Capstone 3 Project Report:
Store Item Demand Forecasting

Problem statement:

The objective of this project is to predict three months of sales for each store and item. The goal is to analyze any patterns in the sales data like seasonality, whether any relationship exists between different stores or items. Different forecasting methods will be used to forecast the sales and performance will be evaluated using SMAPE.

Context:

Accurate sales prediction is a crucial element for the operational success of major retail players like Walmart, Target, and Costco. It enables these companies to strategically manage inventory by making informed decisions on which products to stock and replenish. A robust forecasting system not only enhances supply chain efficiency but also provides a competitive advantage in the fiercely competitive retail landscape. Given the abundance of choices for each product and the dynamic nature of the market, precise anticipation of customer demand becomes instrumental in outperforming other retailers.

Data:

Data comprises of two csv files ‘train.csv’ and ‘test.csv’. As the names suggest the ‘train.csv’ will be used for training the forecasting model and ‘test.csv’ will be used to test the model performance. The data comprises of 4 columns (features) including ‘date’, ‘store’, ‘item’ and ‘sales’. The training data is big (17 MB) and the testing data is small as expected (1 MB). The ‘data’ column lists the date of the sale. The ‘store’ column lists the store id and ‘item’ lists the item id. The ‘sales’ column lists the number of items sold that that day.

Since we do not have the ‘Sales’ information from the ‘test.csv’ file, we will use last three months from “train.csv” as test data. That way we have a way to validate our prediction and calculate the errors in prediction.

Data Wrangling:

The original dataset was loaded and there were four columns and 913000 rows or instances. The four columns are: ‘date’, ‘store’, ‘item’ and ‘sales’. Next the datatype for the columns were analyzed. The column ‘date’ was of Dtype ‘object’ and rest were of type ‘int’. Next, I looked at the simple statistics for the three columns of integer type.

	store	item	sales
count	913000.00	913000.00	913000.00
mean	5.50	25.50	52.25
std	2.87	14.43	28.80
min	1.00	1.00	0.00
25%	3.00	13.00	30.00
50%	5.50	25.50	47.00
75%	8.00	38.00	70.00
max	10.00	50.00	231.00

Table 1: Basic statistics for the three integer type columns for the entire dataset.

It seems like ‘store’ stores the store_id and ‘item’ stores the ‘item_id’ that were sold on ‘date’. So each row indicates a transaction which stores the information of the store where the sale was made and the item. Next, the number of unique stores and items were examined in the dataset.

Below is a list of unique store ids in the training data and their representation:

1	91300
2	91300
3	91300
4	91300
5	91300
6	91300
7	91300
8	91300
9	91300
10	91300

Below is a list of unique items in the training data and their representation:

1	18260
2	18260
3	18260
4	18260
5	18260
6	18260
7	18260
8	18260

9	18260
10	18260
11	18260
12	18260
13	18260
14	18260
15	18260
16	18260
17	18260
18	18260
19	18260
20	18260
21	18260
22	18260
23	18260
24	18260
25	18260
26	18260
27	18260
28	18260
29	18260
30	18260
31	18260
32	18260
33	18260
34	18260
35	18260
36	18260
37	18260
38	18260
39	18260
40	18260
41	18260
42	18260
43	18260
44	18260
45	18260
46	18260
47	18260
48	18260
49	18260
50	18260

Table 2: Unique store ids and item ids check

There were a total of 10 stores and a total of 50 items were sold in each store. The number of sale instances were same for each store and item combination. The data has a very uniform distribution and is likely synthetic dataset. Next the dataset was checked for any null values and there were none found. Very clean dataset and ready for analysis.

Exploratory Data Analysis:

After data wrangling exploratory data analysis was done. Specifically, the following steps were taken to analyze the data:

- Convert time variable to pandas datetime object
- Generate other columns for date such as day of the week, month, year etc.
- Check total time range: start date, end date, total number of days
- Check for missing days
- Generate summary stats for each store
- Generate summary stats for each item
- Plot histogram of daily sales for each store and all stores combined
- Plot box plots for each day of the week for each store and all stores combined
- Time plot of daily sales for each store
- Time plot of daily sales for each store with a moving average window
- Time plot of weekly sales for each store
- Time plot of monthly sales for each store
- Correlation plot for daily sales from different stores
- Check for any degeneracies in the data

Date analysis:

The ‘Date’ column was already a pandas date time object. The start date for the dataset is 2013-01-01 to 2017-12-31. There were a total of 1826 days and there was an entry for sales for each day for each store and item. There were no missing days in the dataset.

Using the ‘Date’ other date related features such as year, month, day of the year and day of the week were extracted and added to the dataframe.

Summary stats for each store and item:

The dataset was grouped by store and the sales of all items in the store were summed and summary statistics were generated. The process was repeated for each item and the summary statistics for the sales of each item across all stores were generated. The following table lists the store-wise and item-wise summary statistics.

Summary statistics for sales at each store

	store	count	sum	mean	median	std	min	max
0	1	91300	4315603	47.268379	44.0	24.006252	1	155
1	2	91300	6120128	67.033165	62.0	33.595810	3	231
2	3	91300	5435144	59.530602	55.0	29.974102	3	196
3	4	91300	5012639	54.902946	51.0	27.733097	4	186
4	5	91300	3631016	39.770164	37.0	20.365757	2	130
5	6	91300	3627670	39.733516	37.0	20.310451	0	134
6	7	91300	3320009	36.363735	34.0	18.684825	1	122
7	8	91300	5856169	64.142048	60.0	32.231751	4	204
8	9	91300	5025976	55.049025	51.0	27.832186	4	195
9	10	91300	5360158	58.709288	54.0	29.554994	3	187

Summary statistics for sales for each item

	item	count	sum	mean	median	std	min	max
0	1	18260	401384	21.981599	21.0	8.468922	1	59
1	2	18260	1069564	58.574151	56.0	20.093015	9	150
2	3	18260	669087	36.642223	35.0	13.179441	7	104
3	4	18260	401907	22.010241	21.0	8.403898	0	66
4	5	18260	335230	18.358708	18.0	7.265167	1	50
5	6	18260	1068281	58.503888	56.0	20.174898	11	148
6	7	18260	1068777	58.531051	56.0	20.146002	11	141
7	8	18260	1405108	76.950055	74.0	26.130697	15	181
8	9	18260	938379	51.389869	49.5	17.790158	6	134
9	10	18260	1337133	73.227437	70.0	24.823725	14	175
10	11	18260	1271925	69.656353	67.0	23.744732	11	170
11	12	18260	1271534	69.634940	67.0	23.738663	12	170
12	13	18260	1539621	84.316594	81.0	28.311031	20	210
13	14	18260	1071531	58.681873	56.0	20.079860	12	152
14	15	18260	1607442	88.030778	85.0	29.522852	17	231
15	16	18260	468480	25.656079	25.0	9.603270	2	70
16	17	18260	602486	32.994852	32.0	11.967610	4	83
17	18	18260	1538876	84.275794	81.0	28.430621	18	208
18	19	18260	736892	40.355531	39.0	14.332645	5	99
19	20	18260	867641	47.515936	46.0	16.490487	9	127
20	21	18260	736190	40.317087	39.0	14.338006	7	109
21	22	18260	1469971	80.502245	78.0	27.118163	14	214
22	23	18260	534979	29.297864	28.0	10.819549	3	81
23	24	18260	1205975	66.044633	64.0	22.531555	14	156
24	25	18260	1473334	80.686418	78.0	27.238817	18	193
25	26	18260	869981	47.644085	46.0	16.723912	8	119
26	27	18260	402628	22.049726	21.0	8.461641	1	59
27	28	18260	1604713	87.881325	85.0	29.501781	16	206
28	29	18260	1271240	69.618839	67.0	23.635631	15	173
29	30	18260	736554	40.337021	39.0	14.363331	5	115
30	31	18260	1070845	58.644304	57.0	20.104705	10	159
31	32	18260	803107	43.981763	42.0	15.574556	5	119
32	33	18260	1270183	69.560953	67.0	23.718343	15	169
33	34	18260	469935	25.735761	25.0	9.617910	2	79
34	35	18260	1201541	65.801807	63.0	22.461990	12	168
35	36	18260	1406548	77.028916	74.0	26.067440	16	188
36	37	18260	534258	29.258379	28.0	10.771547	3	74
37	38	18260	1470330	80.521906	77.0	27.141799	15	188
38	39	18260	801311	43.883406	42.0	15.511550	7	112

