Promotion Bump Assignment Report

Botan Bulut

CONTENTS

1. DATASET INFORMATION

Assignment 4.1a.csv The data contains daily sales of samples of items in several stores on a specific time frame. Negative sale quantities represent returns. Each row represents a sale (or return) activity for an item in a store at a specific day. If a store-item combination has no observation on a certain day you can assume there are no sales for that item at that store on that day. Information of the dataset is given below:

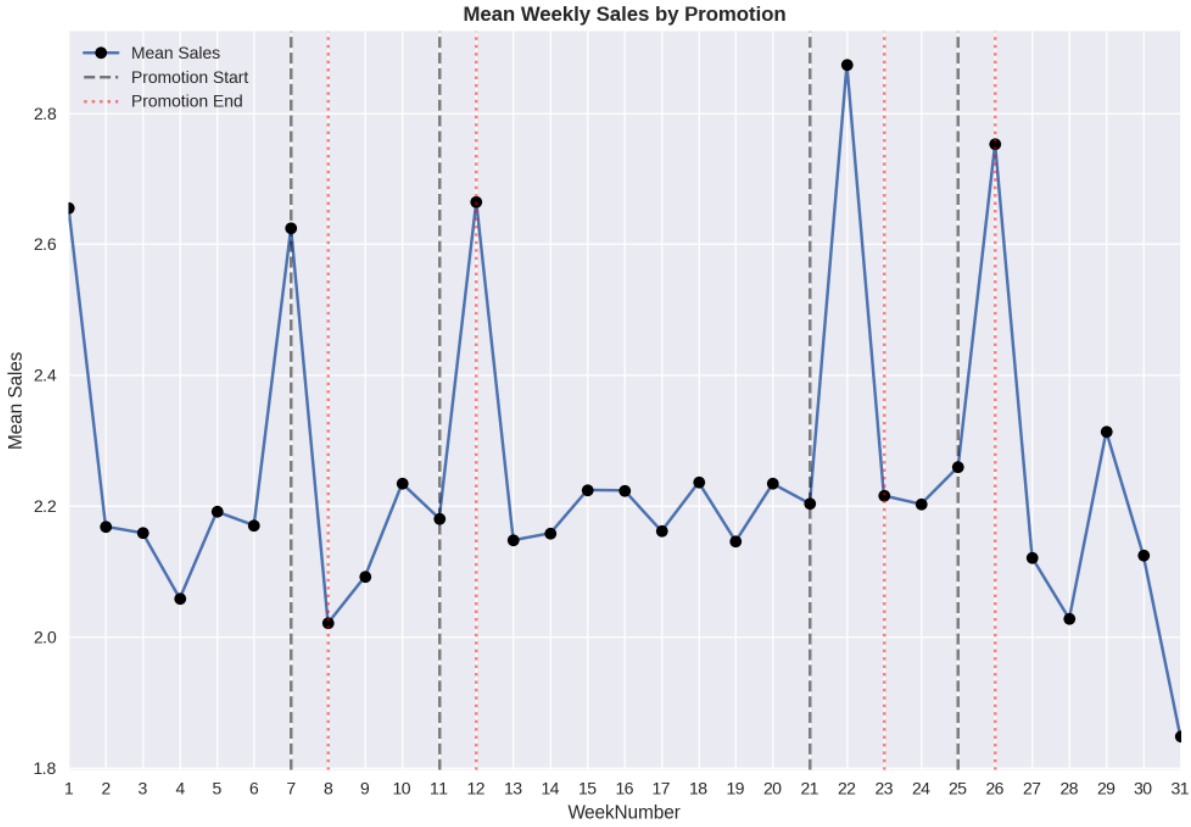
```
RangeIndex: 1873618 entries, 0 to 1873617
Data columns (total 4 columns):
 #   Column              Dtype
---  ------              -----
 0   Date                object
 1   StoreCode           int64
 2   ProductCode         int64
 3   SalesQuantity       int64
dtypes: int64(3), object(1)
memory usage: 57.2+ MB
```

The other dataset, Promotiondates.csv, contains beginning and the end dates of 6 promotions that took place in 2015. Complete data is given below:

Period	StartDate	EndDate
Promo1	2/10/2015	2/17/2015
Promo2	3/15/2015	3/22/2015
Promo3	5/24/2015	6/1/2015
Promo4	6/21/2015	6/28/2015
Promo5	1/9/2015	6/9/2015
Promo6	20/11/2015	27/11/2015

2. DATA CLEANING AND PREPROCESSING

We start data cleaning and preprocessing tasks with converting the date column from object type to datetime type for both 4.1a.csv and Promotiondates.csv. We also fix the date values of Promo6 so that it is consistent with other promotion date formats. We generate week number (ISO calendar week) columns from the date information in both data sets. This will also be useful in the weekly data grouping. Start week and end week of the promotion dates are ambiguous:



Plot above allows us to identify the mean sale jump weeks during promotions. Namely, week 7, 12, 22, and 26 are the promotion jump weeks. So we will use these weeks to generate a dummy variable named IsPromoWeek. Following code output shows the value counts of IsPromoWeek:

```
IsPromoWeek

False     1614390

True      259228

Name: count, dtype: int64
```

We also distinctly show the promotion week as dummy variable (in the case of assignment4.1a.csv first 4 promotions were present). Results are shown below:

```
IsPromo1        67939
IsPromo2        75285
IsPromo3        56793
IsPromo4        59211
dtype: int64
```

We will divide products and stores into 3 clusters each. Products with higher average weekly sale per store during non-promotion periods will be called “Fast items” and items with lower weekly average sale per store will be labeled as “Slow items”, items in between will be called “Medium items”. We will apply a similar approach to stores as well.

