IEOR4574 Project 1 – Forecasting Online Retail II Data Set
Process Documentation

Yunpeng Wang
yw3954

Contents

Introduction

- This project is essential to IEOR4574: Forecasting: A Real-World Application. In this project, we leverage time series analysis and models which we learned during classes and implement them to extract insights from real-world data. More importantly, it is a learning experience.
- The data we are interested in is an actual online retail transaction data set of two years. We focus on distinct product sales data in the United Kingdom.
- Apply explanatory data analytics to develop a robust approach to develop machine learning models to study stationary and non-stationary components among different products' sale data
- Standardize a robust methodology (including analytical approach and data requirements) for the development of forecasting that will be leveraged in the coming months

The purpose of this document is to provide a detailed technical overview of:

1. Design specifications and parameters
2. Data inclusions and exclusions
3. Predictive variable creation process
4. Target variable definition
5. Reproducible model-building process
6. Metric evaluation process

Data Extraction

The section explains the source and storage of data.

The whole online retail transaction data is manually pulled from UCI Machine Learning Repository. The primary source is Dr. Daqing Chen, who is MSc in Data Science at London South Bank University.

The format of the data is Microsoft Excel worksheets.

To process data efficiently, data is uploaded and stored in Google Drive. Data are adjusted and integrated using Python scripts

Data Overview

The purpose of this section is to provide overview of data we assigned.

This Online Retail II data set contains all the transactions occurring for a UK-based and registered, non-store online retail between 01/12/2009 and 09/12/2011. The company mainly sells unique all-occasion giftware. Many customers of the company are wholesalers.

All transactions are recorded chronologically, split into two sections based on the time transaction came about and noted on two worksheets, whose names are ‘Year 2009-2010’ and ‘Year 2010-2011’.

The total record count for each worksheet is 525461 and 541911 respectively.

There are 8 attributes that illustrate the details of trades. They are the following:

InvoiceNo: Invoice number. Nominal. A 6-digit integral number is uniquely assigned to each transaction. If this code starts with the letter 'c', it indicates a cancellation.

StockCode: Product (item) code. Nominal. A 5-digit integral number is uniquely assigned to each distinct product.

Description: Product (item) name. Nominal.

Quantity: The quantities of each product (item) per transaction. Numeric.

InvoiceDate: Invoice date and time down to minutes. Numeric. The day and time when a transaction was generated.

UnitPrice: Unit price. Numeric. Product price per unit in sterling (Â£).