39	40	18260	534094	29.249398	28.0	10.822959	3	74
40	41	18260	401759	22.002136	21.0	8.402470	2	60
41	42	18260	669925	36.688116	35.0	13.215112	5	96
42	43	18260	936635	51.294359	49.0	17.801008	9	126
43	44	18260	536811	29.398193	28.0	10.797738	3	78
44	45	18260	1471467	80.584173	78.0	27.318402	18	205
45	46	18260	1070764	58.639869	56.0	20.220879	11	150
46	47	18260	401781	22.003341	21.0	8.420102	2	61
47	48	18260	937703	51.352848	49.0	17.881917	8	130
48	49	18260	535663	29.335323	28.0	10.874788	3	77
49	50	18260	1203009	65.882202	63.0	22.416031	12	164

Table 3: Store-wise and item-wise summary statistics

Next the total sales for each store and each item were computed and were plotted. Below, is a plot that shows the total sales.

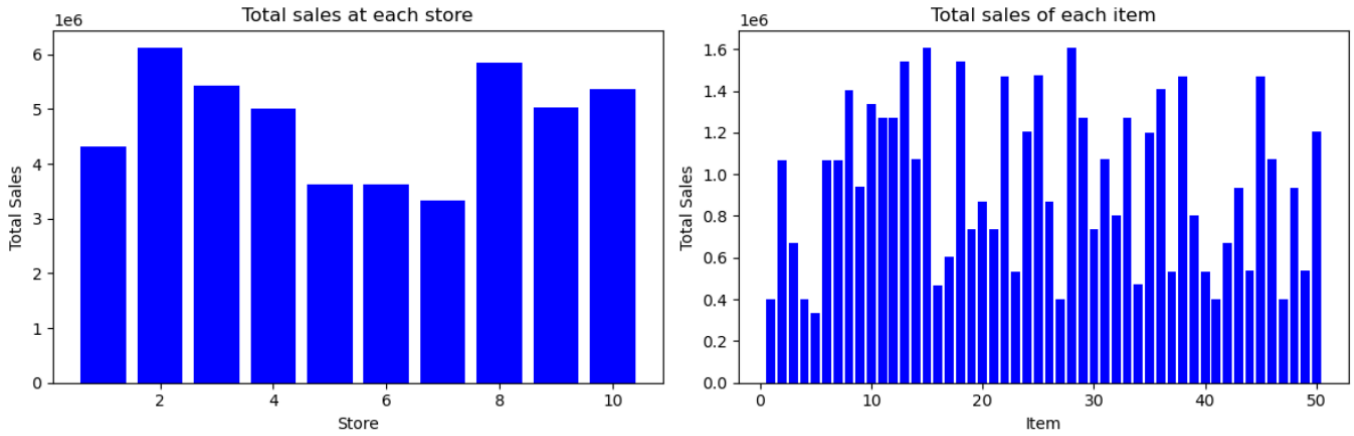


Figure 1: Store-wise and item-wise summary statistics

Daily sales analysis:

Daily sales data was generated by grouping the data by date. Total sales for each store individually and all store combined were generated. Histograms were plotted to understand the distribution of the daily sales data. From the Histograms shown in Figure 2 we can see that the distributions were mostly normal and were also fairly similar to each other. It would have been interesting if the distributions for some of the stores were different from others. It seems like we could choose any one store for our analysis.

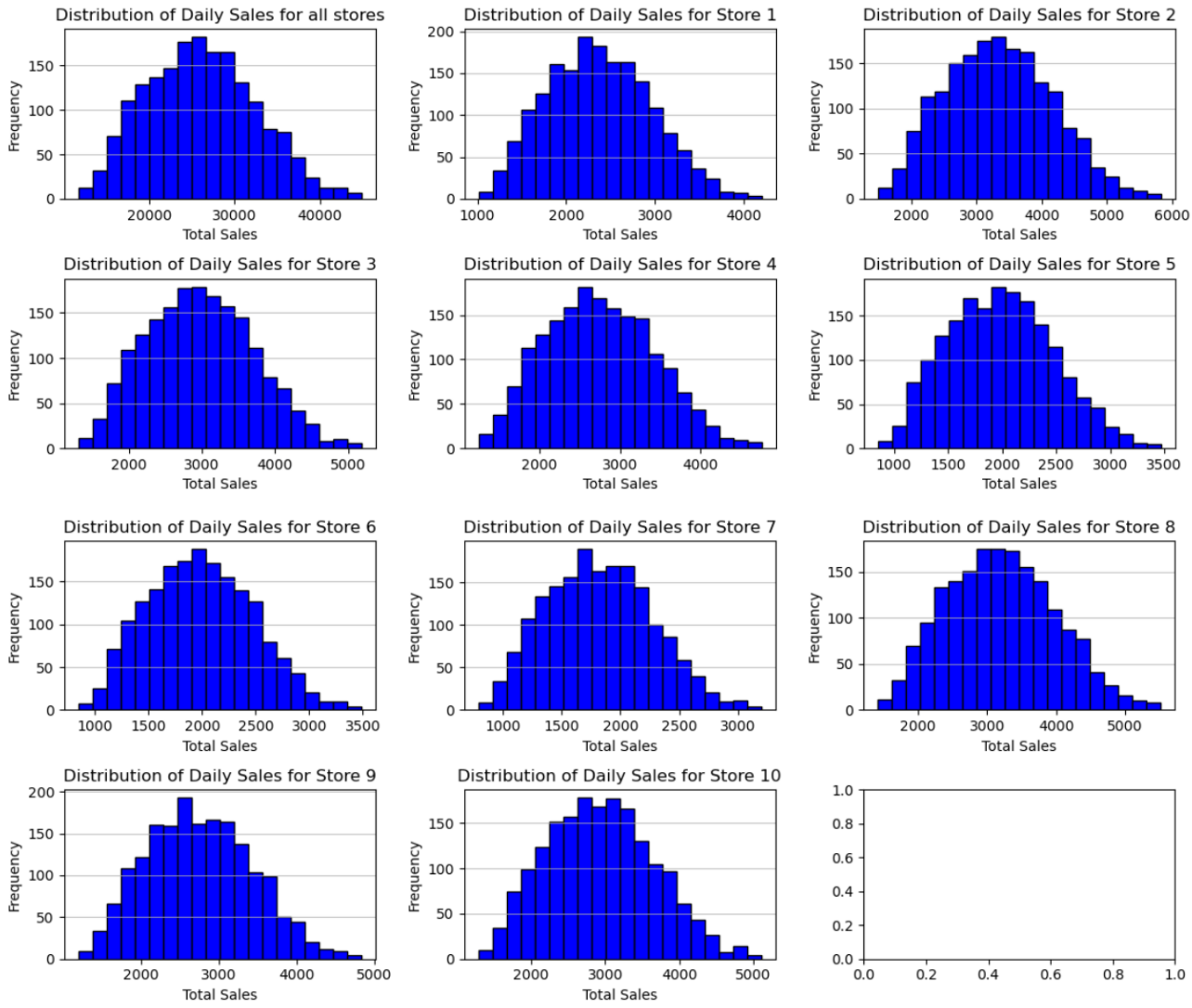
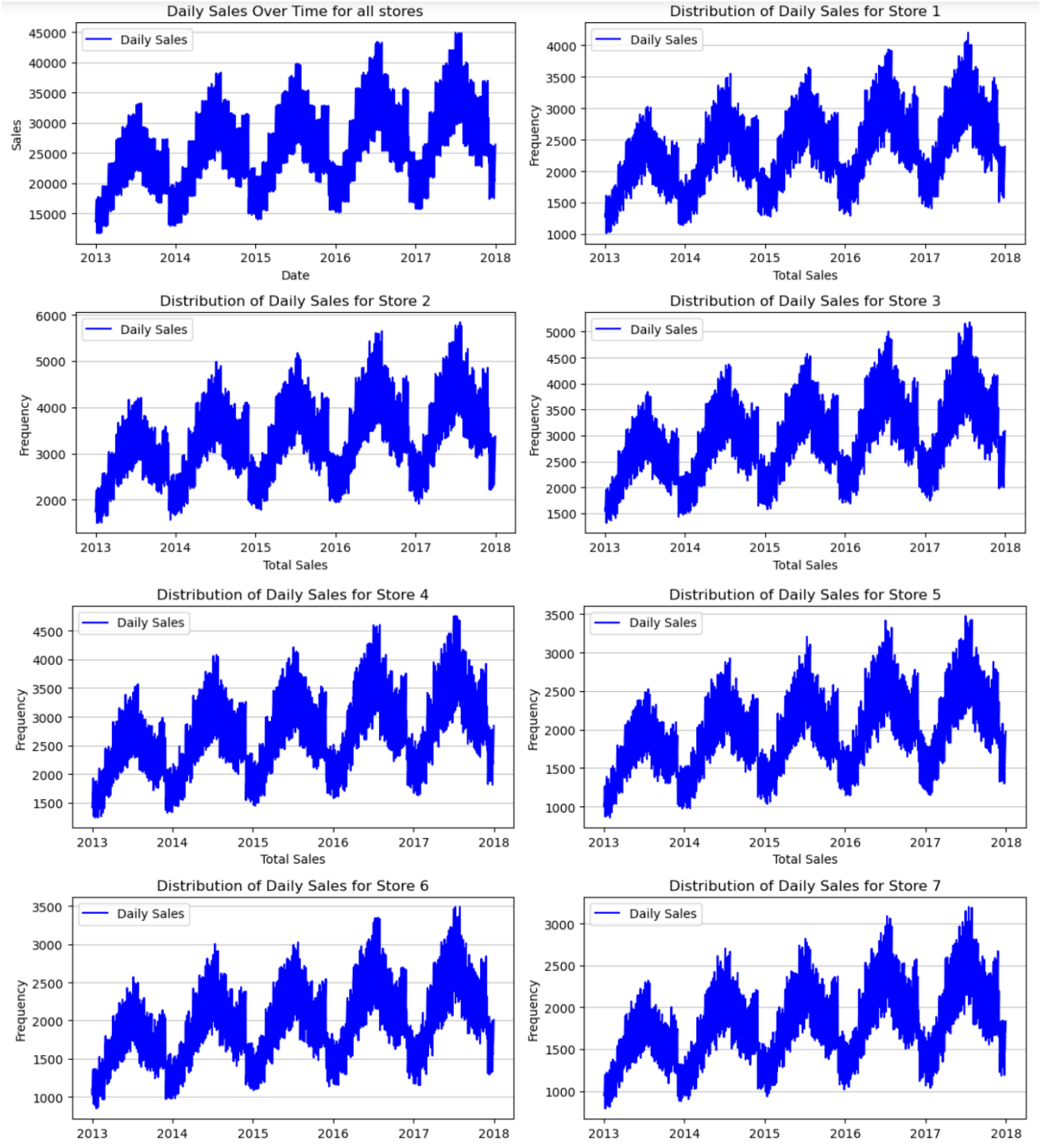