We generate no-promotion data as a subset of assignment4.1a.csv using a boolean mask where `IsPromoWeek == False` condition is satisfied. Clustering process for the products is given below:

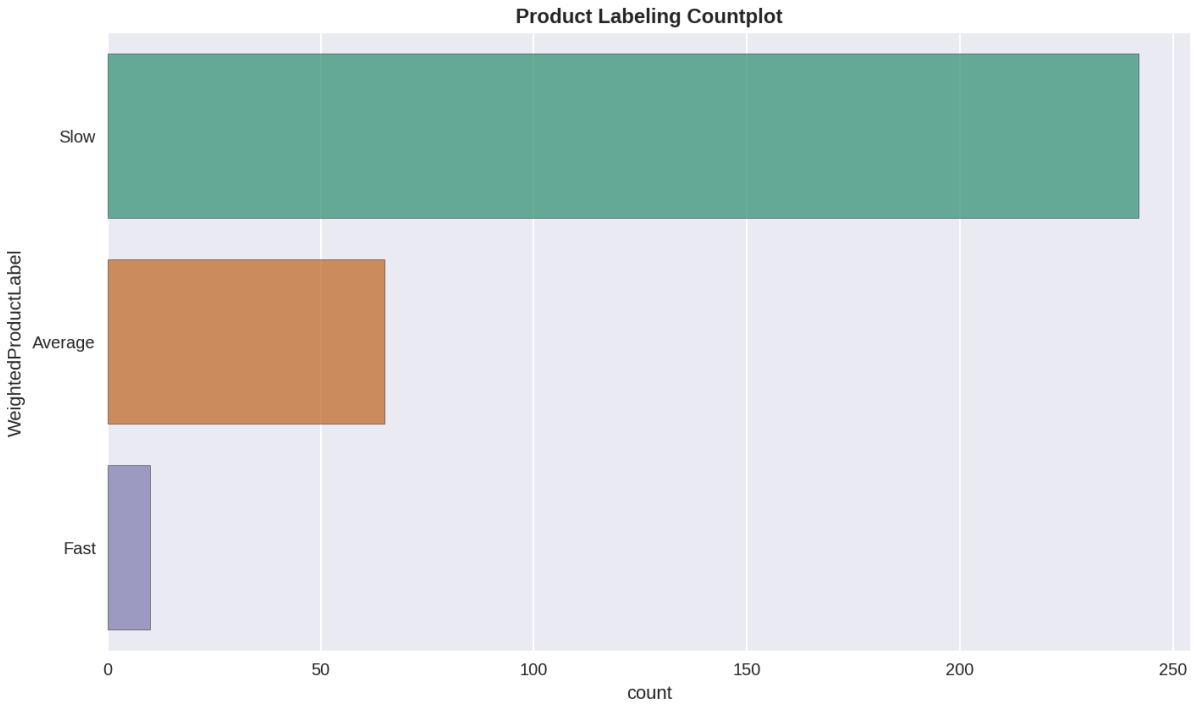
1. We calculate the mean sales value for each product per store by week using pivot table and fill missing pivot values with 0 (since missing information indicates no sales).
2. We use a pivot table to calculate specific store mean sales for a specific week and store this information.

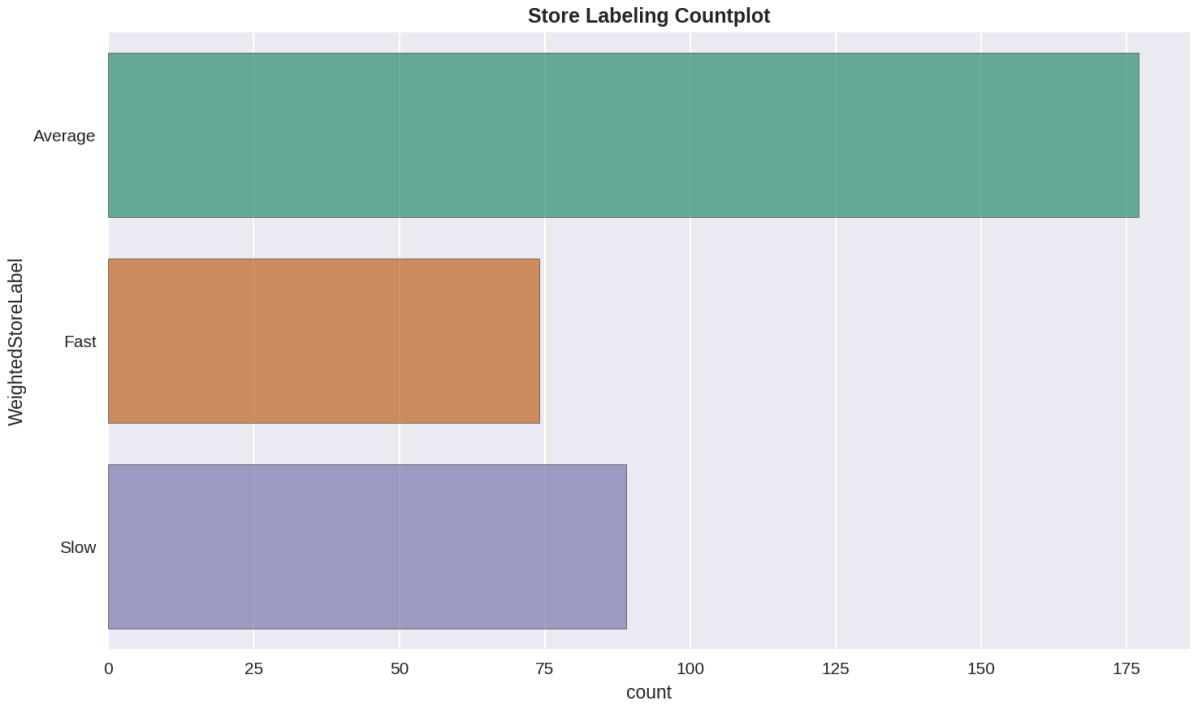
3. For product classification, we assume that attaining higher mean sales for a given store code and week number mean (for all products) indicates success within the week. On the other hand, if the mean is less than or equal 0 then we can say that for that specific week and store combination product is “slow”. Everything in between will be considered as “average”.
4. We assign 0, 1, and 2 for slow, average and fast items respectively. Then, we group the performance dataframe by product codes and take the mean of label numbers. If the mean is above 1.33, then the product's general performance is “Fast”, so its weighted label is “Fast”. “Slow” products are also labeled in a similar manner: if the mean value is less than 0.66, we apply the slow label. Everything in between becomes average.