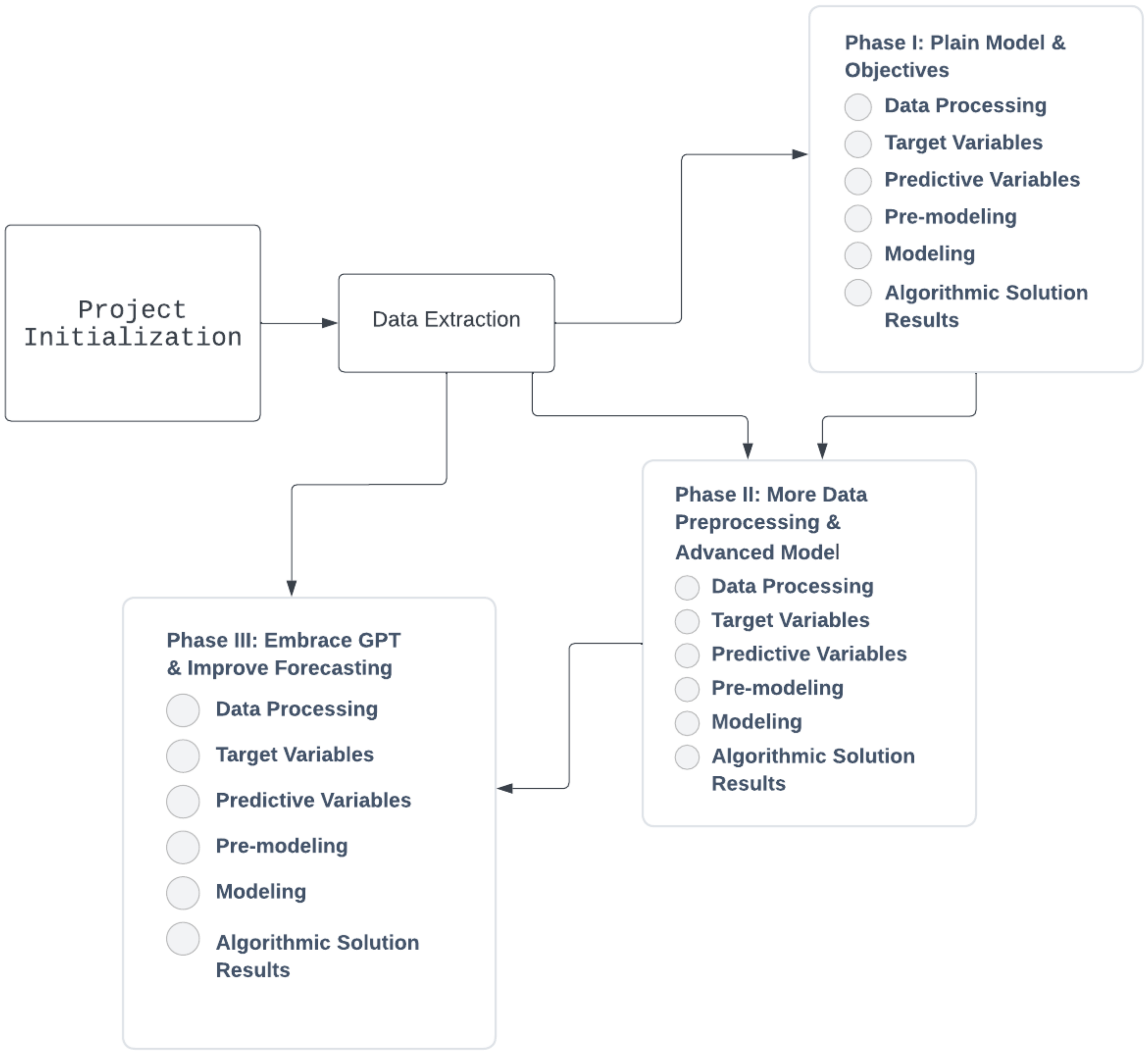
CustomerID: Customer number. Nominal. A 5-digit integral number is uniquely assigned to each customer.

Country: Country name. Nominal. The name of the country where a customer resides.

Demo:

Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
489434	85048	15CM CHRISTMAS GLASS BALL 20 LIGHTS	12	12/1/09 7:45	6.95	13085	United Kingdom
489434	79323P	PINK CHERRY LIGHTS	12	12/1/09 7:45	6.75	13085	United Kingdom
489434	79323W	WHITE CHERRY LIGHTS	12	12/1/09 7:45	6.75	13085	United Kingdom
489434	22041	RECORD FRAME 7" SINGLE SIZE	48	12/1/09 7:45	2.1	13085	United Kingdom

Process Breakdown



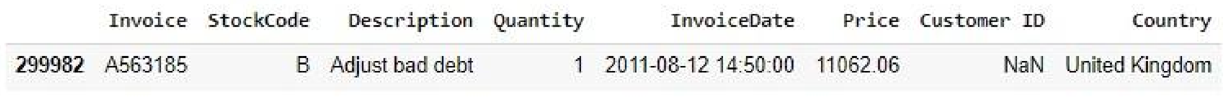
Phase I: Plain Model & Objectives Data Processing

This section illustrates the nascent stage of project workflow, where data are preprocessed and models are tried out for the first time. Nothing too profound or complicated. But the goal of this phase is to find straightforward methods for forecasting time series given certain constraints.

Data Processing

Data diagnostics

Invoice is the first attribute scrutinized. According to attribute information, an **Invoice** is a 6-digit integral number uniquely assigned to each transaction. If this code starts with the letter '**C**', it indicates a cancellation. However, it does not specify what codes with the initial letter '**A**' mean. After inspection, '**A**' indicates adjustments to bad debts.