Figure 2: Histograms of daily sales data for each store and all stores combined.

Next, we generated time series plots for the sales data for all items for each store and all stores combined. Figure 3 shows the time series plots. From a visual inspection the time series plots are very similar. There is a gradual positive slope which indicates the sales have an increasing trend yearly. For a particular year the sales are start low and are highest around July and then again are lower. The trends are overall quite consistent across various stores. There could be other potential trends and they will be analyzed when we decompose the time-series plot.

Moving average of a time series, filters the high frequency variations in the data and shows the trends better. Figure 4 shows moving average of Store 1 sales data with different window lengths: 7days, 14 days, 30 days.


Daily Sales Over Time for all stores
Daily Sales
Sales
Date
Distribution of Daily Sales for Store 1
Frequency
Total Sales
Distribution of Daily Sales for Store 2
Distribution of Daily Sales for Store 3
Distribution of Daily Sales for Store 4
Distribution of Daily Sales for Store 5
Distribution of Daily Sales for Store 6
Distribution of Daily Sales for Store 7
2013
2014
2015
2016
2017
2018

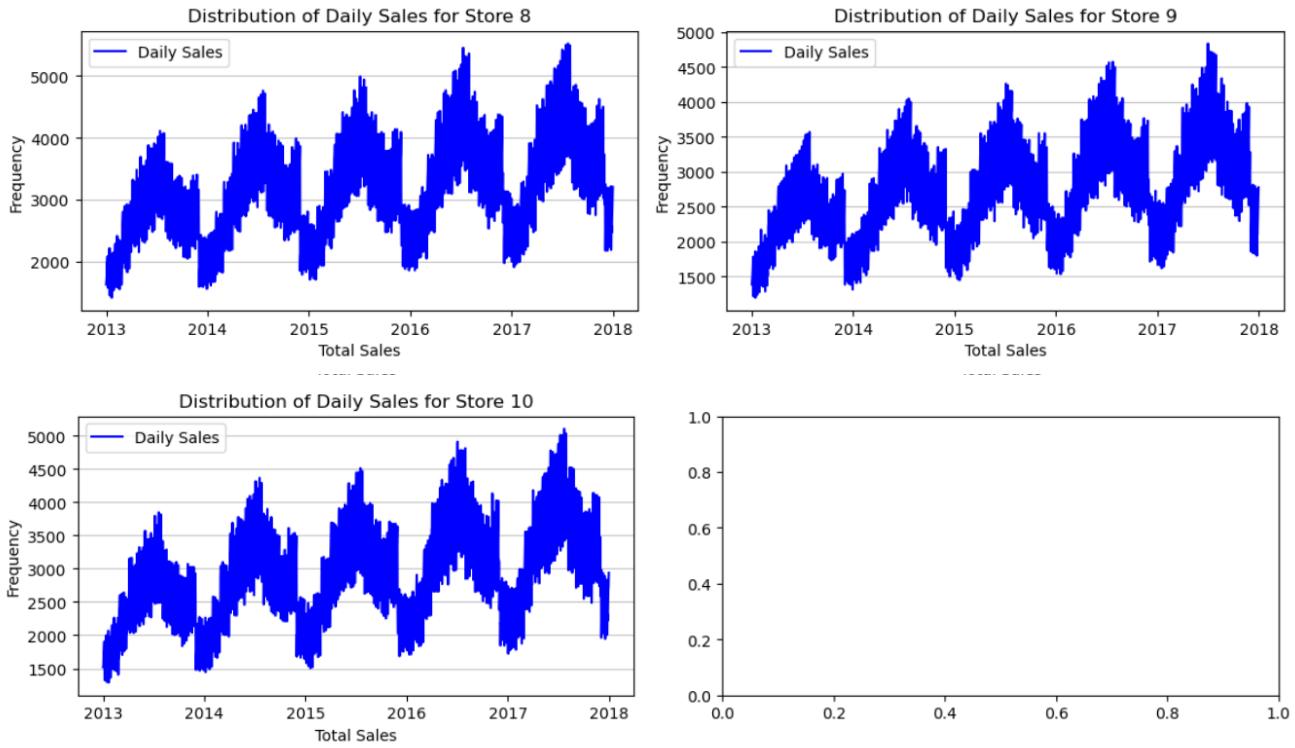


Figure 3: Time series plots of the daily sales data for each store and all stores combined.

