Summary

- Pipeline Details:
    - Data Cleaning and Preprocessing:
        - Source on public kernels (on Kaggle): https://www.kaggle.com/khyeh0719/how-to-win-kaggle-competition-final-project
            - Executable on Kaggle.
        - Data Cleaning:
            - Repeated rows: 24 rows found and removed.
            - Outlier Values: item_price <0 and >30000 (Really really far from others)
        - EDA:
            - Monthly sales count distributions.
            - Item prices distribution:
                - Right skew heavily, so I mapped it to log space with np.log1p.
    - Hyper-parameters optimization:
        - Simple Holdout (Time-based Split)
        - Write my own Grid Search Function on validation data.
            - See uploaded notebook: Notebook for Peer Review (1st\2nd Level Model Training)
                - customized_grid_search_simple_holdout_evaluate(_stage2)
    - First Level Training:
        - Source Code:
            - **Notebook for Peer Review (1st Level Model Training)**:
                - Executable, but need ENOUGH memory and is computational intensive. Apologies for inconvenience, since I didn't have time to clean the code, serialize those models and test them again…
        - Data (part1): Preprocessed Data WITHOUT text features.

        - Models:
            - Linear Models: Ridge, Lasso
            - Tree-Based Models: Random Forest, LightGBM.
            - Neighbor Models: K-Nearest Neighbors
            - Clustering: Minibatch KMeans.
    - Data (part2): Preprocessed Data WITH text features.
        - TF-IDF + TF-IDF(Binarized) + HashingVectorizer + HashingVectorizer(Binarized), and applied TruncatedSVD to reduce dimensions.
            - Note: PCA does not support sparse input.
        - Model: Ridge.
- Second Level Training:
    - Source Code:
        - **Notebook for Peer Review (2nd Level Model Training)**
            - Executable, and should be fast with serialized lightgbm model in the uploaded files.
    - Data: From First Level's Validation Data.
        - First Level Models' Predictions (+ Feature Engineering) + 1st Level Features.
            - Take min\max\med\stddev\geo-mean of first level model predictions.
        - Model:
            - LightGBM.

- Uploaded Files(s):
    - Source Code:
        - Data Cleaning + EDA: https://www.kaggle.com/khyeh0719/how-to-win-kaggle-competition-final-project
        - Notebook for Peer Review (1st Level Model Training)
        - Notebook for Peer Review (2nd Level Model Training)

        - Intermediate Data:
            - stage2_data.dataframe
            - stage2_data_text.dataframe
            - text_features.dataframe
            - "input" folder (for preprocessed data):
                - proc_train.csv.gz
                - proc_test.csv.gz
                - test_id.csv
        - Serialized Model:
            - lgb_stage2.model
                - For 2nd level submission generation.
- Requirement and How to run:
        - To run fast, execute "Notebook for Peer Review (2nd Level Model Training)"
            - Open Jupyter notebook.
            - Run all.
        - Library:
            - My local computer:
                - numpy: 1.21.1
                - pandas: 0.21.0
                - sklearn: 0.19.1
                - lightgbm: 2.0.10
            - Kaggle:
                - Latest Docker image.
        - Computational Power:
            - Kaggle & My local computer:
                - 4 core, 16 GB RAM.
- Tool(s)
        - Programming Language: Python.
        - Platform\Environment:
            - EDA: On Kaggle using its kernel environment.
            - Model Training: Locally with Jupyter notebook.
- How long it takes to train models.

- Hyper-parameter optimization: 4 Hrs.
- First Level Training: 3 Hrs.
- Second Level Training: 1 Hrs.