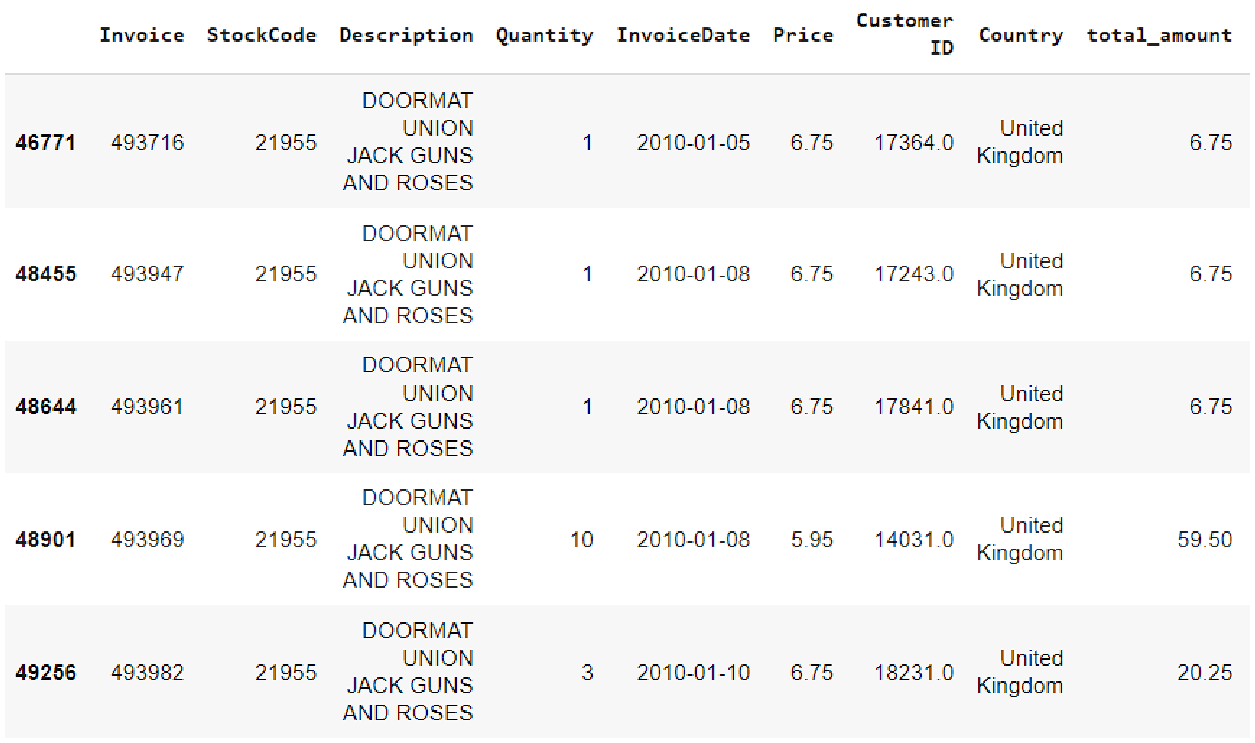
	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
299982	A563185	B	Adjust bad debt	1	2011-08-12 14:50:00	11062.06	NaN	United Kingdom

Then, since both **Description** and **StockCode** are signatures of products and each product has its own unique **Description** and **StockCode**, it is necessary to check if the products' **Description** and **StockCode** are consistent. The consistency of these two attributes is verified by calling unique().

Last but not least, in order to cut to the point and dive into models' trial and error straight away, some key steps in data cleaning, such as replacing missing values, handling outliers, and adding auxiliary data, are being skipped. **Aggregating data into time periods** is implemented directly.

Data Cleaning

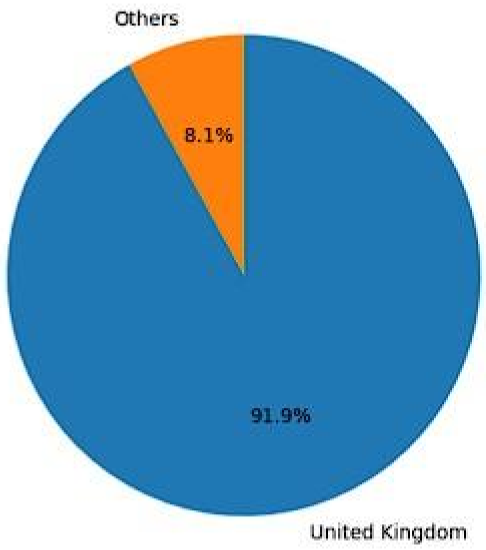
Firstly, the feature called **total_amount**, which is calculated by multiplication of **Price** and **Quantity**, is added to the data, because **Total_amount** is a key market performance indicator.