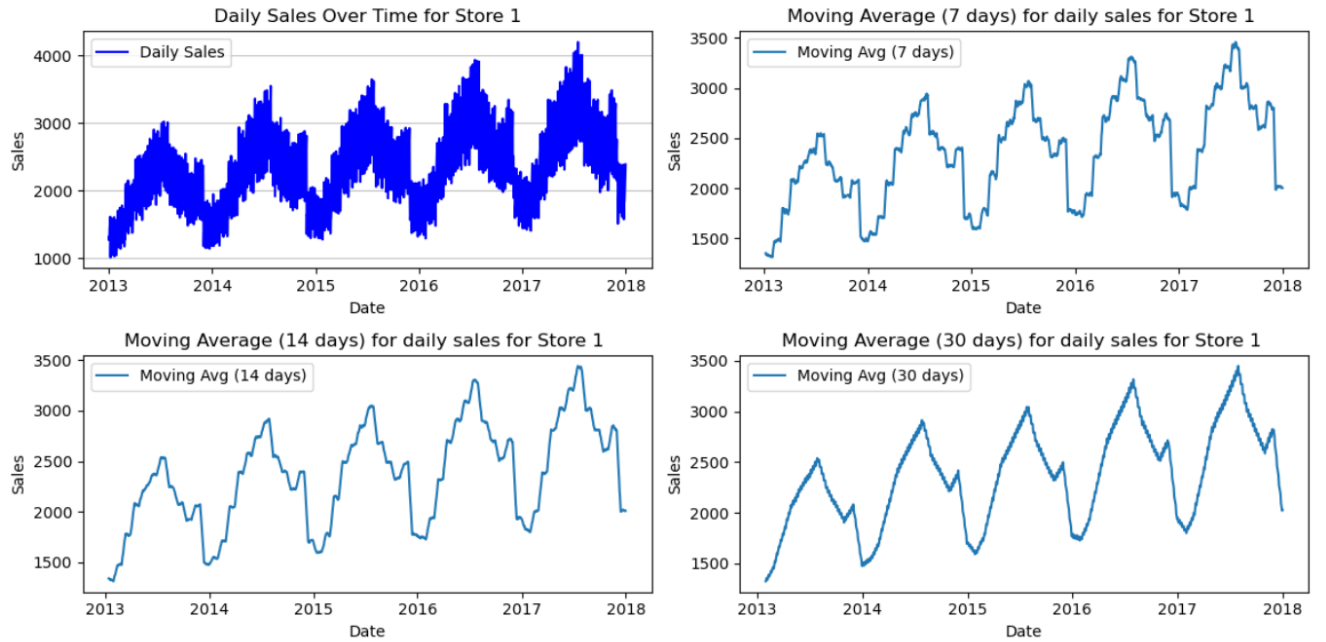


Figure 4: Moving average plots of daily sales data for Store 1.

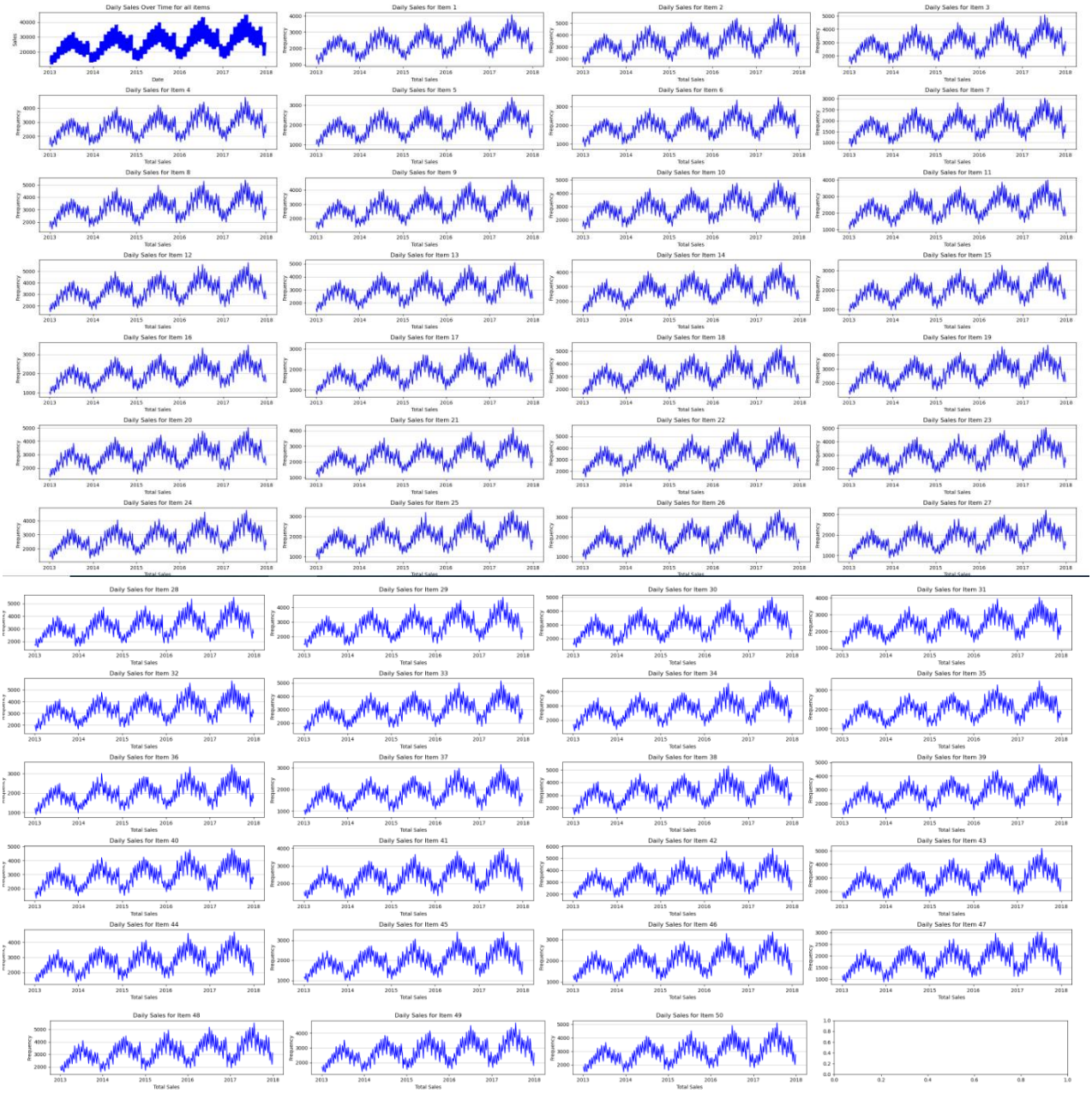


Figure 5: Moving average plots of daily sales data for each item.

Figure 5 shows the time series plot of sales data for each item (across all the stores). The objective is to see any patterns in the data for different items. As we can see that the sales data is quite consistent for all the items across all the stores. As we assumed earlier there is more evidence that the data is a synthetic dataset generated with the same parameters for each store and each item plus some noise. Next, we will check for any degeneracies in the data.

Checking of degeneracies:

In order to understand the degeneracies or inter-dependencies in the data I computed mean relative sales for each item over the years. The same calculation is repeated for relative sales of each store over the years. This shows the yearly sales pattern for all the items and stores and as expected they follow a similar trend.