Clustering process for the stores is very similar to the clustering of products. However, the labeling (not weighted labeling) method is different. It uses the 25th and 75th percentiles of general distribution. If store weekly mean sales is above this 75th percentile value it is a “Fast” store, and if its mean is less than 25th percentile value then it is a “Slow” store. In between values are considered as “Average”. Percentiles used in the calculation are given below:

```
25%         1.5740
75%         2.4767
```

Countplots below shows the resulting clustering count plots for product and stores:





Note that we observe a lot of slow products in the cluster. This happened because of the filling missing values with 0. Finally, we merge weighted label information to our main dataframe so that it can be used in exploratory data analysis and machine learning.

3. EXPLORATORY DATA ANALYSIS

Table below shows the descriptive statistics of sales column:

```
count   1873618.0000
mean          2.2466
std           5.0290
min         -60.0000
25%           0.0000
50%           1.0000
75%           2.0000
max         912.0000
Name: SalesQuantity, dtype: float64
```

We observe that most of the values are between 0 and 2 with mean value being 2.2466. Data contains extreme values which drives mean value up.

We also present descriptive statistics of product code and store codes:

	ProductCode	StoreCode
count	1873618	1873618
unique	317	340
top	149	331
freq	54060	17470

There are 317 unique products in the dataset. Product 149 has the most records with 54,060 appearances. There are 340 unique store codes and store 331 has the most records with 17,470 appearances.

We will identify the impact of the promotion ons sales by using the percentage change function. We calculate percentage change of sales from the previous week and promotion week for 4 promotions using products and stores separately. Following tables show the head of calculated percentage changes for each promotion per store and product dataframes:

StoreCode	Promo1Response	Promo2Response	Promo3Response	Promo4Response	AverageResponseRate	STDResponseRate
1	26.6854	18.2004	6.3707	20.3125	17.8922	8.4857
2	55.8471	13.6153	-12.9429	43.009	24.8821	30.7955
3	147.5309	12.9577	43.4307	32.5243	59.1109	60.2797
4	80.3536	139.1429	-7.7951	1.8824	53.3959	69.4676
5	76.0383	-9.2593	175.1515	108.0402	87.4927	76.5907

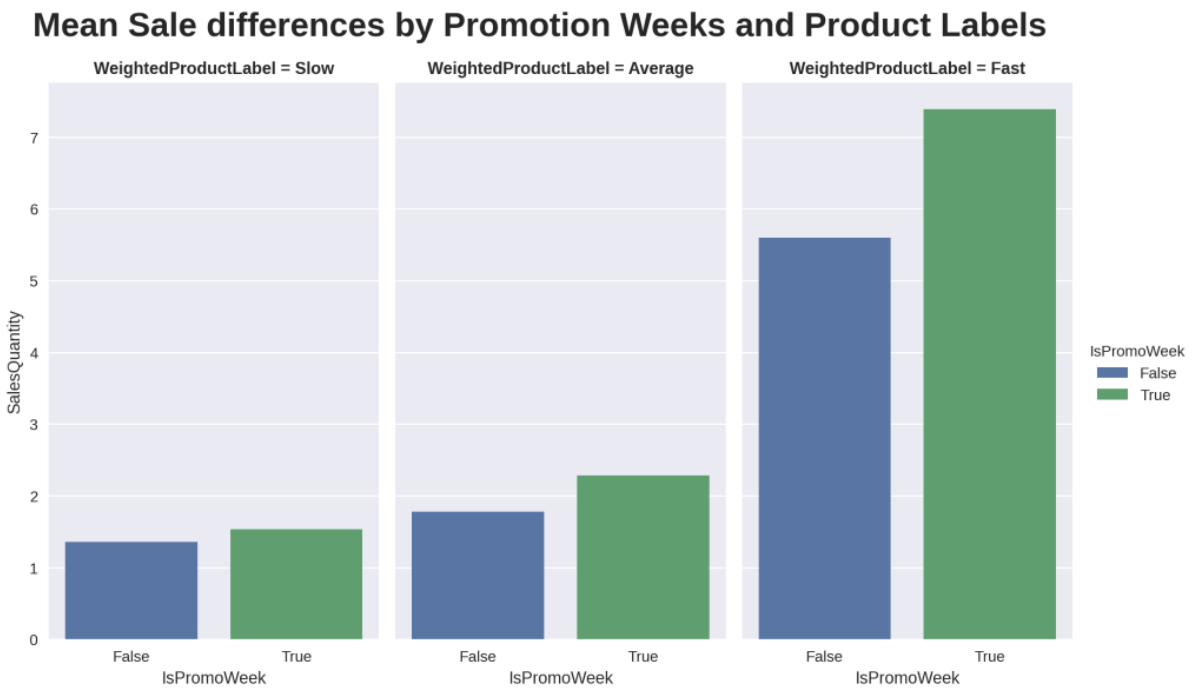
ProductCode	Promo1Response	Promo2Response	Promo3Response	Promo4Response	AverageResponseRate	STDResponseRate
1	1.2165	-0.6424	-7.6687	34.8958	6.9503	19.0192
2	-5.6426	62.0253	35.6401	9.9206	25.4859	29.7181
3	102.9197	53.6458	80	45.977	70.6356	25.9914
4	-2.1352	23.4234	58.75	-9.6639	17.5936	30.8763
5	4.1401	21.2903	9.0909	69.2661	25.9469	29.7652

We define that 50 percent increase in total sales for products, and 75 percent increase in sales for stores on average per promo can be considered as a successful sales increase. We use boolean masks to grab these products and stores. Then biggest sale experiencers are given below:

Products that experienced the most sales increase during promotion weeks: 3, 20, 21, 22, 24, 48, 53, 59, 61, 62, 66, 67, 71, 99, 122, 152, 185, 186, 188, 205, 207, 208, 209, 210, 213, 214, 215, 217, 218, 219, 220, 221, 222, 236, 237, 239, 248, 260, 268, 278, 282, 289, 290, 299, 302, 303, 317.