	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country	total_amount
46771	493716	21955	DOORMAT UNION JACK GUNS AND ROSES	1	2010-01-05	6.75	17364.0	United Kingdom	6.75
48455	493947	21955	DOORMAT UNION JACK GUNS AND ROSES	1	2010-01-08	6.75	17243.0	United Kingdom	6.75
48644	493961	21955	DOORMAT UNION JACK GUNS AND ROSES	1	2010-01-08	6.75	17841.0	United Kingdom	6.75
48901	493969	21955	DOORMAT UNION JACK GUNS AND ROSES	10	2010-01-08	5.95	14031.0	United Kingdom	59.50
49256	493982	21955	DOORMAT UNION JACK GUNS AND ROSES	3	2010-01-10	6.75	18231.0	United Kingdom	20.25

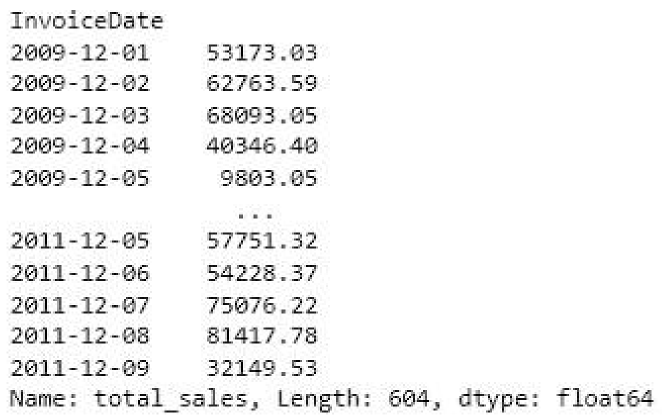
Afterward, **InvoiceDate** is reformatted into datetime down to specific **days**.

Moreover, data is filtered by **Country**. Those records whose Country is not the **United Kingdom** are dropped off because the majority of records happen within the United Kingdom market.



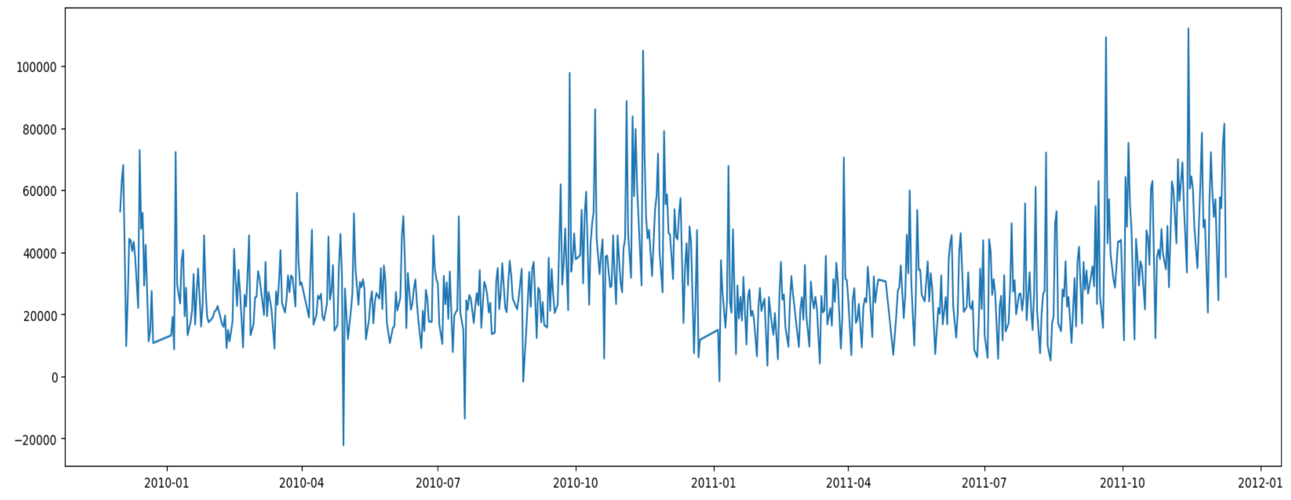
Two worksheet data are processed in accordance with the logic above and merged into one worksheet. Then duplicate transactions are cut off and **total_amount** is added up by **InvoiceDate**.

The final version is represented in the following table.



```
InvoiceDate
2009-12-01    53173.03
2009-12-02    62763.59
2009-12-03    68093.05
2009-12-04    40346.40
2009-12-05     9803.05
                ...
2011-12-05    57751.32
2011-12-06    54228.37
2011-12-07    75076.22
2011-12-08    81417.78
2011-12-09    32149.53
Name: total_sales, Length: 604, dtype: float64
```

Data Visualization



Target Variable

A target variable is the variable to predict as a result in algorithmic solutions.

Total_amount in sterling (Â£) per InvoiceDate is the **target variable**.