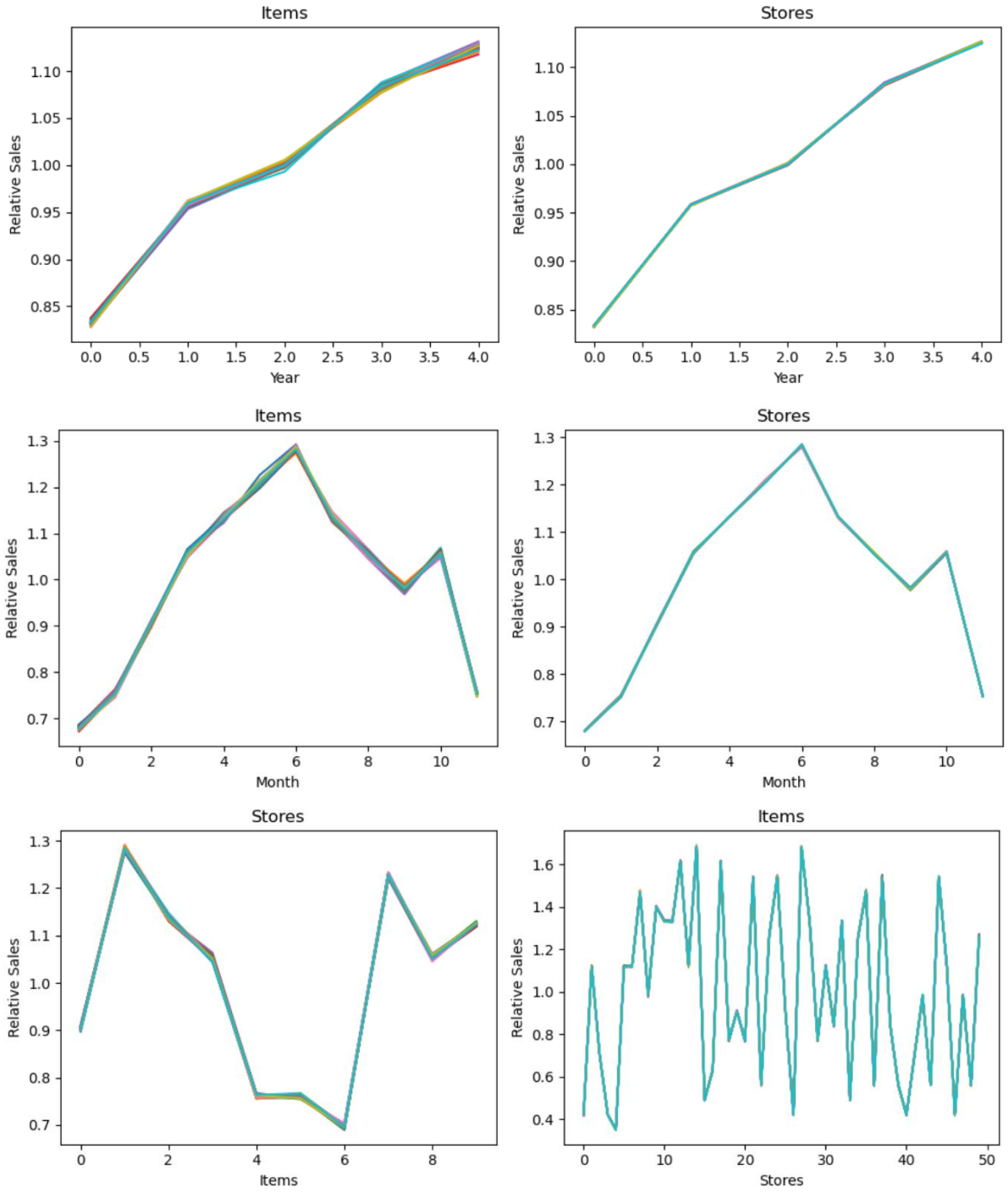


Figure 6: Relative sales trend for items and stores over time.

Figure 6 shows the plots of the relative sales over years and also over months. We observe a very similar trend for all items and stores. The relative sales for items for each store were also checked and again no apparent degeneracies were found in the data.

Seasonal decomposition:

The time series sales plot is decomposed into trend, seasonality and residual using an additive assumption and multiplicative assumption. The decomposition plots are shown in Figure 7 and Figure 8. From the decomposition plots we can see that the additive model gives a better estimate of trend and seasonality because the residuals are more normally distributed and random.

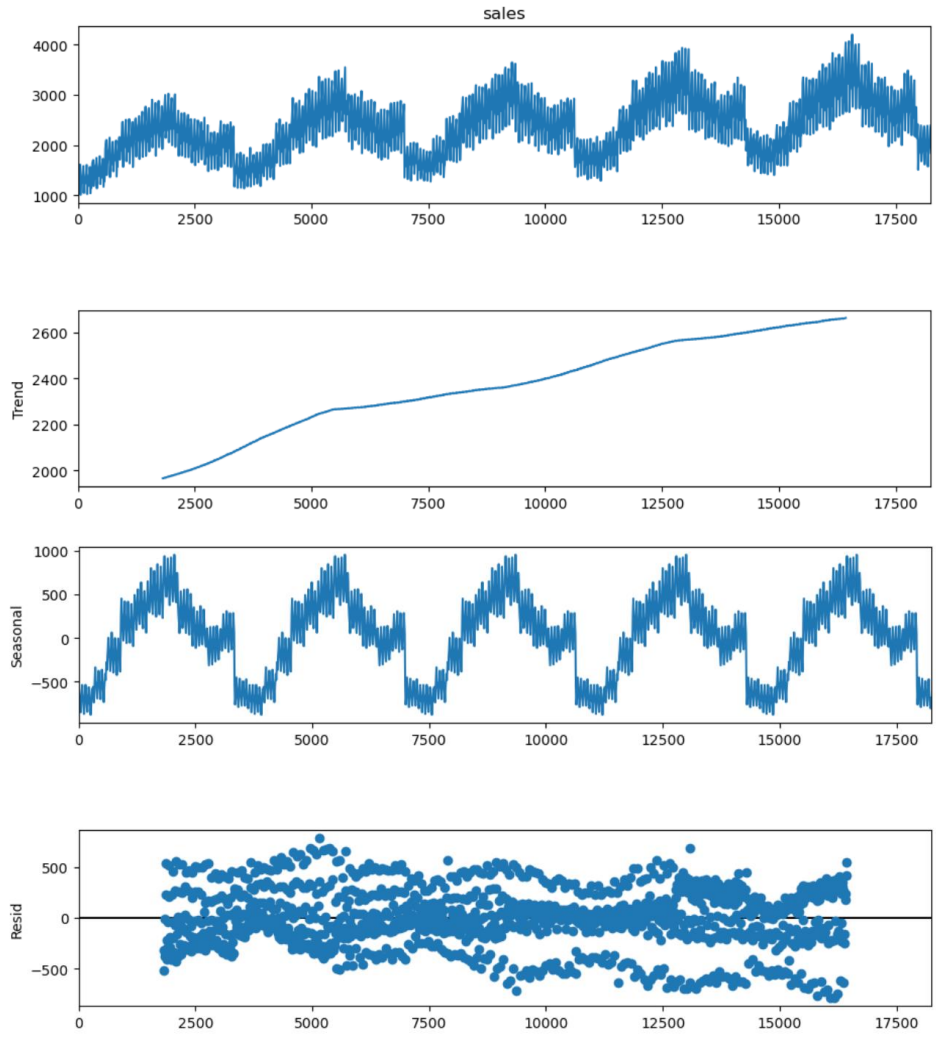


Figure 7: Seasonal decomposition plot assuming an additive model.

