**Project Title**

Travel Destination Recommendation System (TDRS)

**GRP: 1**

**Group Members’ Names and ID**

LONG ZHEN A0297168L

ZHOU YUKANG A0296841R

An DONGQI A0296362W

TAO ZHEN A0296066U

**Introduction**

In today's digital age, travelers are constantly seeking personalized experiences to make their journeys memorable. The Travel Recommendation System (TRS) aims to address this demand by leveraging advanced algorithms and data analytic to provide customized travel suggestions. This system analyzes various data points from public dataset, including weather, cost of living index, and city type to offer tailored recommendations for destinations. By integrating existing evaluations and machine reasoning techniques, the TRS adapts to deliver more accurate and relevant suggestions over time. Besides to choosing from preset tags, our system supports natural language input by users. This function enable users to describe the expect destination to include more specific and personalized cases. This project not only enhances user satisfaction but also helps travel businesses optimize their offerings, ultimately contributing to a more engaging and efficient travel planning experience.

**Market Context**

The travel industry has been undergoing rapid digital transformation, especially with the growing demand for personalized experiences. Travelers today expect tailored suggestions for destinations, accommodations, activities, and restaurants based on their preferences, budgets, and past experiences. This shift has created a significant market for travel recommendation systems.

Enabled by the machine reasoning system and big data technology, travel recommendation systems can now analyze vast amounts of information from various sources (such as reviews, social media, and user behavior) to provide highly personalized recommendations. The travel industry is increasingly adopting such tools to attract and retain customers and propose the best fitted experiences for travelers with evolving needs.

**Market Research (Existing Travel Recommendation Software and Platforms)**

Existing TRSs could be summarized into two categories – booking oriented TRS and the mixed TRS. For booking oriented TRSs, we have *Airbnb* (Offer local experiences, tours, and recommendations for things to do in specific locations), *Skyscanner* (A flight comparison tool but also offers recommendations for hotels, car rentals, and travel itineraries), and *Booking.com* (Offers a wide variety of accommodations, from hotels to vacation rentals, along with recommendations for restaurants, activities, and landmarks). These systems provide specific options for transportation, hotel, and restaurants booking. Some of these, such as Airbnb, also provides local tour plan. However, these TRSs merely demonstrate the possible choices. Users still struggle to decide the destination and the plan to pick.

The other category is the mixed TRS. *TripAdvisor* (One of the largest and most popular platforms for travel recommendations. It provides reviews, travel-related content, and suggestions for hotels, activities, and restaurants based on user feedback and ratings) and *Google Travel* (offers trip planning, including recommendations for hotels, flights, attractions, and itineraries based on user data) are two leading systems in this category. These mixed TRSs provides the whole travel plan along with the transportation and hotel recommendations. On the other hand, large-scale platforms disturbed by their magnitude. The overloaded content confuses the users, the travelling plan might not be personalized enough (prioritize sponsored content), and the data shared protocol brings the privacy concerns to users.

To address these problems, we aim to develop a destination-focused recommendation system based on the existing tour evaluation dataset. Satisfaction and personalization are the main performance indicators for our systems. Only adopting existing dataset will not bring any privacy concerns to users. Our system provides:

1. Accurate filtering
2. Personalized recommendations
3. Privacy protection

**Project Scope**

We aim on providing the most accurate and personalized destination recommendations system according to users’ preference and historical travel patterns (if any). Knowledge based reasoning system (knowledge graph/model trained by datasets), big data mining (recommending system), and cognitive techniques (supporting natural language input) are the main approaches to achieve the system. Due to the limited time and scale, we will not integrate the modules that requiring third-party authorization, such as the real-time transportation booking and hotel rental service.

**Data Collection and Preparation**

1. Global Air Pollution Datasets

**Air Pollution** is contamination of the indoor or outdoor environment by any chemical, physical or biological agent that modifies the natural characteristics of the atmosphere. Household combustion devices, motor vehicles, industrial facilities and forest fires are common sources of air pollution. Pollutants of major public health concern include particulate matter, carbon monoxide, ozone, nitrogen dioxide and sulfur dioxide. Outdoor and indoor air pollution cause respiratory and other diseases and are important sources of morbidity and mortality.

1. World's Best Cities for People and the Planet

This dataset contains the index, from global design firm Arcadis and the Centre for Economics and Business Research, ranks cities’ success based on social, environmental, and economic factors.

Arcadis used 32 indicators and a cross section of the world’s urban areas, so not all capitals or large cities are necessarily represented. A city is scored on each of the three sustainability factors; its overall score is the average of those.

1. Cost of Living

The **Mercer Cost of Living City Ranking** is an annual survey that compares the cost of living in cities worldwide. The **Numbeo** is another comprehensive datasets that record the through out cost of living index from continuous years.

1. OpenWeather

**OpenWeather** is a platform that provides weather data and forecasts through Application Programming Interfaces (API) for developers, businesses, and organizations. It aggregates and delivers comprehensive weather information from multiple sources, including meteorological stations, radar, and satellite data, allowing users to access real-time weather updates, historical data, and forecasts.

**System Design:**

**Implementation:**

**Chat-bot:**

1. Text Preprocessing

Standardize user input by performing lowercase conversion, abbreviation expansion, punctuation removal, lemmatization, and spell correction. These steps ensure consistency of user input for further processing. We do not remove all the stop words, because they are useful in later semantic analysis.

1. Keyword Detection

Predefine keywords, including “city”, “cost”, “transport”, “temperature”, “climate”, and “air”. They are converted to word vectors using a spacy model.

The chatbot extracts nouns from the user’s input and compares them with the keywords based on semantic similarity. If the similarity exceeds a threshold, the word is considered relevant, and the chatbot attempts to extract adjectives that modify these nouns for more detailed information.

