- P1.大家好，我們是第二組，我們的主題是 user-oriented book recommendation system with user interface
- P2. 現今賣書網站已有許多能快速查找書籍的方式，但在個人化推薦的部分可能還稍有欠缺，而使用者的回饋通常包含評分和評論，由於不能保證兩者在評價正負面上的一致性，因此我們想針對使用者購買書籍後的評論進行情感分析，並與其原始評分結合計算新的推薦指數，最後使用兩種推薦模型過濾出推薦書籍給使用者。
- P3. input 的資料集包含讀者評論、評分，以及書籍基本資料像是出版日期、綜合評分等，而 process 部分則利用讀者的購書資訊和回饋進行分群，並在各讀者分群中尋找專屬於該群的購書關聯規則，最後 output 時推薦最適合該讀者的書籍，增加平台銷量。
- P4.target performance 的部分，我們參考一般推薦系統所使用的模型評估標準，分別是 MRR 以及 MAP，我們的系統預計使用的推薦方式為 knn clustering 加上 association rule 沒有現成的參考比較依據，因此我們也同時實作 Memory-based with User-based 協同過濾推薦法，作為比較的基準，而鑒於我們參考部分相關資料所示，能讓兩個 model 在 MRR 皆有大於 0.45 的分數，而在 MAP 有大於 0.4 的分數就已經達到優秀的結果
- P5. 這邊針對兩種評估指標做一些簡介，相關公式細節如畫面所示，首先 MRR 是將所有 user 推薦結果的第一個命中位置取倒數相加做平均，而 MAP 則是還有考量到後續命中的位置
- P6. data 的部分，我們使用的是 kaggle 上的 Amazon book review 這個資料集，包含兩個 csv 檔案，共 1000 多位使用者針對 21 萬本不同書籍所做之 30 萬筆評論
- P7.兩個 csv 分別為 book_rating 及 book_data，這邊列出檔案個別包含的 attributes 細節
- P8.接著我們對資料進行簡易的視覺化分析，左圖為前十大 categories 在資料集中的分布狀況，可以看到 Fiction 類型在資料集的比例遠大於其他任何類型；右圖部分是資料集中購買不同數量書籍的讀者分布，其中購買 1~200 本書的讀者佔了絕大多數。
- P9.Challenge 的部分:針對目前所遇到的問題，我們發現 categories 特徵在資料集中的種類非常多(共 1 萬多種)，但作為分群的重要依據，無

法將這個特徵刪除，因此我們目前的的解決方法主要有兩種:第一種為篩選前十大的 categories，第二種為利用 unsupervised clustering 建立新的 categories 特徵；另外由於原始資料集中沒有可以用來計算讀者相似度的特徵，因此我們會新增新的特徵補足這方面的缺失

- P10. 再來進入資料前處理，主要可分為三大步驟；首先資料清洗會刪去含有未知使用者以及重複出現的資料；而特徵編碼會將類別型特徵例如 categories 進行 one hot encoding；接著針對讀者對於所購買書籍的評論利用 fine tune 後的 BERT 模型進行情感分析，結合讀者原始評分計算出新的特徵，表現讀者的喜愛程度。

- P11. 工作流程可分為資料前處理以及建立推薦模型兩部分；首先我們會使用前述方法將原始資料進行清洗和前處理，並根據不同使用者分割成訓練資料和驗證資料，用以衡量推薦模型的效能。我們將會建立兩種推薦模型，第一種是利用 KNN 演算法將資料庫中的讀者進行分群，並根據新讀者所在的群體尋找購書的關聯規則；第二種是利用 memory-based with user-based 協同過濾推薦，找出和新讀者具有高度相關的讀者，將該讀者的購買紀錄做為推薦依據。在使用驗證資料進行參數調整後，我們會將上述推薦模型和使用者介面進行整合，呈現出最後的成果。
- P12. 使用者介面的設計理念是希望藉由讀者對於特定種類書籍的喜好，找出最適合該讀者的 10 本書，並依照推薦指數排序，呈現給使用者。為了瞭解不存在於資料庫中的讀者偏好，系統會獲取使用者對特定問題的回饋，例如隨機列出書籍讓使用者依照興趣選取，以此作為推薦系統後端模型的輸入，最後輸出推薦指數最高的 10 本書籍。
- P13. 接著環境的部分我們主要在 windows10 上使用 VS code 平台用 python 語言進行開發
- P14.最後是我們的預期進度，感謝聆聽。