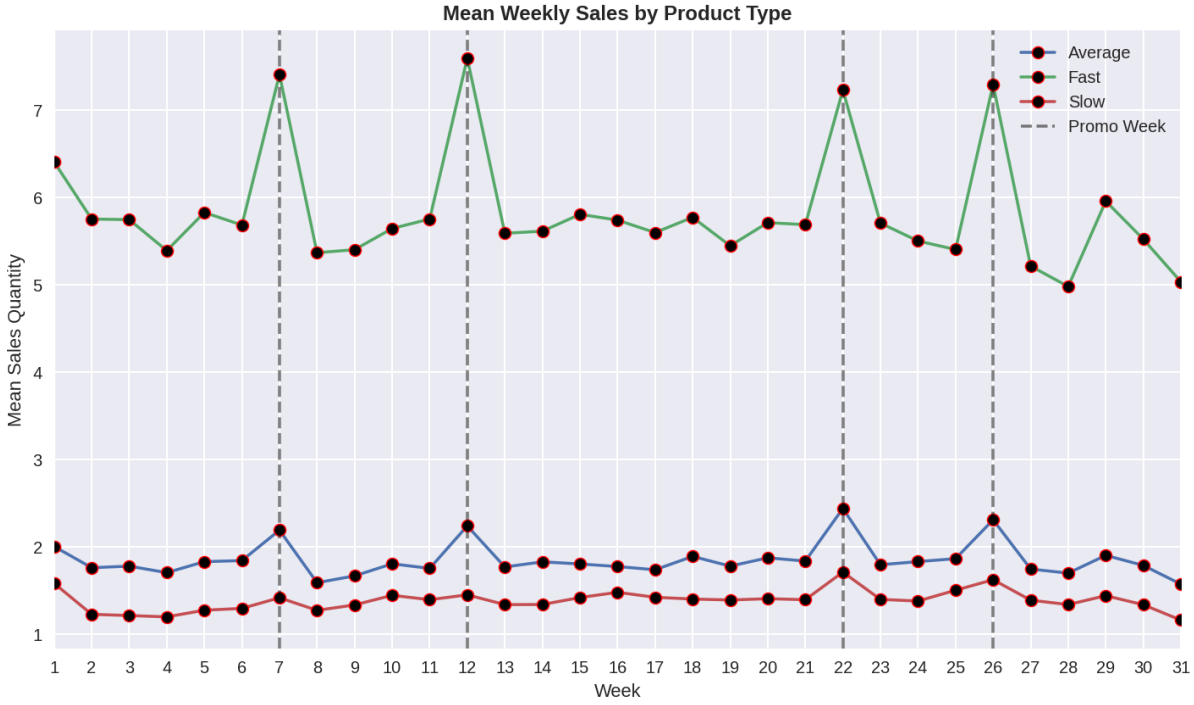
Stores that experienced the most sales increase during promotion weeks: 5, 12, 30, 44, 54, 61, 65, 82, 84, 117, 143, 155, 183, 188, 200, 207, 210, 225, 248, 256, 277, 284, 312, 332.

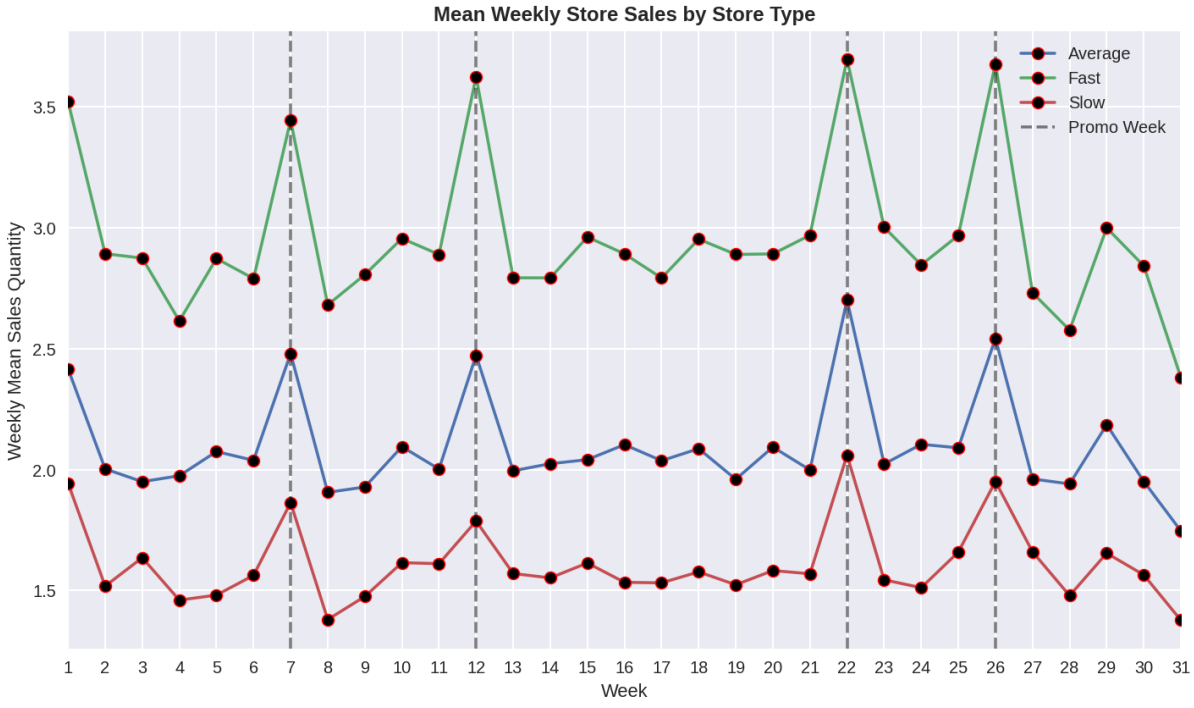
Plots below shows the general promo impact on store sales and product sales by their weighted labels:



We observe that fast stores and products are more responsive to the promotion weeks.