Predictive Variable

A predictive variable is a variable used in algorithmic solutions to predict the target variable. During our analysis, we categorized predictive variables into two categories:

1. Direct variable – These variables were directly from the dataset that was provided by direct customers
2. Derived variable – These variables were created by manipulating the direct variables

Direct Variable List:

1. InvoiceDate: Features of time are critical in the modeling because they capture seasonality

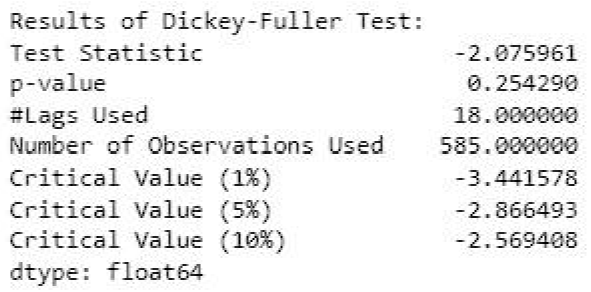
Derived Variable List:

1. const: constant dummies
2. lags: serial dependence
3. trend, trend_squared, trend_cubed: time trends
4. s(2,7), s(3, 7) …: weekly seasonality
5. sin(23,freq=A-DEC), cos(23,freq=A-DEC) …: annual seasonality
6. Holiday: US national holidays. The holiday has an effect on people's purchasing patterns.

Pre-modeling

Data stationarity check:

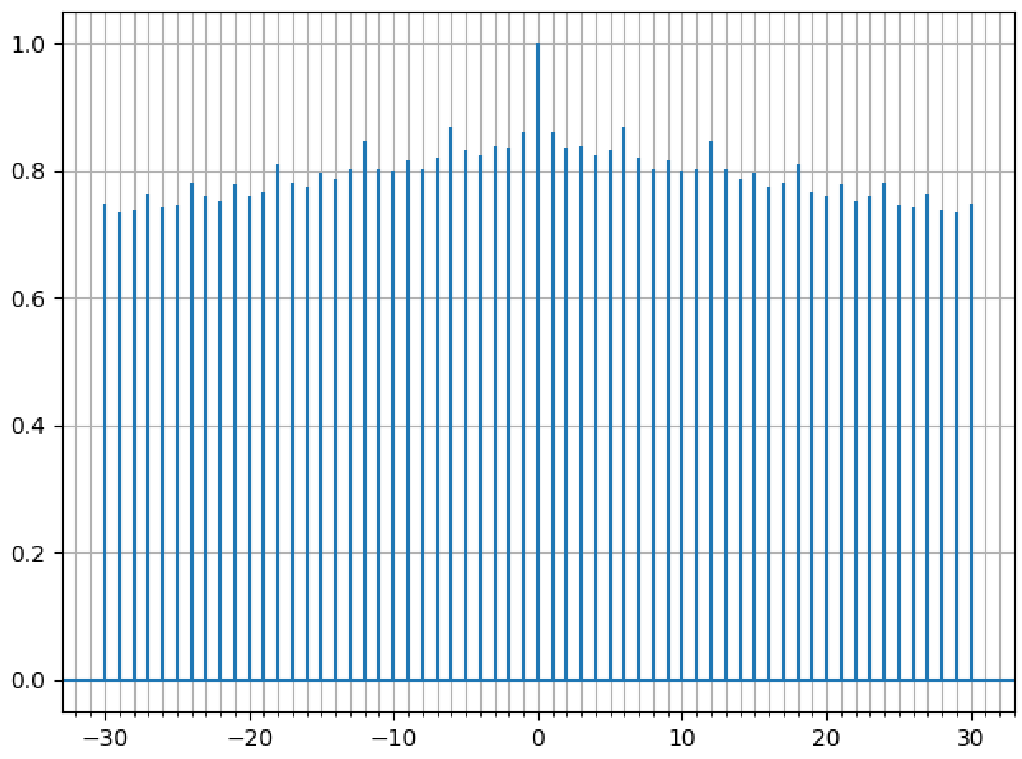
Augmented Dickey-Fuller is a statistical significance test, especially widely used for time series stationarity checks. As the Dickey-Fuller test resulted in a p-value a lot higher than 0.05, it means the null hypothesis of non-stationarity cannot be rejected at a 0.05 significance level.