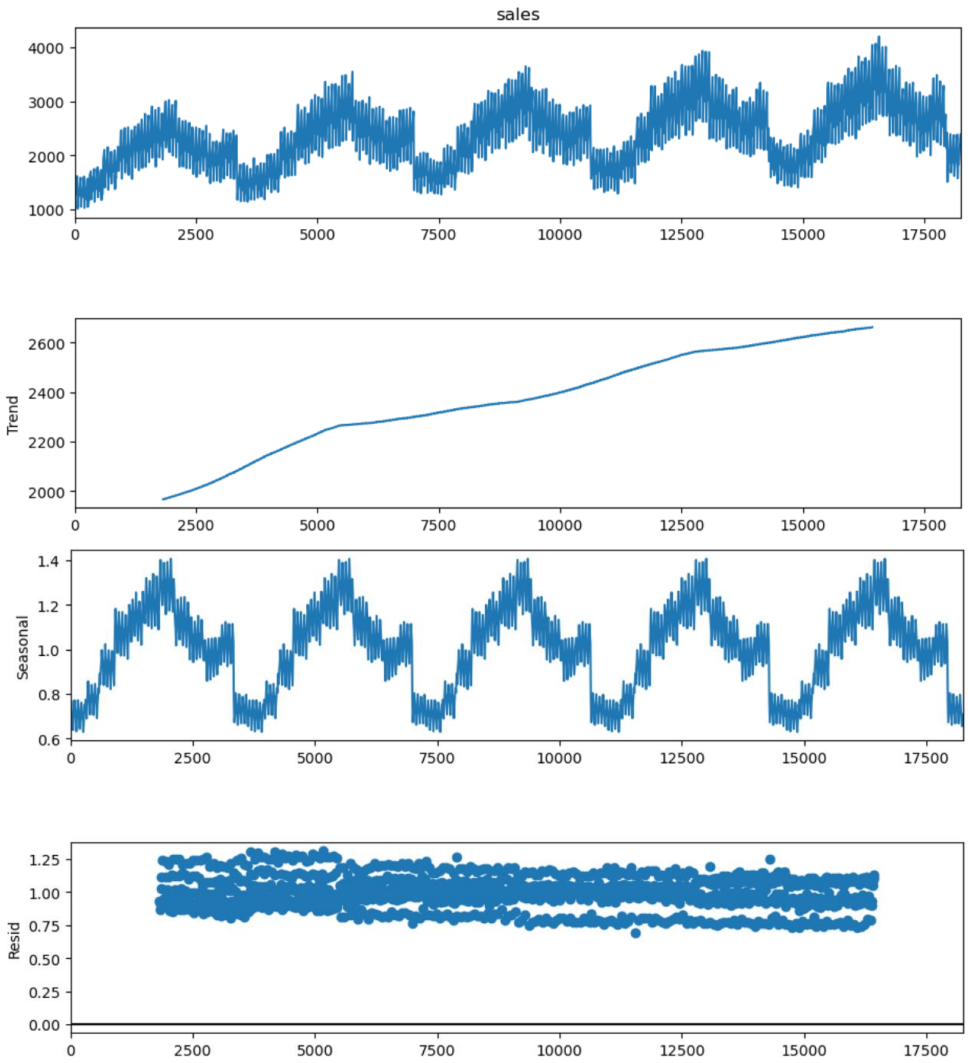


Figure 8: Seasonal decomposition plot assuming a multiplicative model.

Data Preprocessing and Modeling:

After EDA, data preprocessing and modeling were conducted which includes the following steps:

- Generate one dataset for time series prediction
    - Timeseries1 = Store 1 item 1
- Divide the data into training data and test data based on date
- For ARIMA model, estimate parameters p,d,q
    - p is the order of the autoregressive component (AR)

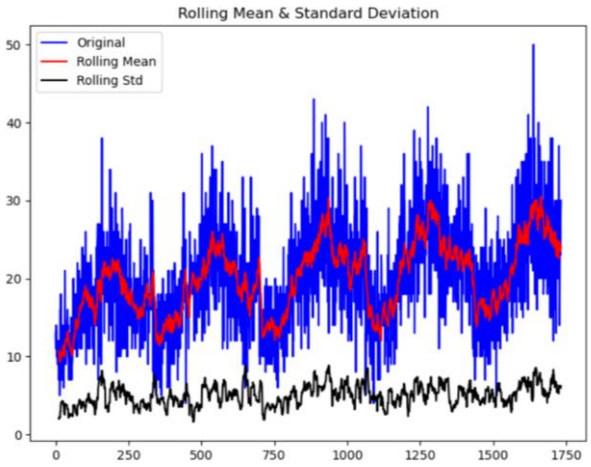
    - q is the order of the moving average component (MA)
    - d is the order of the differencing required to make the time-series stationary
- Estimate p and q from the ACF and PACF plots
- Test for stationarity using Dickey-Fuller test
- If the data is not stationary then test again for first difference
- Continue taking nth difference until the data is stationary
- Model using ARIMA (p,d,q)
- Define an error function to calculate error in prediction
- Derive best parameters of p,d,q by cross-validation
- Test SARIMA model for seasonality effects
- Test SARIMA using exogenous parameters

Generate training and test data:

From EDA it was quite apparent that if we could predict for one store and one item we could use a very similar model or procedure to predict other store-item combinations. To start with we generated daily sales data for store 1 and item 1. The data was split by time (2017-10-01). Last three months of data will be used for test and rest of the data will be used to train the models.

Test for stationarity and estimate d:

Dickey-Fuller test was performed on the time series data to test for stationarity.

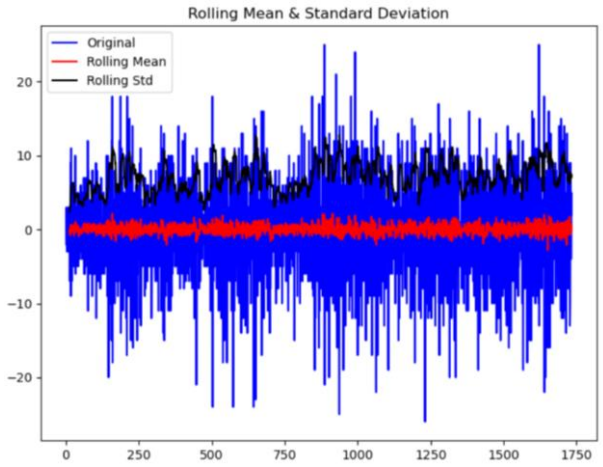



```
Results of Dickey-Fuller Test:
p-value = 0.0413. The series is likely non-stationary.
Test Statistic                  -2.936551
p-value                          0.041260
#Lags Used                      23.000000
Number of Observations Used    1710.000000
Critical Value (1%)             -3.434180
Critical Value (5%)             -2.863232
Critical Value (10%)            -2.567671
```

Figure 9: Test for stationarity for the time series data.

Additionally, the time series plot was overlaid with its rolling mean and rolling standard deviation using a window size of 12 samples. From the results of the Dickey-Fuller test we can see that the p-value is not less than 0.05, hence the we cannot reject the null hypothesis that the data is not stationary.

Next, I generated the first difference time series and repeated the stationarity test. Results are shown in Figure 10.




```
Results of Dickey-Fuller Test:
p-value = 0.0000. The series is likely stationary.
Test Statistic                -1.232606e+01
p-value                        6.585276e-23
#Lags Used                     2.200000e+01
Number of Observations Used    1.710000e+03
Critical Value (1%)           -3.434180e+00
Critical Value (5%)           -2.863232e+00
Critical Value (10%)          -2.567671e+00
dtype: float64
```

Figure 10: Test for stationarity for the time series data after first difference.

The results from the Dickey-Fuller test for first difference data shows that we can reject the null hypothesis that the data is not stationary. From this test we can use d=1 for the ARIMA model. Next, the parameters p and d will be estimated from ACF and PACF plots.

ACF and PACF and estimate p,q:

The auto-correlation and partial auto-correlation plots are generated for the training data after first difference for max lags of 40. Figure 11 shows the two plots. Based on the plots the ACF is small after lag 1 and PACF has small values after lag 6. We can use p and q values of 6 and 1 for ARIMA modeling.