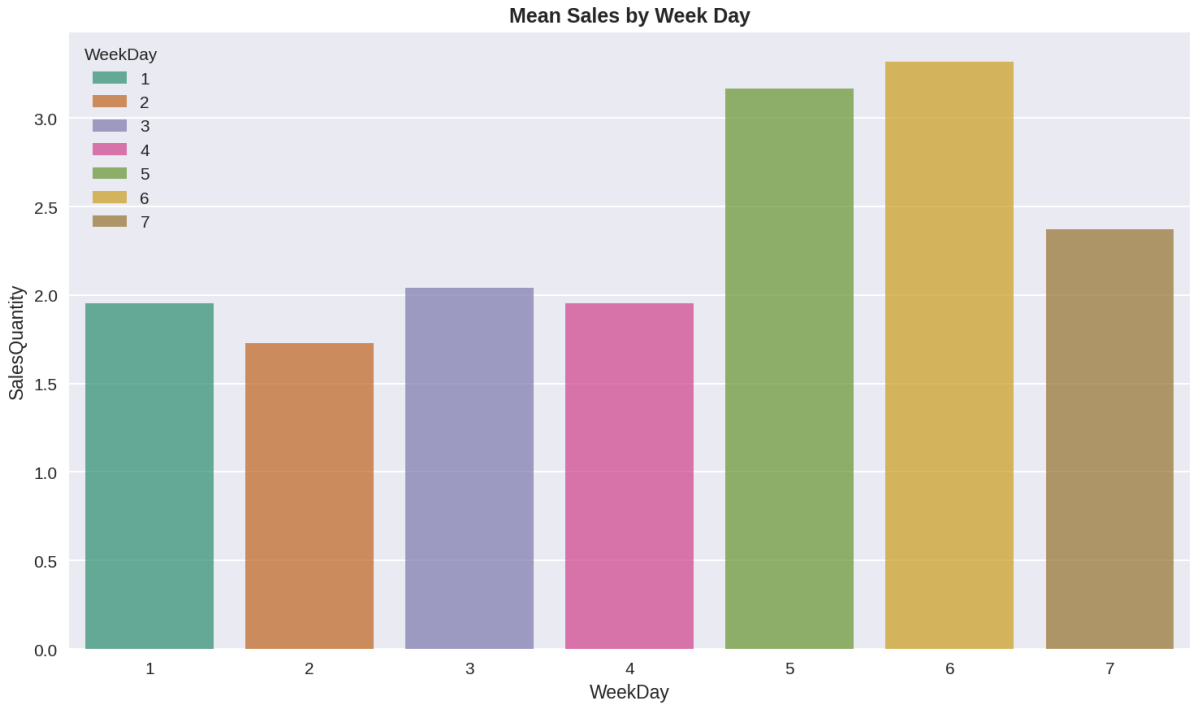
To further fortify this finding, We plot the weekly mean sales of products and stores:

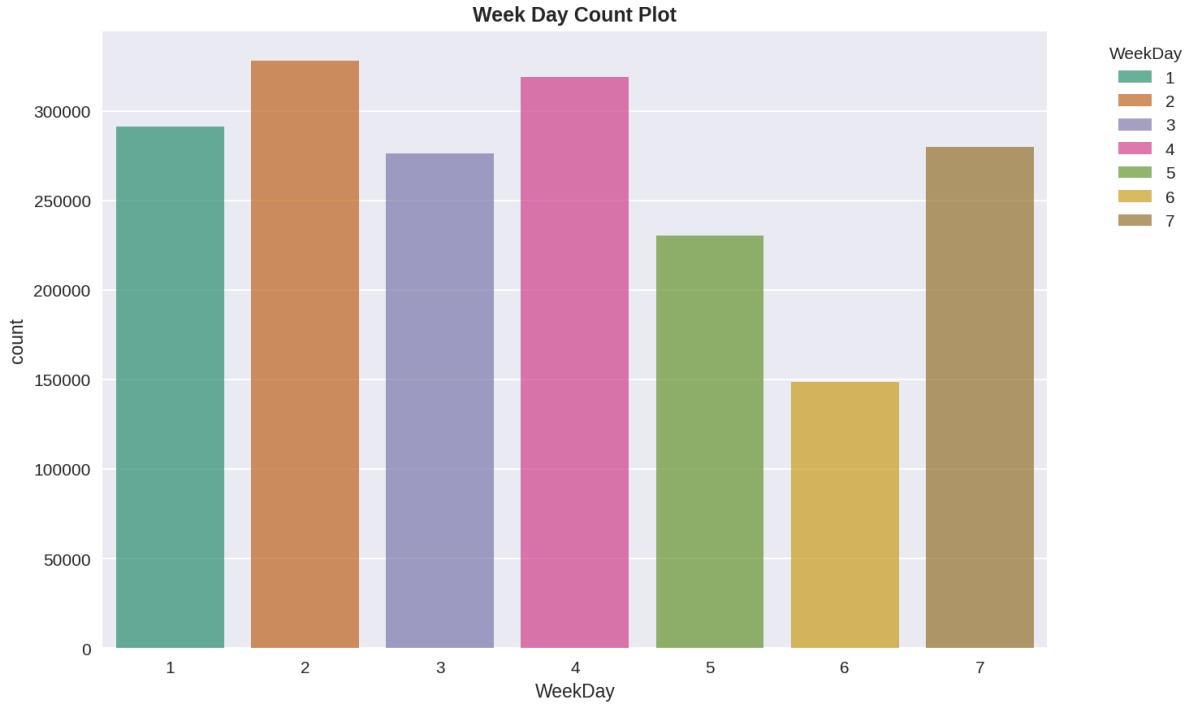




The figure above suggests that the product separation algorithm is not performing properly. On the other hand, we can visually see that store clustering performed satisfactorily. We observe that most of the mean sales increase are observed in fast stores and products. We can conclude that fast stores and products have the highest response rate to the promotions.

