AP LAB_1

~ Prof. Ishan Maheta

✅ Title 1: Text File Analysis for Product Reviews

🎯 Objective:

To develop a Python script that reads multiple customer review files from a directory, extracts structured data using regular expressions, computes the average rating per product, identifies the top-rated products, and writes a detailed summary to a text file.

📘 Task Description:

Consider a scenario where you are working as a data scientist for a large e-commerce company. Your team is responsible for analyzing customer feedback data, which is stored in multiple text files. Each text file contains customer reviews for different product categories. Your task is to write a Python script that performs the following operations:
Read the contents of all the text files in a given directory. For each review, extract the following information:
• Customer ID (a 6-digit alphanumeric code)
• Product ID (a 10-digit alphanumeric code)
• Review date (in the format "YYYY-MM-DD")
• Review rating (an integer between 1 and 5)
• Review text (the actual feedback provided by the customer)
Calculate the average review rating for each product and store it in a dictionary where the product ID is the key and the average rating is the value.
Determine the top 3 products with the highest average review ratings.
Create a new text file named "summary.txt" and write the following information into it:
• The total number of reviews processed.
• The total number of valid reviews (reviews with all required information extracted successfully).
• The total number of invalid reviews (reviews with missing or incorrect information).
• The product ID and average rating of the top 3 products with the highest average ratings.
Your Python script should be robust, handling any potential errors or exceptions during the file handling process.
Additionally, you should implement efficient algorithms to handle large volumes of data without consuming excessive memory or processing time.

Write the Python script to achieve the above objectives and provide detailed comments explaining each step of your implementation.

💻 Python Code:

```python
import re
from datetime import datetime
import os

def process(row: str, products: dict[str, int], ratings: dict[str, int]) -> None:
    pattern = re.compile("\"[^\"]*\"")
    row = re.sub(pattern, "", row).strip().split()
    if len(row) > 4:
        raise ValueError("Extra Values")
    if len(row[0]) == 6 or row[0].isalnum():
        if len(row[1]) == 10 or row[1].isalnum():
            if datetime.strptime(row[2], "%Y-%m-%d"):
                if 1 <= int(row[3]) <= 5:
                    products[row[1]] = products.get(row[1], 0) + 1
                    ratings[row[1]] = ratings.get(row[1], 0) + int(row[3])
                    return
    raise ValueError("Some Error")

def read(file: str, products: dict[str, int], ratings: dict[str, int]) -> tuple[int, int]:
    valid, invalid = 0, 0
    try:
        with open(file.replace("\\", "/"), "r") as f:
            for row in f.readlines():
                try:
                    process(row, products, ratings)
                    valid += 1
                except:
                    invalid += 1
    except Exception as e:
        print(f"Cannot open file {file}\nError: {e}")
    return valid, invalid

def start(dir: str, products: dict[str, int], ratings: dict[str, int]) -> tuple[int, int]:
    valid, invalid = 0, 0
    for name in os.listdir(dir):
        file = os.path.join(dir, name)
        if os.path.isfile(file):
            v, i = read(file, products, ratings)
            valid, invalid = valid + v, invalid + i
        elif os.path.isdir(file):
            v, i = start(file, products, ratings)
            valid, invalid = valid + v, invalid + i
    return valid, invalid

def main():
    dir = r"files"
    products, ratings = {}, {}
    valid, invalid = start(dir, products, ratings)
    average = {p: ratings[p] / products[p] for p in products}
    top_keys = sorted(average, reverse=True, key=lambda x: average[x])[:3]
    with open("summary.txt", "w+") as f:
        f.write(f"1) The total number of reviews processed -> {valid + invalid}\n")
        f.write(f"2) The total number of valid reviews -> {valid}\n")
        f.write(f"3) The total number of invalid reviews -> {invalid}\n")
        f.write("4) Top 3 products with the highest average ratings\n")
        for i, key in enumerate(top_keys):
            f.write(f"\t{i+1}) {key} -> {average[key]}\n")

if __name__ == "__main__":
    main()
```

🧪 **Sample Output:**

summary.txt

```
1) The total number of reviews processed → 30
2) The total number of valid reviews → 28
3) The total number of invalid reviews → 2
4) Top 3 products with the highest average ratings
   1) NSVSSEHGIJ → 5.0    "NSVSSEHGIJ": Unknown word.
   2) V2S8H8SJBJ → 4.0    "SJBJ": Unknown word.
   3) NV2C87IA1S → 4.0
```

✅ **Conclusion:**

The Python script reliably ingests multiple review files, extracts structured fields, and validates records to separate valid and invalid reviews. It computes per-product average ratings, identifies the top three products, and writes a clear summary. With robust error handling and scalable design, the script performs well on large datasets, turning unstructured text reviews into actionable insights.