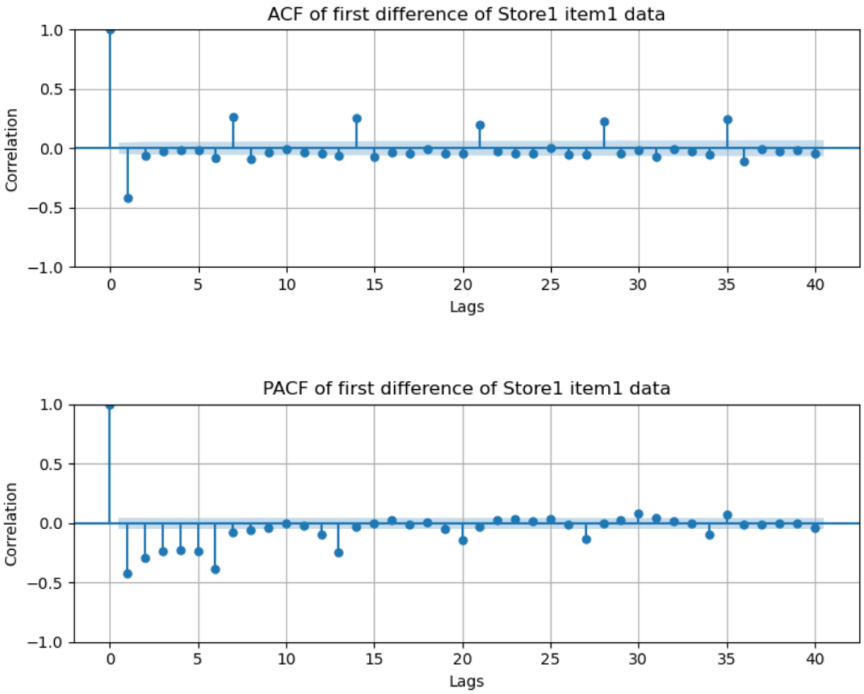


Figure 11: ACF and PACF of the first difference time series data.

Modeling Results:

ARIMA (6,1,0):

The training data is modeled using an ARIMA model of order 6,1,0. Below is a summary of model parameters, coefficients, AIC, BIC etc.

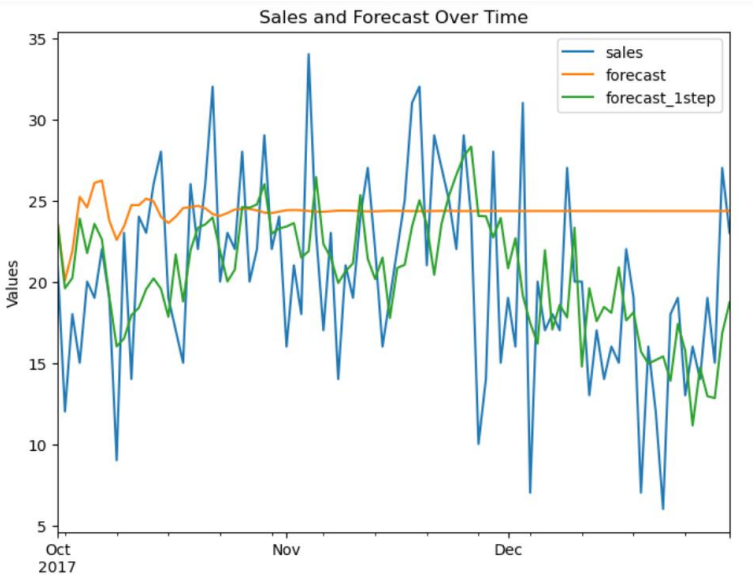
SARIMAX Results

Dep. Variable:	sales		No. Observations:		1734	
Model:	ARIMA(6, 1, 0)		Log Likelihood		-5313.067	
Date:	Sat, 13 Jan 2024		AIC		10640.134	
Time:	20:23:07		BIC		10678.337	
Sample:	0		HQIC		10654.263	
	- 1734					
Covariance Type:	opg					

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.8146	0.022	-37.871	0.000	-0.857	-0.772
ar.L2	-0.7557	0.025	-30.106	0.000	-0.805	-0.707
ar.L3	-0.6938	0.027	-26.154	0.000	-0.746	-0.642
ar.L4	-0.6210	0.027	-22.585	0.000	-0.675	-0.567
ar.L5	-0.5215	0.025	-20.511	0.000	-0.571	-0.472
ar.L6	-0.3887	0.021	-18.391	0.000	-0.430	-0.347
sigma2	26.9123	0.827	32.525	0.000	25.291	28.534

Ljung-Box (L1) (Q):	1.49	Jarque-Bera (JB):	21.72
Prob(Q):	0.22	Prob(JB):	0.00
Heteroskedasticity (H):	1.40	Skew:	0.16
Prob(H) (two-sided):	0.00	Kurtosis:	3.45

Figure 12: Summary of ARIMA model.



	Predict for 3 months	Predict 1 step
MAE:	5.79	4.37
MAPE:	39.63%	26.24%
SMAPE:	28.57%	22.88%

Figure 13: Forecast results using ARIMA (6,1,0) model.

The model was used to forecast the sales for last three months. The forecasting was done in two ways. First, all the future values were forecasted in one go. In the second method only for 1 day in future the forecast was predicted and true value from test data was appended to the training and iterated until the last sample. An error function was defined that calculates the error in prediction. The function returns three types of error: MAE (mean absolute error), MAPE (mean absolute percentage error) and SMAPE (symmetric mean absolute percentage error).

Based on the two modeling results shown in Figure 13 above here are the observations:

- From the plot we can see that ARIMA [6,1,0] model did not quite accurately model the predictions for store1 item1.
- For the first week the prediction seemed to be close to true values but after one month the prediction was constant.
- The prediction curve tapered after a few samples.
    - The MAPE error is ~40% and SMAPE is ~29% which seems a little high.
- When the modeling was used to predict only one sample at a time then the results improved.
    - Overall seasonal trend can be seen on the predictions which is encouraging.

- The MAPE error is ~26% and SMAPE error is ~23% which is a big improvement from the previous model.

Ideally, we would like to predict for all the future values (3 months) using the training data only. Considering the results, it is possible that there is some seasonal component that ARIMA is not able to model and/or the order of the ARIMA model was not accurate. We will explore the second possibility in the next section. For different p,d,q values different ARIMA models will be generated and AIC-BIC will be used to select the best model. Errors from prediction will be explored for all models.

ARIMA model iterating over various p,d,q values:

For all combinations of values of p from 1 to 10, d from 1 to 3 and q from 1 to 10, an ARIMA model of order p,d,q is modeled and fit to training data. The model is used to forecast the values for the test data. Error is calculated for each iteration. The data is documented in a data-frame and the models with lowest AIC, BIC, MAE, MAPE and SMAPE are filtered.

Below is a list of models with lowest overall values.

```
Order with minimum AIC: (4, 2, 2)
Order with minimum BIC: (4, 2, 2)
Order with minimum MAE: (3, 3, 9)
Order with minimum MAPE: (8, 3, 9)
Order with minimum SMAPE: (3, 3, 9)
```

As we know that first difference of 1 was enough for the stationarity test. So, for d=1 the models with lowest AIC, BIC and error are shown below:

```
Order with minimum AIC (when d=1): (7, 1, 9)
Order with minimum BIC (when d=1): (7, 1, 9)
Order with minimum MAE (when d=1): (9, 1, 10)
Order with minimum MAPE (when d=1): (9, 1, 10)
Order with minimum SMAPE (when d=1): (9, 1, 10)
```

Considering the information from these tests, ARIMA models with order 3,3,9 and 9,1,10 will be tested to see if the forecasting results are better than 6,1,0. Figure 14 and Figure 15 show the results from forecasting using the two models. From the figures it seems like even the error is small the prediction is not great. Likely the model is overfitting and our parameters estimated earlier 6,1,0 are probably better.

