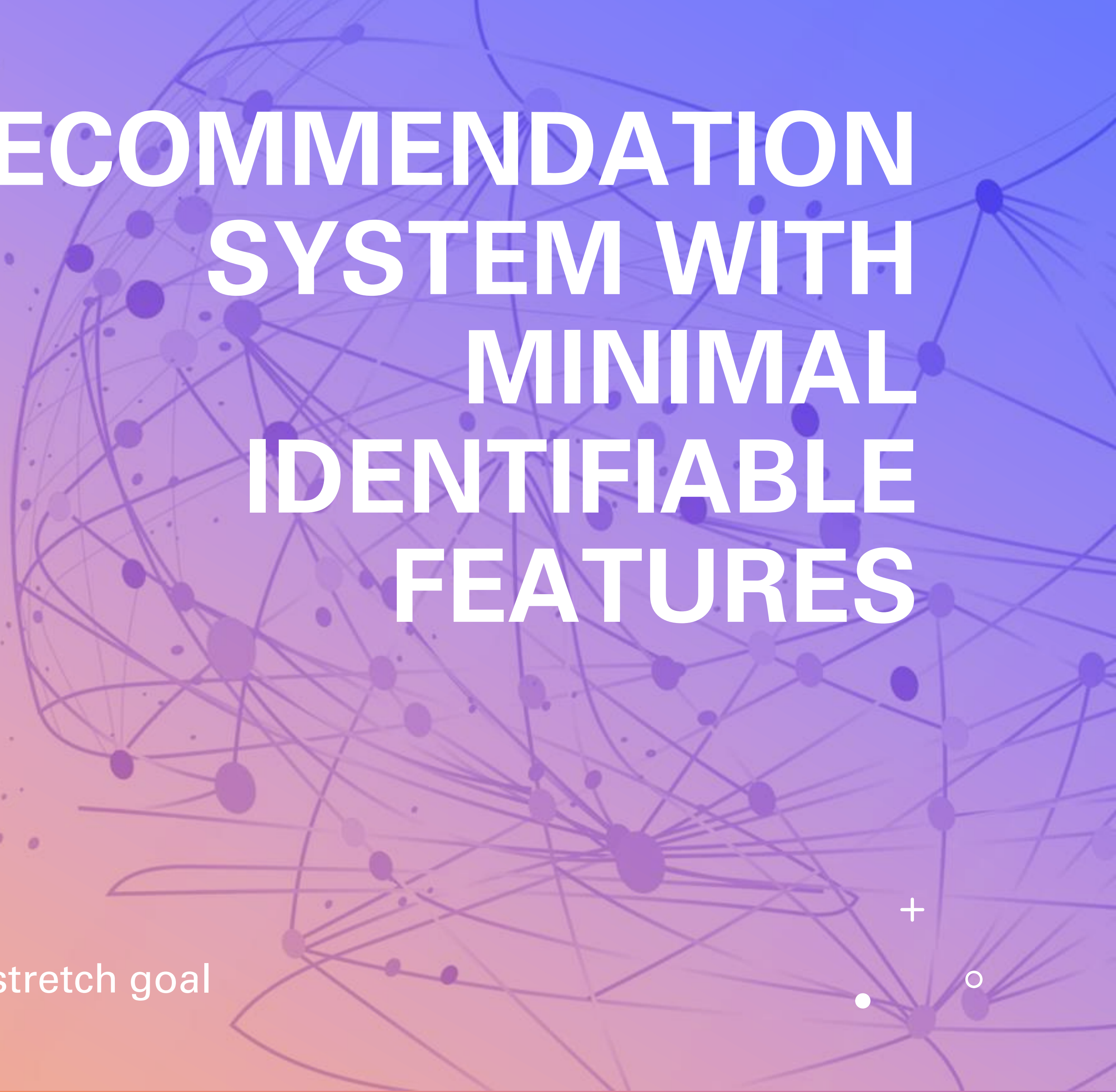
RECOMMENDATION SYSTEM WITH MINIMAL IDENTIFIABLE FEATURES



Exploring the possible feature / stretch goal

Data and Computational workflow

Initial_parameters.json

student_all_interests.csv

clubs_similarity.csv

pipeline.py

populate_dataset.py

club_similarity.py

recommend_clubs.py

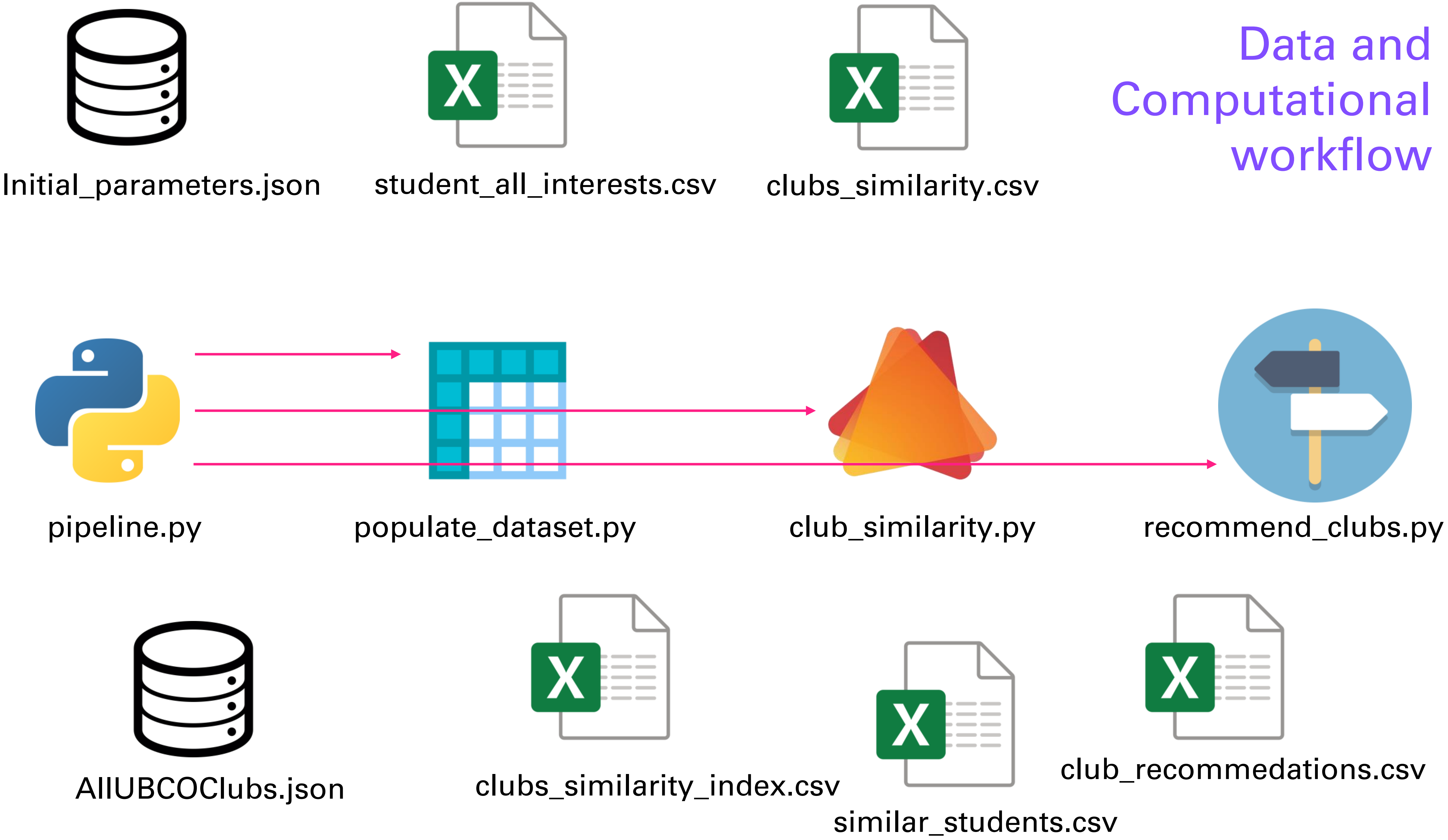
AllUBCOClubs.json

clubs_similarity_index.csv

similar_students.csv

club_recommedations.csv

Data and Computational workflow



Initial_parameters.json

student_all_interests.csv

clubs_similarity.csv

pipeline.py

populate_dataset.py

club_similarity.py

recommend_clubs.py

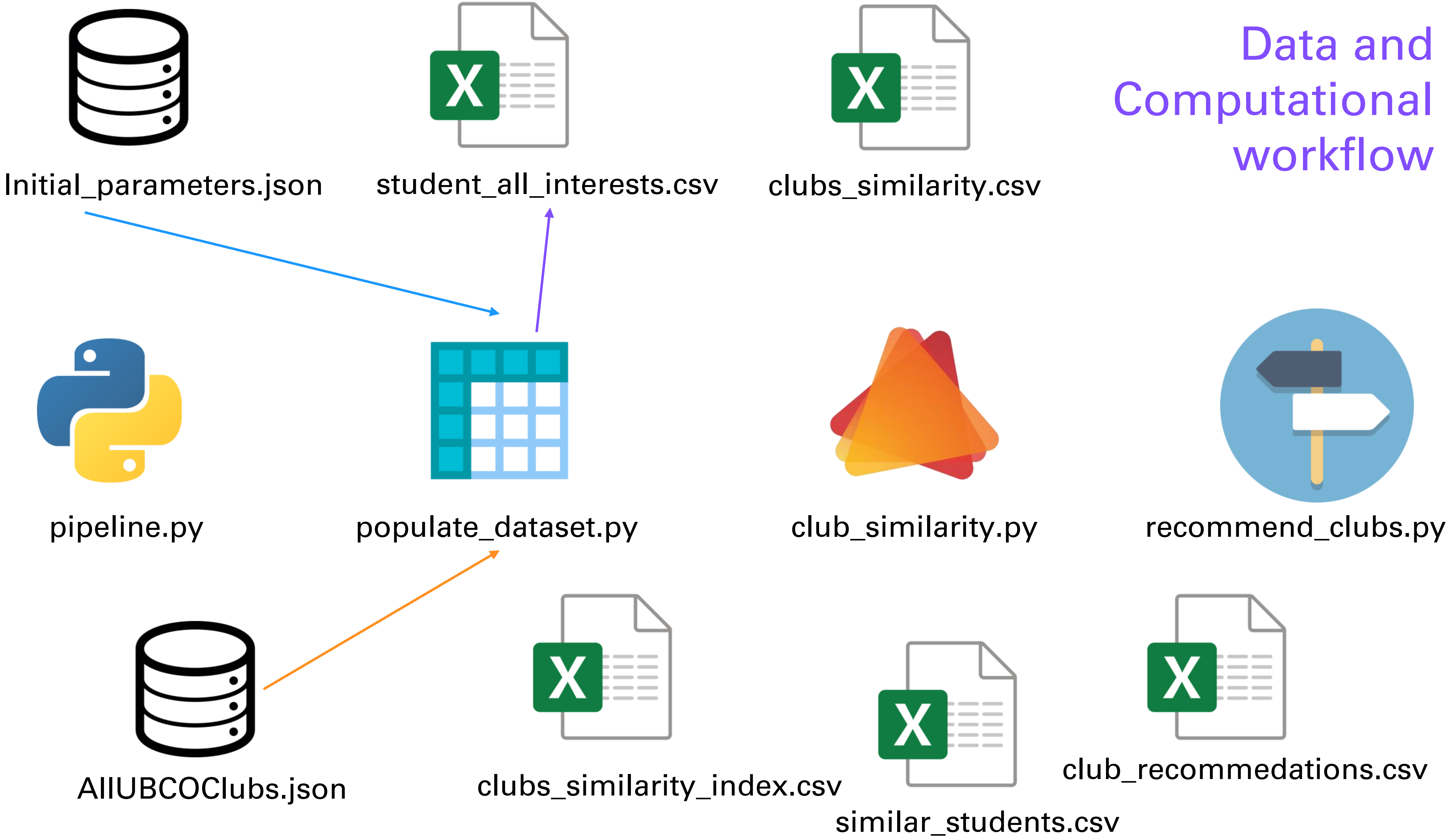
AllUBCOClubs.json

