Note: I had two local commits, so three of them were attempted at the same time. Here's the record.

```
commit 8d1e2e5bfcc9d9c47d8e4dc4997f8cf45eec69df8 (HEAD -> main, origin/main)
Author: Cher Wang <checher@bu.edu>
Date:   Sun May 4 17:44:29 2025 -0400

    Finalized changes and refined test

commit c55dd6d4c4384d11f978d984e08c64e826849f01
Author: Cher Wang <checher@bu.edu>
Date:   Sun May 4 17:05:47 2025 -0400

    Added tests

commit c4245bbd993e99150ea5200f2129f2e8103458eb
Author: Cher Wang <checher@bu.edu>
Date:   Sat May 3 23:11:34 2025 -0400

    Implement K-means clustering and feature vector logic

commit 4d4fc94dc590334a56c3fe0b4af7d4732b8c580c
Author: Cher Wang <168463267+checherehc@users.noreply.github.com>
Date:   Sun May 4 17:38:55 2025 -0400

    Create README(Wrietup).md

commit 46fe86b40b6c027ab8b11523a4b77df38f5aeafd
Author: Cher Wang <checher@bu.edu>
Date:   Sat May 3 13:38:14 2025 -0400

    initial commit: create project structure
cher@chechers-MacBook-Pro customer_segmentation %
```

Project Overview

Goal:

To segment customers into clusters based on purchasing behavior and demographic traits using K-means clustering, in order to uncover meaningful consumer profiles.

Dataset:

- Source: Kaggle
- Size: 2,240 customer records × 29 columns
- Format: CSV

Data Processing

Loading into Rust:

- Used the csv crate to read the datasets
- Customer struct implements serde::Deserialize for structured parsing.

Cleaning / Transformation:

- Parsed date field Dt_Customer using chrono::NaiveDate
  Transformed features
  - Age from birth year
  - num_of_children from Kidhome + Teenhome
  - accepted_campaign_index for which campaign they accepted first
- Converted relevant features into numerical vectors for clustering.

Code Structure

Modules:

- main.rs: Program entry point, runs the clustering and prints summary.
- data_loader.rs: Handles reading and preprocessing the dataset.
- clustering.rs: Implements K-means and analysis functions.

Key Structs & Functions:

Customer Struct

Represents one customer.

- **Inputs**: Parsed CSV row
- **Outputs**: Populated Rust struct
- **Key fields**: income, education, marital_status, acceptedcmp1...
- Method:
    - feature_vector() → [f64] features for clustering

kmeans_cluster(customers, k, iterations)

Performs unsupervised K-means clustering.

- Inputs: List of Customer, number of clusters, iterations
- Output: Vec<usize> cluster index for each customer
- Logic:
    - Randomly initializes cluster centers
    - Iteratively assigns customers to closest center
    - Recalculates centers based on means
    - Prints cluster feature averages

Test

```
running 2 tests
test clustering::tests::test_distance ... ok
test clustering::tests::test_clustering ... ok

test result: ok. 2 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; fini
shed in 0.00s
```

Test descriptions:

- test_distance_symmetry: Confirms distance metric is symmetric.
- test_clustering_returns_valid_assignments: Validates clustering assigns all customers correctly.

Result

```
Loaded 2240 customers

 Cluster feature averages
Cluster 0
Income          :74033.84375
Education       :1.5546875
Marital Score   :0.6933035714285715
Recency         :49.10825892857143
Frequency       :0.004383602300354548
Total Spend     :1175.3995535714287
Number of Children:0.5825892857142857
Campaign Accepted:1.2734375
Age             :58.22767857142857

Cluster 1
Income          :21655.866812227076
Education       :1.0873362445414847
Marital Score   :0.7213973799126654
Recency         :49.52838427947598
Frequency       :0.0012901493151726835
Total Spend     :94.29694323144105
Number of Children:1.0043668122270741
Campaign Accepted:0.6135371179039302
Age             :50.801310043668124

Cluster 2
Income          :44613.09029345372
Education       :1.618510158013544
Marital Score   :0.6957110609480813
Recency         :48.893905191873586
Frequency       :0.002284936293771152
Total Spend     :294.1783295711061
Number of Children:1.2945823927765236
Campaign Accepted:0.8623024830699775
Age             :56.925507900677204


 Cluster summary:
Cluster 0 -> 896 customers | Avg Total Spend:1175.40
Cluster 1 -> 458 customers | Avg Total Spend:94.30
Cluster 2 -> 886 customers | Avg Total Spend:294.18
```

Interpretation:

- High spending tier has way higher income than other two tiers, around 2 times higher than middle tier and more than 3 times higher than low tier
- High spending tier is more valuable than other tiers not only because they spend more, but also because they spend more frequently(0.004), 4 times higher than the low tier.
- High spending consumers have a lower chance to have children. Therefore, campaign can detour from advertising to a household with children.
- High spending consumers are older than consumers in other two tiers
- Cluster 0: high spending, high income, high education, no children, older audience, responsive to campaign, frequent purchaser
- Cluster 1: low spending, low income, younger audience, usually have 1 child, not sensitive to campaign
- Cluster 2: middle spending, middle income, purchase more recently, usually have 1-2 child, mildly responsive to campaign

Instructions

To build: cargo build – release
To run: cargo run –release
Command-Line Input:

- None required — reads from data.csv

Expected Runtime:

- < 5 seconds for 2,240 rows with 3 clusters and 10 iterations.

AI-Assistance Disclosure and Other Citations

Use of ChatGPT:

- Debugging serde errors:
    - I encountered issues when deserializing CSV rows into the Customer struct. ChatGPT explained that I needed to use #[serde(rename = "...")] to match field names in the CSV (which were in PascalCase or camelCase) with my struct fields (which were in snake_case).
- Clarifying NaiveDate parsing with chrono crate:
    - ChatGPT helped me apply a manual deserializer for Dt_Customer due to its custom date format (dd-mm-yyyy).