from the file `breed_traits.csv` to rate the dog breeds.

The major computation this part of the program performs is a decision matrix. The main function for this algorithm is `weight_raw_preference_data` where the raw weighted score is calculated for each dog breed, and `normalize_preference_recommendations` where the raw score is processed to fit the scale from 0.0 to 1.0 for the data to be represented as a percentage of how much the dog matches the user. Holistically, the users input the weighting that they would give to each criteria, and indicate whether they consider a characteristic the dog has to be negative or positive for them. Then, the decision matrix will multiply the weighting of the criteria by the score that the dog breed's score and sort the the list reversed so that we can extract the top 5 dog breed according to this user's preferences.

4.3 Interactive User Interface

The user interface of this program is an interactive questionnaire where the user can click on the question boxes and input in their preference for the weighting of each category as you can see below.





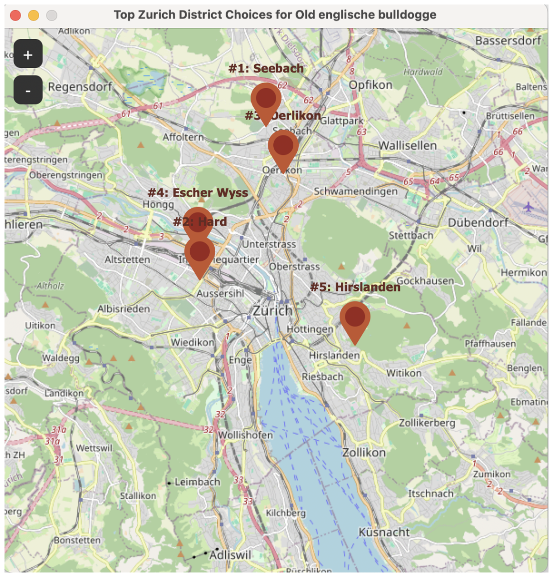
Dog Breed Recommendations Based on your Demographic:

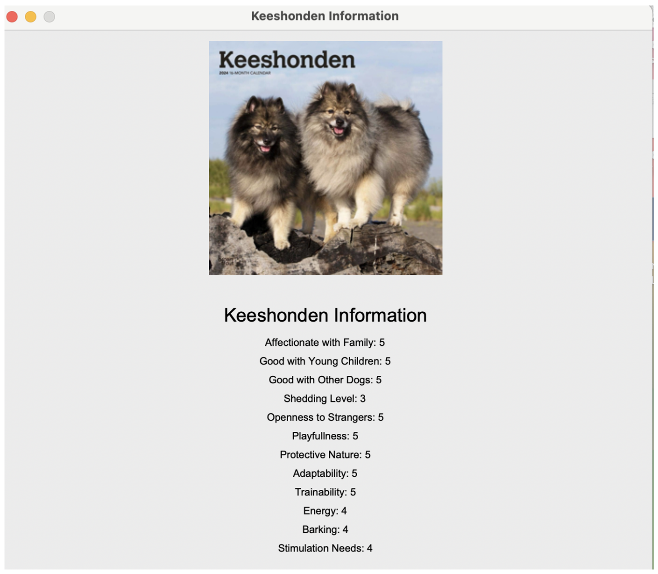
Old English bulldog: 79% match
French bulldog: 78% match
Dwarf spitz: 78% match
English bulldog: 75% match
Dwarf pinscher: 74% match

Dog Breed Recommendations Based on your Preferences:

Poodles: 75% match
Keeshonden: 75% match
Portuguese Water Dogs: 71% match
Papillons: 71% match
Yorkshire Terriers: 68% match

Then, after a series of questions about the user's demographic and preferences, a result page will show up. Firstly, it will show you a list of dog breeds, and besides these dog breeds, we have the matching percentage score, a button where you can click on for the top 5 districts of dog population most similar to your dog, and a button where you can see the dog's information, which is a pop up window that contains the dog's image and the dog's characteristics.