clubs_similarity_index.csv

similar_students.csv

club_recommedations.csv

Data and Computational workflow



Initial_parameters.json

student_all_interests.csv

clubs_similarity.csv

pipeline.py

populate_dataset.py

club_similarity.py

recommend_clubs.py

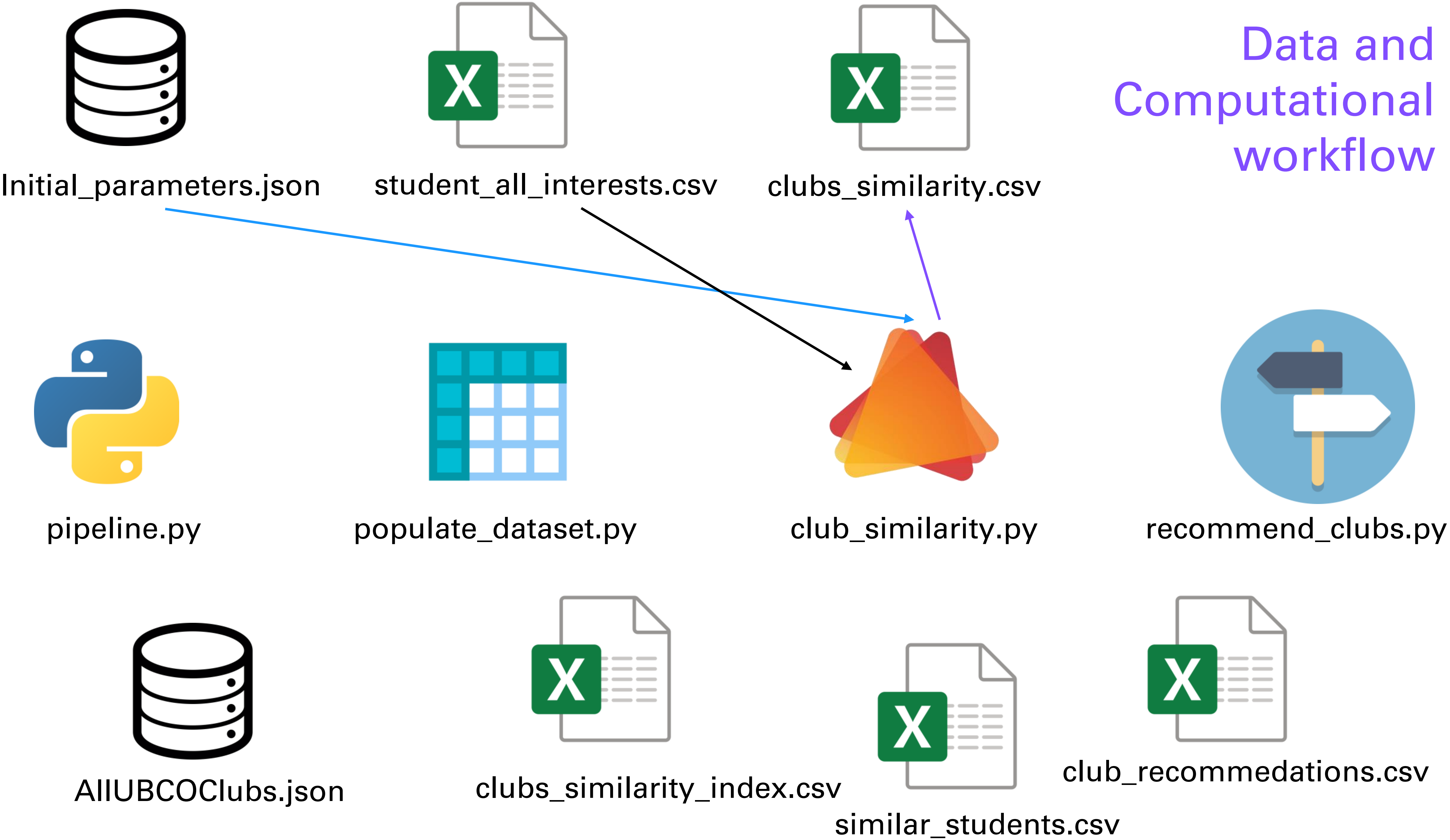
AllUBCOClubs.json

clubs_similarity_index.csv

similar_students.csv

club_recommedations.csv

Data and Computational workflow



Initial_parameters.json

student_all_interests.csv

clubs_similarity.csv

pipeline.py

populate_dataset.py

club_similarity.py

recommend_clubs.py

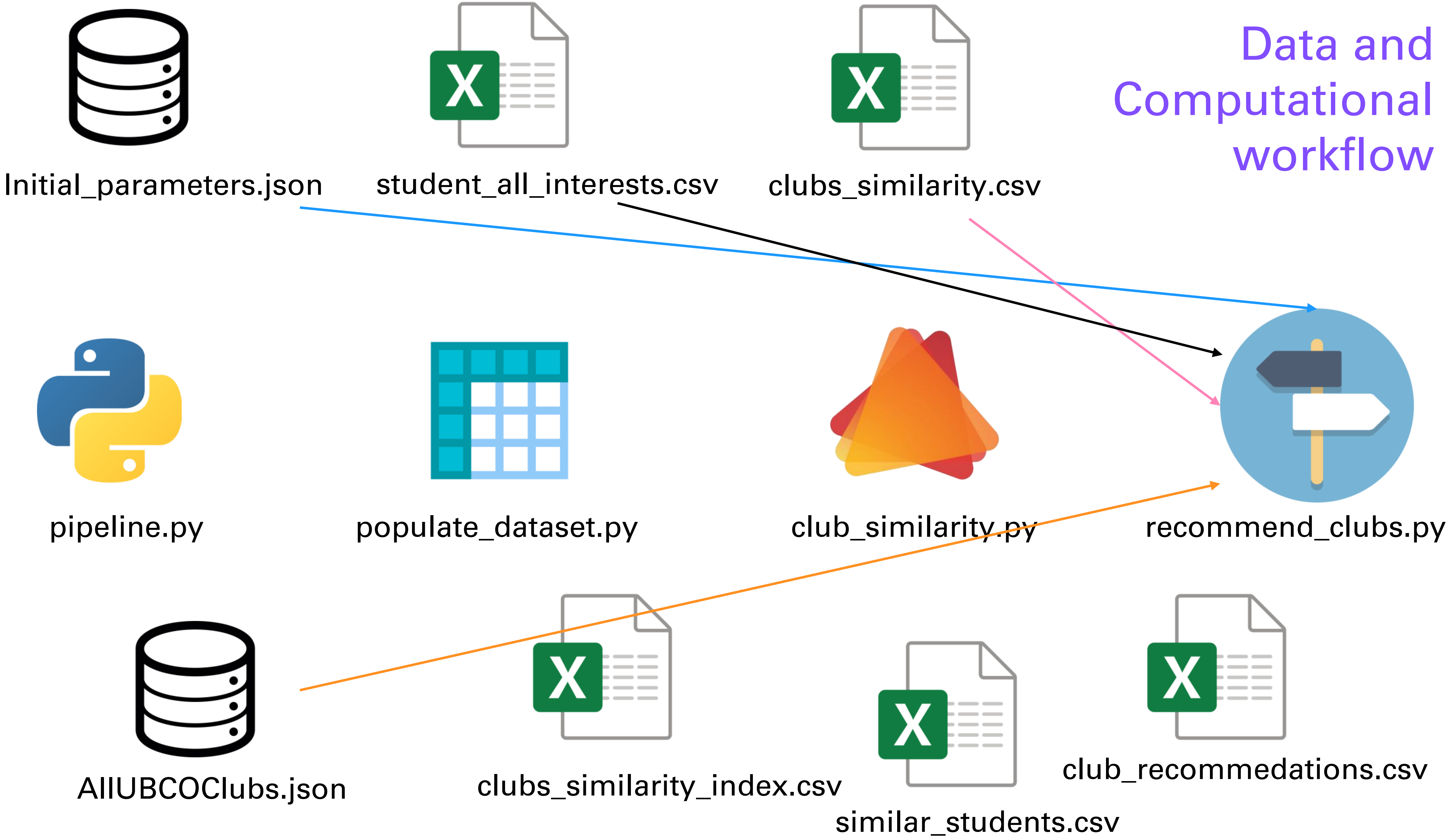
AllUBCOClubs.json

clubs_similarity_index.csv

similar_students.csv

club_recommedations.csv