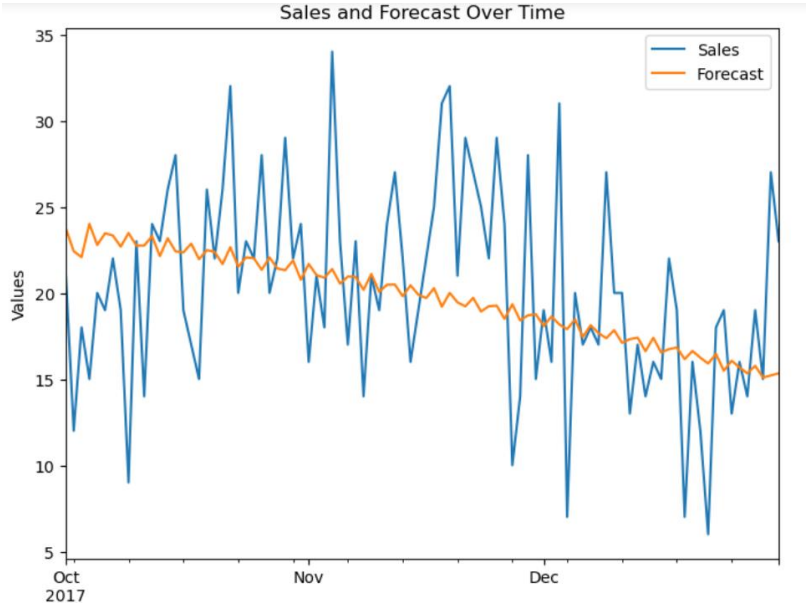


Figure 14: Forecast results using ARIMA (3,3,9) model.

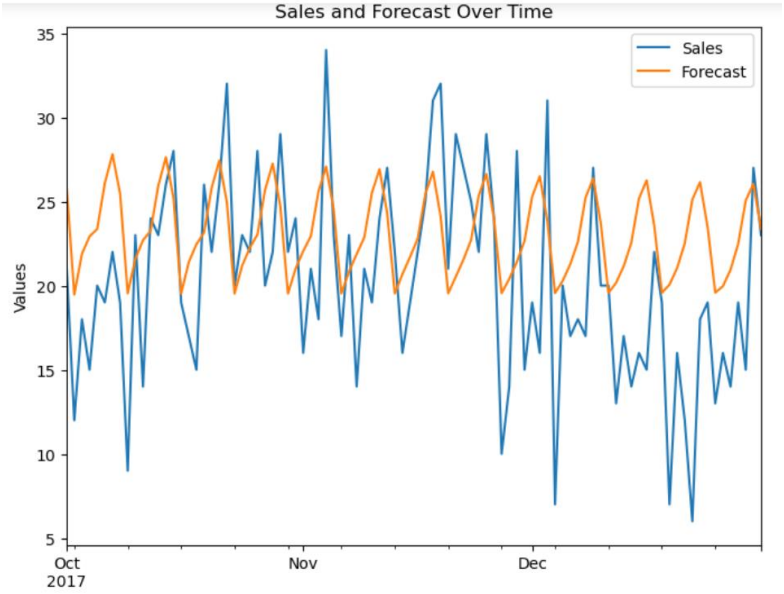
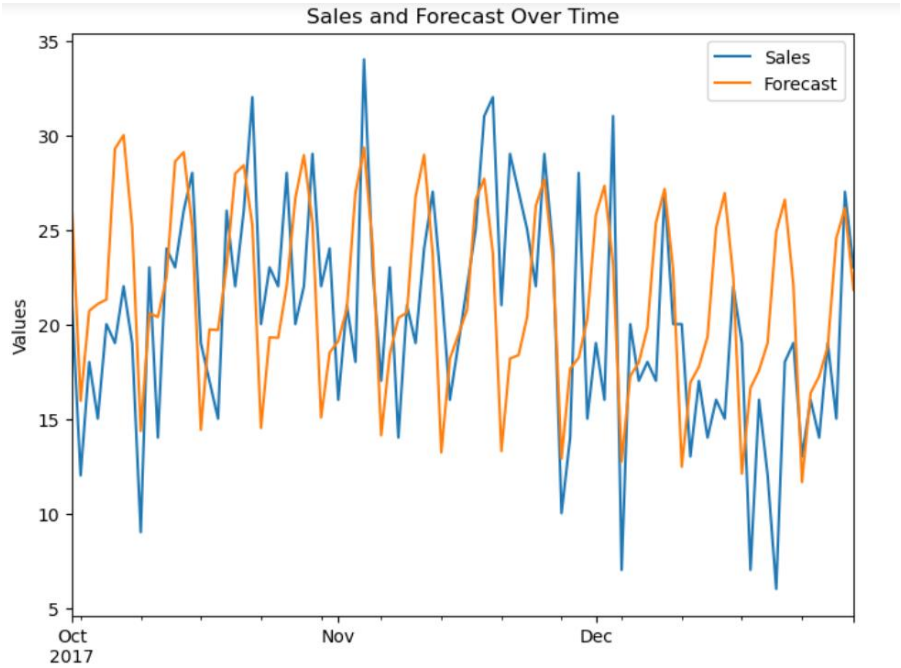


Figure 15: Forecast results using ARIMA (9,1,10) model.

Since changing the model order did not improve the prediction any further, next seasonality will be tested using SARIMA model.

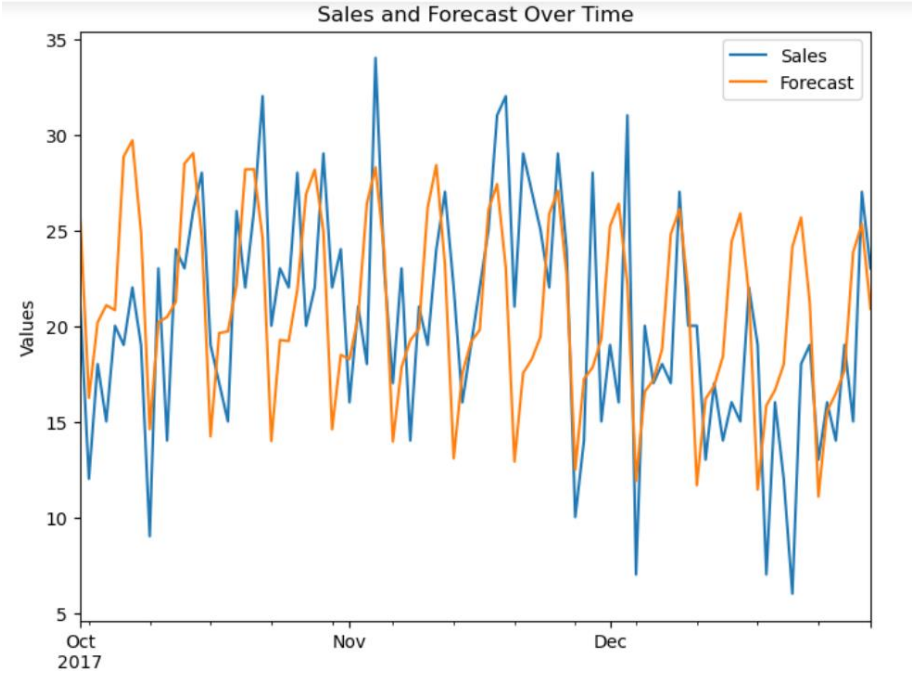
SARIMA (6,1,0) with a seasonal component of 7 days and exogenous variables:

From the time series plots, PACF etc. we know there is a weekly seasonal component. Here SARIMA was tried with same p,d,q (6,1,0) and with a seasonal value of 7. The forecasting results are shown in Figure 16. Additionally, the other time related features extracted earlier month, year, day of the year, day of the week were used as the exogenous component to the model. The model was used for forecasting and the results are shown in Figure 17 as well. The errors from forecasting for both the models are shown as well.



MAE: 4.8
MAPE: 29.02%
SMAPE: 24.49%

Figure 16: Forecast results using SARIMA (6,1,0,7) model.




```
MAE:  4.73
MAPE: 28.34%
SMAPE: 24.44%
```

Figure 17: Forecast results using SARIMA (6,1,0,7) model with exogenous features.

Overall, SARIMA model forecasting results were better than ARIMA. The forecasting improved a little by adding the exogenous variables. The MAPE error is lowered from 40% to 28%. SMAPE error is also less from 29% to 24%. For this data SARIMA with exogenous variables gave the best results. In order to predict for another store item combination we can repeat the steps of predicting model order and retrain and forecast.