Keeshonden Information

Keeshonden Information

Affectionate with Family: 5
Good with Young Children: 5
Good with Other Dogs: 5
Shedding Level: 3
Openness to Strangers: 5
Playfullness: 5
Protective Nature: 5
Adaptability: 5
Trainability: 5
Energy: 4
Barking: 4
Stimulation Needs: 4

4.4 Python Libraries

Explain how your program uses Python libraries to accomplish its tasks. Refer to specific functions, data types, and/or capabilities of the library that make it relevant for accomplishing these tasks.

- **Tkinter:** Tkinter is utilized for creating an interactive user interface, allowing users to input preferences, create pop-up windows, and creating buttons to move on to the next page.
- **Requests:** The Requests library is used to make HTTP requests to the Google Maps Matrix API, Google Custom Search API, and Geocode API for distance calculation between districts, find pictures of dog breeds, and find the geolocations of districts, respectively.
- **googletrans:** The googletrans library is used to translate german words to english. This library uses google's google translate API. The main usage was the `tanslator.translate()` to translate german into english.
- **tkintermapview:** The tkintermapview is a small library that displays an interactive map where the user can plot location markers, zoom in, and zoom out. The main usage was using the `TkinterMapView` to create a map object and `set_marker` to place location markers.
- **pillow:** The pillow library was used to display images of dog breeds.

4.5 Project Layout

4.5.1 Data Generation

The following files contain functions used to generate just a few of our 7 datasets (the others from online sources). This is necessary so that we can pre-process the API information and ensure that the data makes sense before using it in the program.

1. **district_closeness:** This file generates CSV data on how close different districts are to each other using the Google Maps API. It inputs their names and finds their distance in KM.
2. **dog_images_translation:** This file generates CSV data of translations of dog breed's names from German to English (with Google Translation API) and finds links to images of the dog breeds using Google Search API.
3. **zurich_map:** (Among other things) this file generates CSV data on the latitude and longitude of districts in Zurich using the Google Geocode API.

4.5.2 Data Processing

The following files are used to process and contextualize the data we have loaded using given user demographics and preferences.

1. **districts:** This file has the District class that just stores information on districts such as their name, ID, and distances to other districts.
2. **graphs:** This file contains general implementations of both ordinary Graphs and WeightedGraphs for us by the demographic recommendation system.
3. **user_demographics:** This file has functions for taking in user demographic information and dog graph information to create user-tailored recommendations
4. **user_preferences:** This file has functions for taking in user preferences on dog attributes and dog breed information to create user-tailored recommendations
5. **userdata:** This file contains the User class that is used to calculate a quantitative similarity between different users

4.5.3 Main Program

1. **data_loader:** This file loads data from all 7 of our data sources and do a little bit of processing. This includes things like loading Zurich dog information, loading district data, loading + processing district distances, loading dog attributes from Kaggle dataset, loading district geo-location information, loading translations from German to English, and loading image links of the dogs.
2. **main:** This file contains the largest amount of logic pertaining to displaying the UI to the user and processing user information using the other data processing files.

5 Instructions

Note:

1. Please ensure internet connection is good while running this program because when displaying image of the dogs, the program fetches images from the internet using links to the image address.
2. Please be patient with the loading time of the map and the dog breed information images, as they can be a bit slow sometimes.
3. Download all of the files submitted to MarkUs onto the computer, open the zip file in a new folder.
4. Install all requirements: pip install -r requirements.txt
5. The zip file of the data sets is also uploaded to Markus, and its name is `data.zip`. Unzip this file, and please ensure that the name of the folder is just 'data' because we're calling using that specific file path.
6. Run the main.py file.
7. The description of the user interface and what you should see is covered in section 3.3
8. below is a sample data you can run.

Sample Input after the program starts:

1. What is your age? - 40
2. What is your gender? - Female
3. What is your district? - Affoltern
4. Input all 5s for preferences, and choose positive trait for both high amount of barking and high needs for mental stimulation.
5. After answering all questions, you should see the interface as shown in section 3.3.

