# **Vendor Qualification System (Peter Le)**

## **Required Libraries**

pip install pandas pip install scikit-learn pip install flask pip install requests

#### **File Overview**

main.py - ingests data and scores it tests.py - runs tests among different categories

### main.py

#### Task 1

- Loads vendor dataset and turns it into pandas dataframe.
- Assigns a custom category column to the right of the dataframe.
  - Considers descriptions with keywords to classify different categories.
- Combines features and description into a singular column to be vectorized.

#### Task 2

- Computes TF-TDF vectors using TfidfVectorizer library.
  - Gives numerical weights of vendors using popular vectorization technique.
- Converts user query into query vector.
- Computes cosine similarity between query and vendors.
  - Angle between two vectors determines correlation between the vectors.
- Apply threshold filter of 0.01 (best threshold out of experimentation).
  - If no results appear, set the threshold to 0.

#### Task 3

- Compute a composite score.
  - Apply a 0.7 rating weight and 0.3 similarity weight (popular heuristic weights that value rating).
- Sort vendors by score (descending).
- Return top 10 vendors

#### Task 4

- Created flask framework.
- Endpoint POST /vendor\_qualification.

- Used port 8000 (5000 did not work with my macbook).

# **Testing**

Two Ways to Test

- 1. python tests.py
  - a. Sends HTTP Post requests to Flask API with requests library.
  - b. Sends post request with test inputs
  - c. Parses json response
  - d. Returns number of results with vendor names, classified custom categories, score, and similarity

e.

- curl -X POST http://localhost:8000/vendor\_qualification -H "Content-Type: application/json" -d '{"software\_category": "Accounting & Finance Software", "capabilities": ["Budgeting"]}'
  - a. 8000 instead of 5000

```
"product_name": "Solid Performers CRM",
"main_category": "crm software",
"custom_categories": [
"crm software",
"accounting & finance software"
                     ],
"score": 3.43,
"similarity": 0.0
                       "product_name": "Fireberry",
"main_category": "crm software",
"custom_categories": [
"crm software",
"accounting & finance software"
                        "score": 3.36,
"similarity": 0.0
                         product_name": "Breakcold",
main_category": "orm software",
custom_categories": [
"orm software",
"accounting & finance software"
                        score": 3.29,
"similarity": 0.0
                           oroduct_name": "Pipeliner CRM",
main_category": "crm software",
custom_categories": [
"crm software",
"accounting & finance software"
                             roduct_name": "Efficy CRM",
ain_category": "crm software",
ustom_categories": [
'crm software",
'accounting & finance software",
"erp software",
"hr software",
                                    duct_name": "Freshsales",
__category": "crm software",
tom_categories": [
rm software",
counting & finance software",
rp software"
                       "product_name": "FinCRM",
"main_category": "crm software",
"custom_categories": [
"crm software",
"accounting & finance software"
                       "product_name": "Kommo",
"main_category": "crm software",
"custom_categories": [
"crm software",
"accounting & finance software"
}∰
(base) peternle@Peters-MBP vendorqualification % ■
```

Challenges

b.

- Finding a fitting threshold that would guarantee results.
- Finding fitting features and experimenting with them.
- Learning Flask API
- Vectorization research

# **Potential Improvements**

- Experiment more with the rating and similarity weights.
- Docker support.
- More unit tests.
- Possible dynamic threshold.