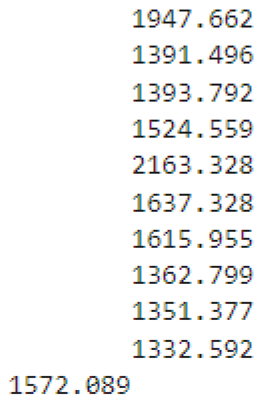
Section 1: Technical Details: Discuss details such as data pre-processing and other non-trivial implementation aspects of your models. Do NOT paste your code in the report. Instead, explain the technical steps in clear English. Your description should be comprehensive enough for your fellow PSL classmates to replicate your results.

First the data is processed using Singular Value Decomposition (SVD) smoothing by different ‘Dept’ values. The numerical columns are extracted, and their mean is calculated. The data is centered by subtracting the mean. SVD is applied to decompose the centered data into three matrices (U, D, Vt). A reduced representation is obtained by keeping only the top 'd' components. The smoothed data is then reconstructed and added to the original mean to obtain the final smoothed values. To reconstruct the smoothed data, NaN values need to be found and deleted. Since I replaced NaN with zero before performing the SVD smoothing, they are easy to find. The zero entries that are in the original data frame is not a problem, since they are already replaced by 0.1 before looping through different ‘Dept’ values. After converting all zeros with NaN, they are removed from the updated train data.
Duplicates are dropped from both train and test data to get the unique Store-Dept Pairs. An inner join is performed on 'Store' and 'Dept' columns to find the common pairs between training and test sets. The common pairs are merged back with the original data using 'Store' and 'Dept'. Then it got fed into the preprocess function. It first converts the “date” column of the input to a pandas DateTime object and stores it in the variable tmp. Then it extracts the ISO week number from tmp and adds a new column named 'Wk' to the DataFrame to store these week numbers. Then it extracts the year from tmp and adds a new column named 'Yr' to the DataFrame to store these years. It converts the 'Wk' column to a pandas Categorical type, specifying the categories as the range from 1 to 52 since there are 52 weeks. Finally, the function returns the modified DataFrame with the added 'Wk' and 'Yr' columns.
For both train and test set, Patsy's dmatrices function is used to create a design matrix (X) and a response vector (y) based on the specified formula. The formula of train set is 'Weekly_Sales ~ Weekly_Sales + Store + Dept + Yr + np.power(Yr, 2) + Wk', and test set is 'Yr ~ Store + Dept + Yr + np.power(Yr, 2) + Wk', where np.power(Yr, 2) is the quadratic term of Yr(year) since it is treated as a numerical variable. The resulting design matrix is grouped by 'Store' and 'Dept' and stored in a dictionary for both train and test. The keys (unique combinations of 'Store' and 'Dept') from the train set are stored in the keys variable.
For each key set, X_train and X_test are defined based on the current key. The response variable Y is extracted from the 'Weekly_Sales' column of X_train. Columns related to the response variable ('Weekly_Sales', 'Store', 'Dept') are dropped from X_train. Columns in X_train and X_test that are entirely filled with zeros are dropped. Columns in X_train are iteratively checked for linear dependence, and if the residuals are below 1e-10, the column is dropped. Ordinary Least Squares (OLS) regression model is fit using the sm.OLS from the statsmodels library. Predictions are made using the learned coefficients on the test set. Predictions for each key are concatenated into a single DataFrame (test_pred). Since fold 5 has high WMAE, and it is primarily due to the inclusion of two holiday weeks, a post-prediction adjustment is made to shift 1/14 of Week 51 weekly sale values to Week 52 for only fold 5 as suggested by the professor. The shift function iterates over each row in the DataFrame. For each row, it checks if the value in the 'Wk' column is equal to 51. If the condition is true, it calculates a reduction

amount based on the 'Weekly_Pred' value and the specified percentage (1/14 was chosen since it gives the best result). It checks the next week's index (next_week_index) and verifies whether it corresponds to week 52. If the condition is met, it adds the reduction amount back to the 'Weekly_Pred' column for the next week. The way fold 5 is identified is by checking the first entry value of the 'Date' column. Then NaN values in the 'Weekly_Pred' column are filled with zero, and the final predictions are saved to a CSV file.

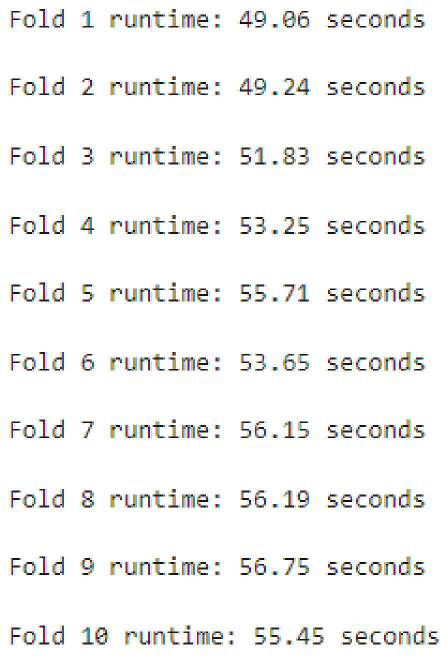
Section 2: Performance Metrics: Report the accuracy of your prediction on each of the 10 test datasets (refer to the evaluation metric described above), the execution time of your code, and details of the computer system you used (e.g., Macbook Pro, 2.53 GHz, 4GB memory or AWS t2.large) for each of the 10 fold.

Performance table shown as below (Average WMAE is at the bottom line):



```
          1947.662
          1391.496
          1393.792
          1524.559
          2163.328
          1637.328
          1615.955
          1362.799
          1351.377
          1332.592
 1572.089
```

My computer system that was used for this project is Windows 11, AMD Ryzen 9 7900X processor, 32gb ram, graphics RTX 3070. The execution time table is shown as below:



```
Fold 1 runtime: 49.06 seconds

Fold 2 runtime: 49.24 seconds

Fold 3 runtime: 51.83 seconds

Fold 4 runtime: 53.25 seconds

Fold 5 runtime: 55.71 seconds

Fold 6 runtime: 53.65 seconds

Fold 7 runtime: 56.15 seconds

Fold 8 runtime: 56.19 seconds

Fold 9 runtime: 56.75 seconds

Fold 10 runtime: 55.45 seconds
```