6 Changes to Project plan

- Originally, we wanted to display two separate results from the Zürich data set: one showing a map that contains the trends of the dog breeds, and one interactive questionnaire that makes dog recommendations for the user.
- However, after consideration, we recognized the combining the two components of this program together will make the interface more coherent. Thus, we combined the map element into the questionnaire, showing the top 5 districts where the user can find dogs most similar to their own dog. This changes makes the program more useful because rather than showing the most popular dog per district, we're showing useful information that can be used by the users to make the best judgment of where to live and where to socialize their dogs.

7 Discussion

7.1 Achieving our Goal

The computation undertaken in our project aimed to provide personalized dog breed recommendations to potential dog owners in Zurich. The results of our program effectively achieved this goal by:

1. Calculating compatibility scores between users and dog breeds based on weighted criteria of both demographic and personal preferences of the user.
   (a) By taking into account the user's demographic, the program does calculations using a weighted graph structure that gives the most similar user compared to this specific user. Then, we calculated what dog breeds similar users had to reach a conclusion of what dog breed the user would most likely to be compatible with.
   (b) By taking into account the user's preferences of their life style, we have a second recommendation system based on user's preferences. This recommendation system is built to give the user more options of dog breeds to choose from, and making up for the possible over fitting of data and the problem of small sample size from demographic recommendation.
2. We visualized these recommendations on an interactive map interface, allows users to explore trends and insights about dog ownership in Zurich. Moreover, the interactive and simple interface makes the program user friendly and efficient.

7.2 Limitations and Obstacles:

- Translating German into English was difficult because the dog breeds have very specific names, and translating them to English didn't give the correct terms, therefore, we manually changed some translations that were wrong.
- API Constraints: Dependence on external APIs like Google Maps Matrix API introduced limitations in terms of usage restrictions and data accuracy, for example when we're using the google's custom search API, for some of the entries, the image link didn't show up, so we manually added the links.
- Tkinter (UI Library) had a lot of constraints of running multiple windows together, making it difficult for us to display our results in pop-up windows. A website-based implementation would likely be more suitable for this project (but we cannot do this in python).

7.3 Next Steps for Further Exploration:

To further enhance our project, we could consider the following next steps:

- Integration of Machine Learning: Implementing machine learning models to predict dog breed preferences based on user demographics and preferences could improve recommendation accuracy. (Also our TA's recommendation)
- Real-Time Data Updates: Implementing mechanisms to continuously update and refresh dataset information to reflect real-time trends and changes in dog ownership patterns.

- Volume of Data: Since we restricted our analysis to a particular city, our results apply primarily to Zurich users, so furthering our analysis to other countries and a larger dataset would help in extrapolating our results.
- User Feedback Incorporation: Incorporating user feedback mechanisms to refine and improve the recommendation system based on user interactions and preferences and further adding more features such as an AI chat bot to take the recommendation system a step further.

7.4 Conclusion

In conclusion, our project successfully addressed the research question of providing personalized dog breed recommendations to potential dog owners in Zurich. Despite encountering limitations and obstacles, we employed computational strategies and algorithms effectively to achieve our project goals. By analyzing and interpreting the results, we gained valuable insights into the complexities of dog ownership trends and user preferences. Moving forward, further exploration and refinement of our project could lead to even more accurate and user-centric recommendations in the future and apply it to a more general dataset once we combine more data together.

References

[1] Rachel A. Casey et al. "Reasons for Relinquishment and Return of Domestic Cats (Felis Silvestris Catus) to Rescue Shelters in the UK". In: *Anthrozoös* 22 (Dec. 2009), pp. 347–358. DOI: `10.2752/089279309x12538695316185`.

[2] Scott Mader. *Dogs of Zurich*. www.kaggle.com, Mar. 2017. URL: `https://www.kaggle.com/datasets/kmader/dogs-of-zurich`.

[3] Federal Statistical Office. *Boundaries of urban quarters of Swiss cities*. www.bfs.admin.ch. URL: `https://www.bfs.admin.ch/bfs/en/home/services/geostat/swiss-federal-statistics-geodata/administrative-boundaries/boundaries-urban-quarters-swiss-cities.html` (visited on 04/04/2024).

[4] Maryna Shut. *Dog breeds*. www.kaggle.com. URL: `https://www.kaggle.com/datasets/marshuu/dog-breeds`.