Data and Computational workflow



Initial_parameters.json

student_all_interests.csv

clubs_similarity.csv

pipeline.py

populate_dataset.py

club_similarity.py

recommend_clubs.py

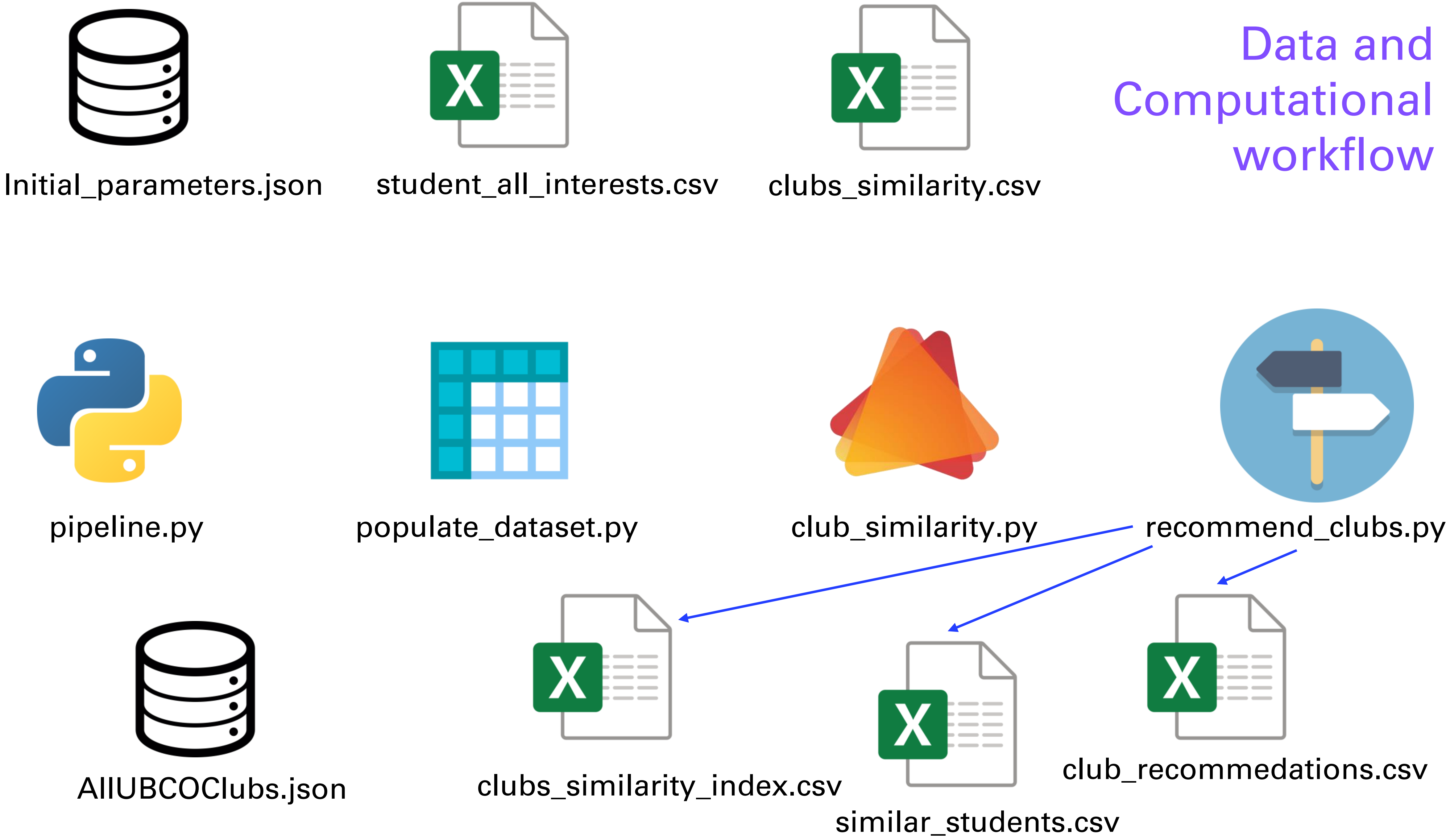
AllUBCOClubs.json

clubs_similarity_index.csv

similar_students.csv

club_recommedations.csv

Data and Computational workflow



Initial_parameters.json

student_all_interests.csv

clubs_similarity.csv

pipeline.py

populate_dataset.py

club_similarity.py

recommend_clubs.py

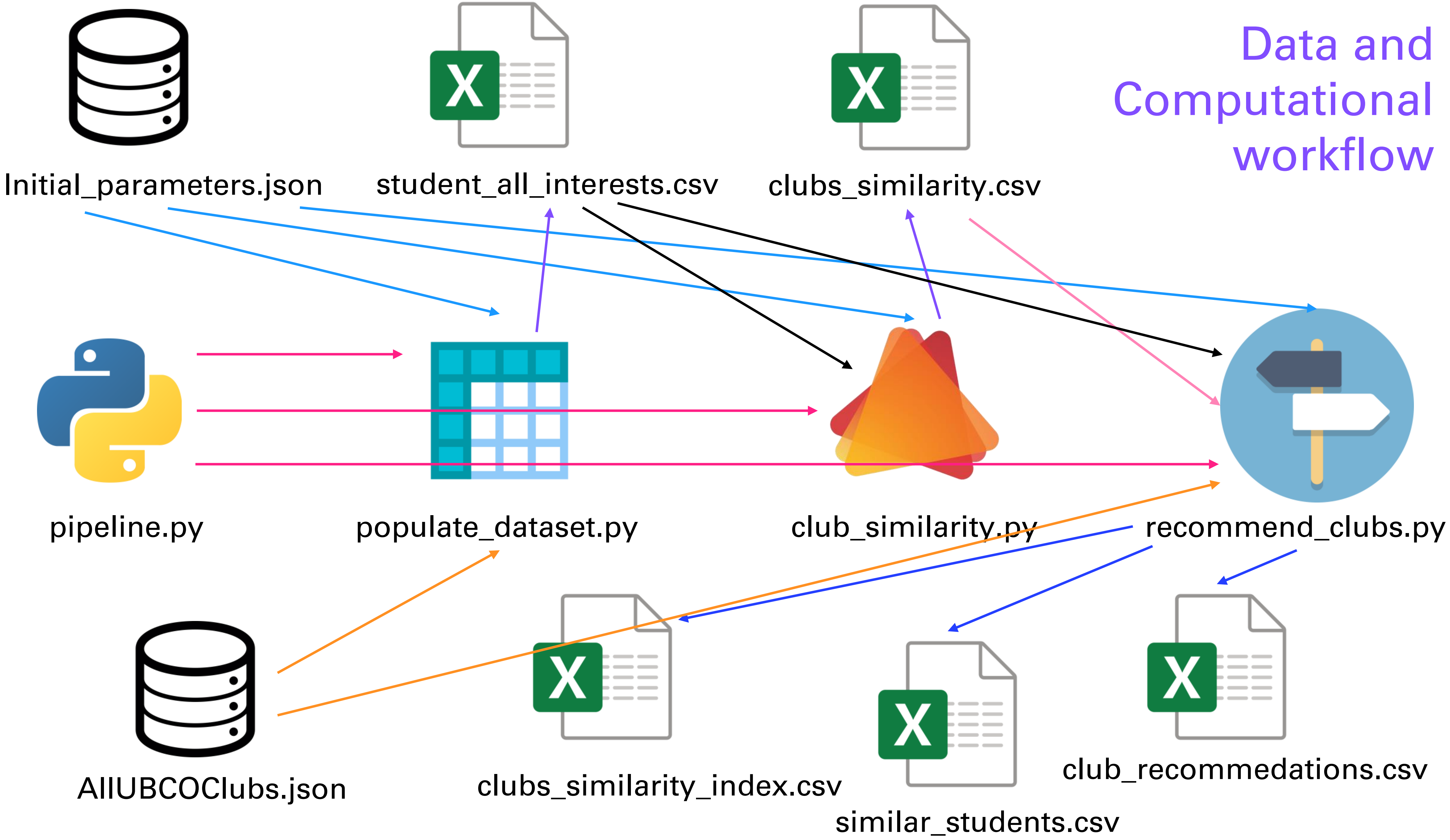
AllUBCOClubs.json

clubs_similarity_index.csv

similar_students.csv

club_recommedations.csv

Data and Computational workflow



Initial_parameters.json

student_all_interests.csv