We also analyze the mean sales by weekdays:





- Most of the sales happened on Tuesday, and the least sales happened on Saturday.
- Friday and Saturday have the highest mean sales.

We also try to analyze the change in item returns after the promotions. However we could not find a significant difference after limiting the data only for the returns and grouping by post promotion weeks:

PostPromo	SalesQuantity
False	-2.1065
True	-2.2424

4. MACHINE LEARNING

We will now build a machine learning model that predicts the quantity of sales using week, store and product pairs' mean sale values. Also we add promotion week information and product and store labels as dummy variables. We drop WeekNumber, StoreCode, ProductCode columns after merging labels. Finalized train dataframe's head is given below:

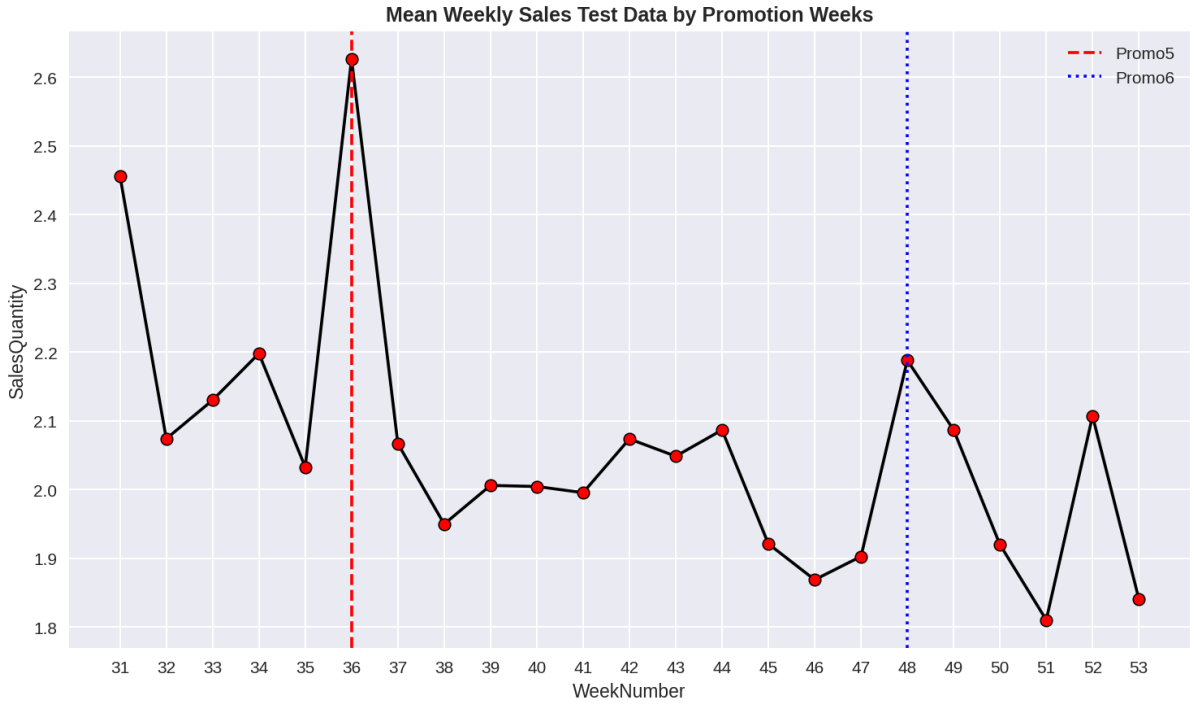
SalesQuantity	IsPromoWeek	WeightedProductLabel_Fast	WeightedProductLabel_Slow	WeightedStoreLabel_Fast	WeightedStoreLabel_Slow
1	FALSE	FALSE	FALSE	FALSE	FALSE
1	FALSE	FALSE	FALSE	FALSE	FALSE
1	FALSE	FALSE	TRUE	FALSE	FALSE
1	FALSE	FALSE	FALSE	FALSE	FALSE
1	FALSE	FALSE	TRUE	FALSE	FALSE

Where,

X = IsPromoWeek,WeightedProductLabel_Fast,WeightedProductLabel_Slow ,WeightedStoreLabel_Fast, WeightedStoreLabel_Slow, and

y = SalesQuantity.