1. Context Analysis

Use spacy to get the semantic structure of user input. We divide it into two types:

* 1. Adjective + noun: In this case, the relationship between key words and adjectives is “amod” or “compound”. Then we add those adjectives to the list of corrosponding key words.
  2. Subject + linking verb + adjective: When a noun is identified as the subject, it checks the associated linking verb to find adjectival complements and negations. This helps capture descriptive information, such as “not” and “hot” from the sentence “The city is not hot”.

1. Conversation Management

User can end the conversation by typing “Bye” or “Enough”. The chatbot will analyze the input to identify missing information and prompt the user to provide additional details until it has gathered sufficient data.

**Matching Algorithm:**

The process is divided into two major components:

1. Semantic Analysis:

The user input is processed through semantic analysis, where several key factors are extracted, including: City Type, Temperature, Cost, Transport, and Air Quality. Each of these features is sent for SBERT (Sentence-BERT) Encoding, which transforms the input into a numerical representation for comparison.

Once encoded, Cosine Similarity is calculated to determine how similar the input data is to predefined categories. This is used in the Category Matching process, where relevant city types or categories are identified.

1. Vector Processing:

This stage involves transforming city features (e.g., temperature, cost, etc.) into vectors. The steps include: Vector Transformation, Encoding Categorical Features, and Scaling Numerical Features. These features are then concatenated into vectors for further similarity comparison.

Two similarity calculations are performed:

X1 Similarity: Based on the SBERT-encoded user input and its semantic analysis.

X2 Similarity: Based on the vectorized city data and user features. The two similarity scores are combined into a Final Similarity Score, which is used to sort and select the top cities most relevant to the user’s input.

1. Specific Calculating Process:

For X1 Semantic Similarity Calculation:

1. User Input Text is processed using SBERT Encoding.
2. The encoded input is compared to predefined category encodings, and Cosine Similarity is calculated for each dimension (e.g., temperature, cost).
3. For each dimension, the maximum similarity is selected, and a normalized average is calculated. This represents the similarity between the user's preferences and the categories of interest.

For X2 Feature Vector Similarity Calculation:

1. User Features (derived from the user’s input) and City Data are vectorized.
2. Once vectorized, Cosine Similarity is computed between the user’s features and the city features.
3. The similarities are then normalized for consistency across different feature scales.

Finally, both X1 and X2 scores are combined in the Final Similarity Calculation, where a weighted average is applied to produce the overall similarity score for each city. This final score helps in ranking cities according to how well they match the user’s preferences.

**Front-end Detail (by Vue API):**

1. Collect User Options:
2. The front-end gathers user options or preferences from the input interface.
3. These collected options are then sent to the back-end server via an API call.
4. Collect User Messages:
5. The front-end collects user messages or queries, which are also sent to the back-end through API requests.
6. Receive Robot Reply:
7. The replies generated by the chatbot on the back-end are received by the front-end.
8. These replies are then displayed to the user in the chat interface.
9. Receive Recommendation Options:
10. The front-end receives recommendation options from the back-end.
11. Similar to the robot replies, these recommendations are displayed to the user, assisting them in making informed decisions.

**Back-end Detail (by Django API):**

1. Receive User Options:
2. The back-end receives user options sent from the front-end.
3. It updates the city information based on these options to tailor the chatbot’s responses and recommendations.
4. Receive User Messages:
5. User messages from the front-end are received and used to update city information.
6. This helps the back-end maintain an accurate context for generating meaningful responses.
7. Generate Robot Reply:
8. The back-end processes the incoming data and generates a suitable reply from the chatbot.
9. This reply is sent back to the front-end to be displayed to the user.Generate Recommendation Options:
10. Based on the processed user input, the back-end generates recommendation options.
11. These recommendations are sent to the front-end to guide the user in their decision-making process.

**Results(可以放图片来展示):**

**Challenges and Roadblocks:**

Data Collection:

1. The Missing Value: We only extracted intended columns from one dataset and combine them together. It caused missing value when the row number of two datasets is different after merging. For example, the living cost dataset has 226 cities while the City type dataset contains only 100 cities. To solve this problem, we try to use the data from other resources (government official report, Google Place API, e.t.c) to make our own evaluation according to the same evaluation criterion of the original dataset. We cannot yield exact same index as original dataset, still these index are reasonable and usable.
2. The Limited Parameter: In this project, we design a system in a relatively small scale and use the vector match algorithm to find the most appropriate result. However, the vector match algorithm we adopted will be affected if the input parameter is too large. Therefore, we try to use the categorical data as much as possible to improve the computation speed. We adopt normalization for those numerical data first, and then convert them into 3 different categories. Finally, we yield a smaller dataset and keep the algorithm fast.

**Future Work:**

Admittedly, this system could be improve in some aspects due to the limit time and resources.

The first possible improvement is the scale of the dataset. The vector match algorithm takes long time to process the calculation if there are many classes to match. Therefore, we must scarify the number of classification of each column. For example, at first we plan to use at least 7 categories to indicate the climate, but we eventually only adopt 3 instead. This cut down undoubtedly reduces the accuracy of the algorithm. In the future, one can get more specific and personalized result by enlarging the categories in the dataset.

The second improvement could be found in the demonstrating stage. Due to the limited time, we now only offer the destinations alone. In the future, we can add on some related links to supply the recommendations, including different types of related travel blog, flight/train/ship/car booking information, and local hotel information to provide the one-stop services.

**Appendix: Functionalities and Techniques Map**

The Whole Recommendation System: Big data mining techniques, RS module

The Chat-bot that supports the natural language input by users: System designed with cognitive tools, CGS

The Vector match algorithm: Knowledge based techniques in decision automation, MR