clubs_similarity.csv

pipeline.py

populate_dataset.py

club_similarity.py

recommend_clubs.py

AllUBCOClubs.json

clubs_similarity_index.csv

similar_students.csv

club_recommedations.csv

populate_dataset.py

Objective: Creates a random dataset for *n* students

- Generates random e-mails
    - Random characters
    - Random special characters
    - Random presence and position of special characters
    - Random domains

populate_dataset.py

Objective: Creates a random dataset for *n* students

- Generates random e-mails
  - Random characters
  - Random special characters
  - Random presence and position of special characters
  - Random domains
- One-way one-time PBKDF2-HMAC-SHA1 encryption of e-mails

populate_dataset.py

Objective: Creates a random dataset for *n* students

- Generates random e-mails
  - Random characters
  - Random special characters
  - Random presence and position of special characters
  - Random domains

- One-way one-time PBKDF2-HMAC-SHA1 encryption of e-mails

- Generates random club interests
  - Random number of interested clubs
  - Random choice of clubs

populate_dataset.py

Objective: Creates a random dataset for *n* students

- Generates random e-mails
  - Random characters
  - Random special characters
  - Random presence and position of special characters
  - Random domains
- One-way one-time PBKDF2-HMAC-SHA1 encryption of e-mails
- Generates random club interests
  - Random number of interested clubs
  - Random choice of clubs
- Generates random event interests [still under development]

club_similarity.py

Objective: Finds similar clubs between all pairs of students

- Computes the intersection of a pair of club lists from two different students club_similarity.py

Objective: Finds similar clubs between all pairs of students

- Computes the intersection of a pair of club lists from two different students
- Generates club_similarity.csv for better interpretability of the recommendation system

recommend_clubs.py

Objective: Generates a .csv of club recommendations

- Finds similarity indices between a pair of different students (how many clubs are similar?)
  - Stored as clubs_similarity_index.csv
- Sorts these values for each student to find most similar students
  - Stored as similar_students.csv

- Computes importance of a similar user
    - Importance = similarity_index(this_user, other_similar_user)

recommend_clubs.py

recommend_clubs.py

- Computes importance of a similar user
  - Importance = similarity_index(this_user, other_similar_user)
- Importance-based scoring system
  - (Let) “student” be 1 student for whom we need recommended clubs
  - (Let) “other_students” be a list of other students ranked by similarity

recommend_clubs.py

- Computes importance of a similar user
  - Importance = similarity_index(this_user, other_similar_user)