We prepare the test data from assignmen4.1b.csv using similar methods. Please note that we try to analyze the impact of promo5 on the sales. Therefore, we limit test data to week 36. The plot below shows the test data weekly mean sales:



We use random forest regression to capture the complex variance in the dataset. We set up randomize cross validation search with parameter distribution below with 100 iterations:

```
param_dist = {
     'n_estimators': np.arange(50, 201, 10),
     'max_depth': [None, 10, 20, 30, 40, 50],
     'min_samples_split': np.arange(2, 11),
     'min_samples_leaf': np.arange(1, 5),
     'max_features': ['auto', 'sqrt'],
     'bootstrap': [True, False],
}
```

Details of the grid search are disclosed in grid_search_results.xlsx. Best estimator results are given below:

```
Best Parameters: {'n_estimators': 60, 'min_samples_split': 6,
'min_samples_leaf': 2, 'max_features': 'sqrt', 'max_depth':
40, 'bootstrap': True}
Best Score: -10.140995074270476
```

Features importances are given below:

```
{'IsPromoWeek': 0.02532836490789763,
 'WeightedProductLabel_Fast': 0.7710079178694155,
 'WeightedProductLabel_Slow': 0.08420238503037962,
 'WeightedStoreLabel_Fast': 0.0888331511644902,
 'WeightedStoreLabel_Slow': 0.030628181027817183}
```

We use test data to make predictions on promo 5 week. Results are given below:

```
Mean absolute error: 2.000469732497644
Root mean squared error: 4.090640985076
```

Y_test data descriptive statistics:

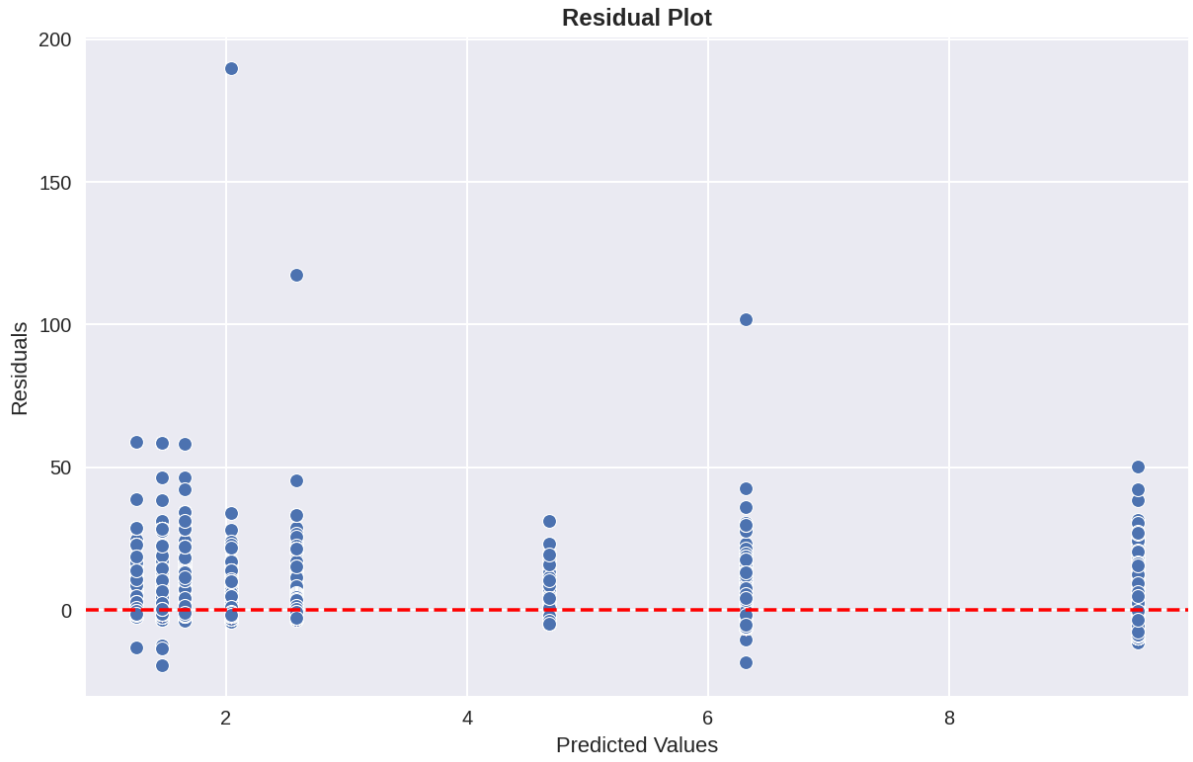
```
count          30209.0000
mean               2.2006
std                4.3141
min              -18.0000
25%                0.5000
50%                1.0000
75%                2.0000
max              192.0000
Name: SalesQuantity, dtype: float64
```

We also share prediction descriptive statistics:

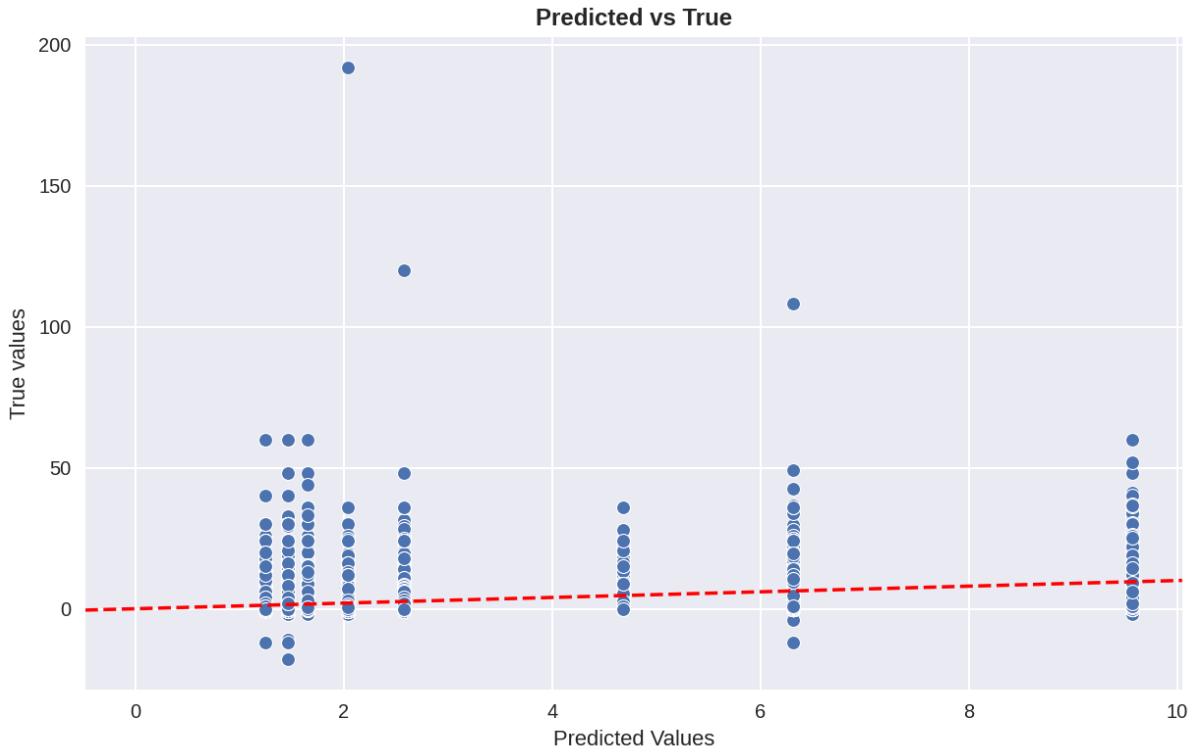
```
count          30209.0000
mean               2.1736
std                 1.5611
min                1.2502
25%                1.4648
50%                1.6571
75%                2.0433
max                9.5680
dtype: float64
```

