Training Dataset Creation Documentation

Data Source

We have two data sources:

a) Reference Data (`referance_rooms-1737378184366.csv`) b) Supplier Data (`updated_core_rooms.csv`)

The core objective of the API is to match room names between the reference and supplier datasets. To achieve this, a model is trained using these data sources.

Exploratory Data Analysis (EDA)

Based on initial analysis (and confirmation from prior queries), the datasets can be combined using `lp_id` . This ID functions similarly to a hotel ID, hence the datasets should be grouped by `lp_id` , consolidating all room names into a single row.

The reference dataset contains approximately 28,000 hotels, while the supplier dataset includes around 40,000 hotels. To train the model effectively, labeled data is necessary. Currently, only hotel-level labeled data is available, which must be converted to room-level labels.

This conversion process will require creating custom labels from existing data.

Creating the Training Dataset

Approaches Attempted

1. TF-IDF Cosine Similarity

TF-IDF is essentially a bag-of-words model and does not capture semantics.

2. Sentence Transformer

Sentence transformers capture semantics, but subtle differences between room names (such as 1 vs. 2, double vs. single, non-refundable vs. refundable) can lead to inaccurate similarity assessments since these differences, though subtle, are highly significant.

3. Using LLM

Attempted using an LLM model hosted on GitHub to identify matched room pairs. However, this approach was slow and produced inaccurate results.

Thought: LLMs could potentially be useful for generating training data or augmenting smaller datasets rather than direct classification tasks.

4. Hybrid Approach with TF-IDF Cosine Similarity

Process steps:

- **4.1 Text Normalization:** Lowercased, removed special characters, and reduced multiple whitespace.
- **4.2 Calculated cosine similarity** between room names within a single hotel group.
- **4.3 Selected the top half** of similar room names with similarity scores greater than 0.5.
- **4.4 Checked if reference room name** is a substring of the supplier room name.
- **4.5 Verified new words** in reference names against a predefined special filter list (words to ignore).
- **4.6 If all above conditions matched:** Analyzed the string to verify if new words appeared at the beginning or end (positions that often contain irrelevant information such as smoking or pet policy).
- **4.7 String Analysis:** Checked if replacement words exist between reference and supplier room names. Replacement words often indicate differences rather than similarity.

Note: This approach was not fully streamlined and was derived from extensive data observation. Optimizations proved insufficient, indicating a need for ML-based data generation techniques, leading to subsequent approaches.

5. Hybrid Approach with spaCy

(Currently being tested.)

- **1. Categorization with LLM:** Created a hierarchy, room type, and abbreviation set using an LLM and the available data to improve categorization accuracy, given repetitive tokens in room names.
- **2. Feature Extraction:** Features such as bedroom numbers and apartment indicators were extracted. However, these features were not robust due to the inclusion of addresses or extraneous information.
  - **2.1 Combined Features:** Merged features like `smoking`, `pet_policy`, and `bed_type` into one categorical output.
- **3. spaCy Pipeline:** Utilized a spaCy pipeline to calculate similarity using NLP parsing and Named Entity Recognition (NER), though the chosen small model lacks word vector capabilities.
- **4. Top Similar Matches:** Initially took the top two highest similarity scores to identify matching pairs. However, including second-best matches introduced incorrect negative matches, generating inaccuracies.

- **5. Custom Filters:** Added custom filters for numeric mismatches and block/floor discrepancies.
- **6. Strict Bedroom Count Validation:** Enforced strict rules for bedroom counts, explicitly defining mismatches such as single vs. double, king vs. queen, as semantic similarities alone were insufficient. This highlighted the need for fine-tuning custom domain-specific models.
- **7. Weighted Sum Scoring:** Implemented a final weighted scoring system combining semantic similarity and various custom filters.

Final Approach Adopted

I selected class 1 labels from the spaCy-based method due to robustness. Out of these, 745 room names did not form exact pairs, so additional exact pairs were included, resulting in 2000 matched (class 1) cases.

For class 0 (unmatched), I used labels from the previous method, as it considered a wide variety of non-matching scenarios. A random sample of 2000 unmatched pairs was selected.

The final balanced training dataset thus consists of 4000 labeled pairs.

Some of the example of the dataset :

example 1:

---------- supplier - rooms ----------
Superior Double Room
Superior Room, Fireplace
Deluxe Double Room
Standard Double Room
Standard Double or Twin Room
Standard Twin Room
---------- reference rooms ----------
Deluxe Queen
Deluxe Queen Room with Extra Bed
Standard Queen
Standard Queen Room with Single Bed
Standard Double
Standard Double with Extra Bed
Standard Twin
Family Room with Private Bathroom
Quadruple Room with Bathroom
---------- result----------
['standard twin room', 'standard twin', 0.8490390520071958, 'lpab7f5']
['standard double room', 'standard double', 0.8503281582595854, 'lpab7f5']
['standard double or twin room', 'standard double', 0, 'lpab7f5']

['superior room fireplace', None, 0, 'lpab7f5']
['superior double room', None, 0, 'lpab7f5']
['deluxe double room', None, 0, 'lpab7f5']

example 2:

Deluxe Double Room
Deluxe Double Room, City View
Family Studio Suite, City View
Deluxe Quadruple Room
Superior Quadruple Room, City View
Family Studio Suite
---------- supplier - reference down ----------
Family Studio Suite
Deluxe Double Room
Deluxe Double Room, City View
Deluxe Quadruple Room
---------- result----------
['superior quadruple room city view', 'deluxe double room city view', 0.6889000248061357, 'lp65570ecf']
['family studio suite city view', 'family studio suite', 0.8571983955209816, 'lp65570ecf']
['deluxe double room city view', 'deluxe double room city view', 1.0, 'lp65570ecf']
['deluxe double room', 'deluxe double room', 1.0, 'lp65570ecf']
['deluxe quadruple room', 'deluxe quadruple room', 1.0, 'lp65570ecf']
['family studio suite', 'family studio suite', 1.0, 'lp65570ecf']