- Importance-based scoring system

  - (Let) “student” be 1 student for whom we need recommended clubs
  - (Let) “other_students” be a list of other students ranked by similarity

  For “each_student” in “other_students”

  1) Find clubs that “student” is not a part of (np.setdiff1d of the club lists)
  2) Update the “weight” for each club based on importance(each_student)

recommend_clubs.py

- Computes importance of a similar user
    - Importance = similarity_index(this_user, other_similar_user)

- Importance-based scoring system

    - (Let) “student” be 1 student for whom we need recommended clubs
    - (Let) “other_students” be a list of other students ranked by similarity

    For “each_student” in “other_students”

    1) Find clubs that “student” is not a part of (np.setdiff1d of the club lists)
    2) Update the “weight” for each club based on importance(each_student)

    - Normalise the weights for each club

recommend_clubs.py

- Computes importance of a similar user
  - Importance = similarity_index(this_user, other_similar_user)

- Importance-based scoring system

  - (Let) “student” be 1 student for whom we need recommended clubs
  - (Let) “other_students” be a list of other students ranked by similarity

  For “each_student” in “other_students”
  1) Find clubs that “student” is not a part of (np.setdiff1d of the club lists)
  2) Update the “weight” for each club based on importance(each_student)

  - Normalise the weights for each club

  - Store them from highest weight to lowest weight (i.e. most recommended to least recommended)

  - Generates club_recommendations.csv

Advantages

- Requires minimal personal student information (one-time one-way encryption of e-mail IDs or any other personally identifiable information)

Advantages

- Requires minimal personal student information (one-time one-way encryption of e-mail IDs or any other personally identifiable information)

- Highly interpretable and intuitive model: better for scalability and for adding new parameters/features in the future

Advantages

- Requires minimal personal student information (one-time one-way encryption of e-mail IDs or any other personally identifiable information)
- Highly interpretable and intuitive model: better for scalability and for adding new parameters/features in the future
- Importance-based scoring removes any advantage that may arise due to a happenstance order

Advantages

- Requires minimal personal student information (one-time one-way encryption of e-mail IDs or any other personally identifiable information)
- Highly interpretable and intuitive model: better for scalability and for adding new parameters/features in the future
- Importance-based scoring removes any advantage that may arise due to a happenstance order
- Recommendations takes all user interests into account, not just those that are most similar. (Can be treated as a hyper-parameter to include n-most similar users).

Advantages

- Requires minimal personal student information (one-time one-way encryption of e-mail IDs or any other personally identifiable information)
- Highly interpretable and intuitive model: better for scalability and for adding new parameters/features in the future
- Importance-based scoring removes any advantage that may arise due to a happenstance order
- Recommendations takes all user interests into account, not just those that are most similar. (Can be treated as a hyper-parameter to include n-most similar users).
- Generates a unique order of all clubs and not just a subset of recommended clubs. This can be used as a native club-view order for each student (each student sees all clubs but in a unique order, personalized to them)

Areas of improvement

- For better personalized lists, more data is needed
  - Women in engineering
  - Indonesian students of Okanagan
  - African Caribbean Students Club
  - Asian Student Association
  - Bible Discussion Club

Areas of improvement

- For better personalized lists, more data is needed
  - Women in engineering
  - Indonesian students of Okanagan
  - African Caribbean Students Club
  - Asian Student Association
  - Bible Discussion Club
- Personal data vs Personalized results trade-off
  - Example: biased results (Women in engineering)
  - Using “categories” data in AllUBCOClubs.json

What about new users?

- Initial run: no club interests for any student

What about new users?

- Initial run: no club interests for any student

- Three possible approaches:

    1. Recommend top clubs from each category
        - Some clubs are “uncategorized”

What about new users?

- Initial run: no club interests for any student

- Three possible approaches:

  1. Recommend top clubs from each category
     - Some clubs are “uncategorized”

  2. [Currently more do-able] Use a questionnaire to select those categories that the student is interested in
     - Reluctance to answer a questionnaire (cannot assume that everyone will respond)

What about new users?

- Initial run: no club interests for any student
- Three possible approaches:
  1. Recommend top clubs from each category
     - Some clubs are "uncategorized"
  2. [Currently more do-able] Use a questionnaire to select those categories that the student is interested in
     - Reluctance to answer a questionnaire (cannot assume that everyone will respond)
  3. (Best case but more high-effort)
     - Obtain the dataset of current students and what clubs they are a part of
     - Generate an average portfolio by faculty
     - Display this order