Note that predictions have very little range compared to the test data values. Which causes root mean squared error to go above the mean absolute error. We disclose the error measurement metric plots below:

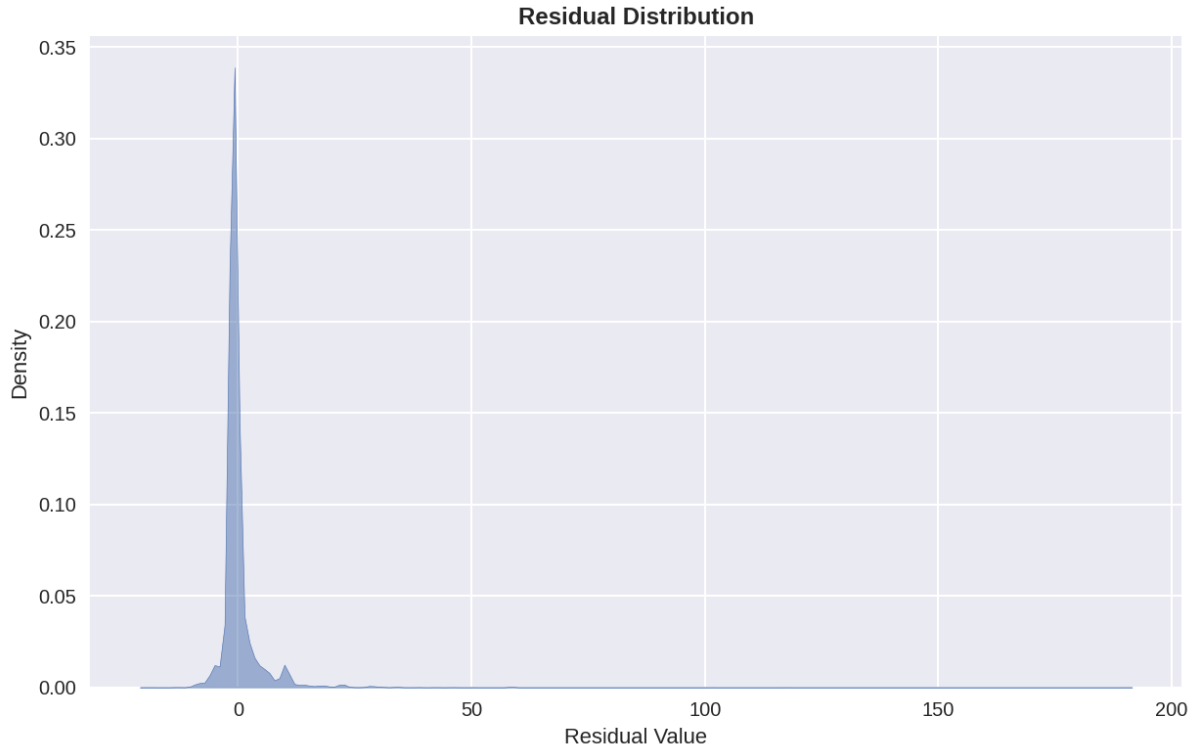
Residual Plot:



Predicted vs True value Plot:



Residual Distribution Plot:



After careful observation of the plots and descriptive statistics we observe that the model failed to capture promotion impact on the sales. In fact, our model was underpredicting the promotion impact. We will make following suggestions to improve the predictive power of the model for the future researches:

- Considering products and stores separately for the grouping. This would allow us to generalize sales and reduce complexity.
- Consider using a weekly date column. We could use a rolling mean function to smooth sales and make more accurate predictions.
- We could separate returns and sales in order to reduce information confusion.
- We could use more clusters (or even a different algorithm) and dummy variables to capture variances in the sales.
- We observe that each promotion is distinct and does not reoccur. This causes us to use general promotion week dummy variable instead of distinct dummies. If test data contains trained promotions we could drill down to the specific promotion impact during the modeling phase.
- We could use different machine learning models, or increase parameter space in grid searches in order to train better models.

5. CONCLUSIONS AND RECOMMENDATIONS

After detailed analysis we we will make the following recommendations:

- We have observed that the fast products and stores have higher response to the promotions in general. We suggest that shifting focus and resources to such products and stores will optimize the sales. We have also disclosed products with 50% sale increase on average and stores with 75% sale increase on average during promotions.
- We compared the average returns between post promotion week (1 week after promotion) and other weeks, and failed to find a significant difference. Therefore, no action is recommended.
- We tried to build a random forest regression model that predicts the mean sale value using week, store code and product code pairs as grouping. However, the model failed to capture promotion jumps. In other words, the model was underpredicting jumps in sales during promotions. We make recommendations on how to avoid problems that stem from the dataset.