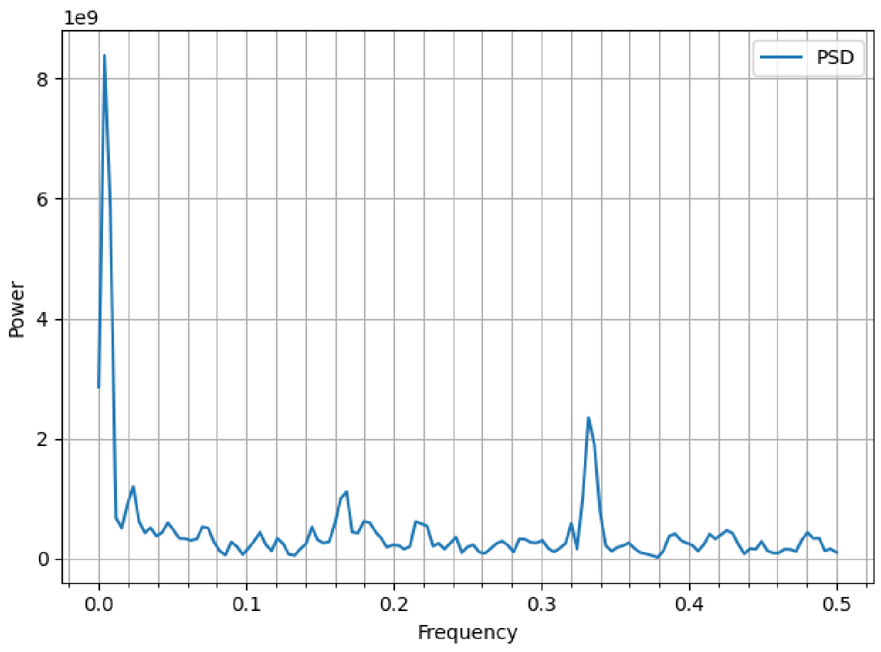
```
Results of Dickey-Fuller Test:
Test Statistic                 -2.075961
p-value                         0.254290
#Lags Used                     18.000000
Number of Observations Used   585.000000
Critical Value (1%)            -3.441578
Critical Value (5%)            -2.866493
Critical Value (10%)           -2.569408
dtype: float64
```

Data seasonality and trend analysis:

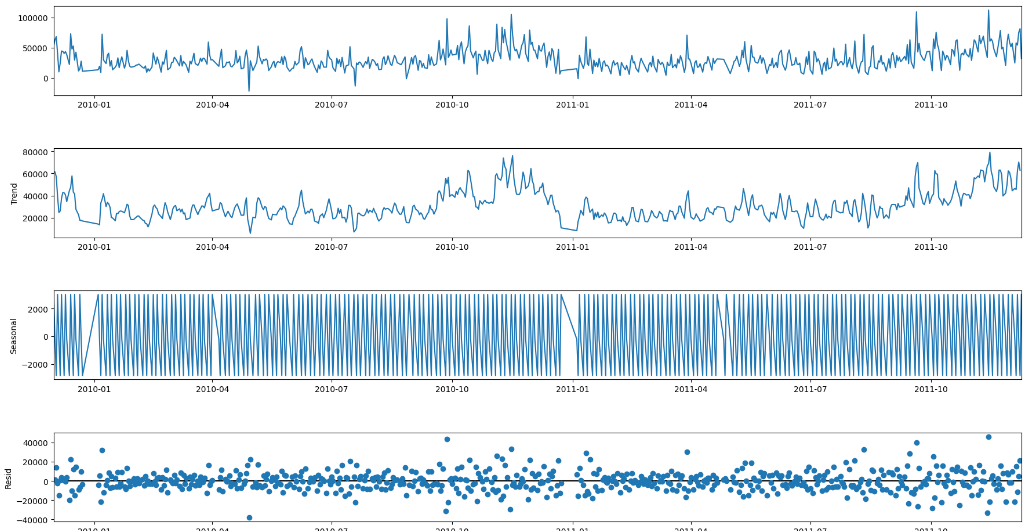
Since non-stationary factors disturb sales data, the following data seasonality and trend analysis proceeds. Following are the auto-correlation graphs with a maximum lag of 30 and 600 days, As lag increases, the auto-correlation gradually decreases and they perform a wave shape with a frequency of 7 days. That makes sense, since the sales may perform periodicity in the time scale of a single week.



The graph below displays the Power Spectral Density (PSD), which reveals a peak near the frequency of 0.002, which indicates that there exists a seasonality in the sales of a year. Moreover, the second peak is located near the frequency of 0.34.

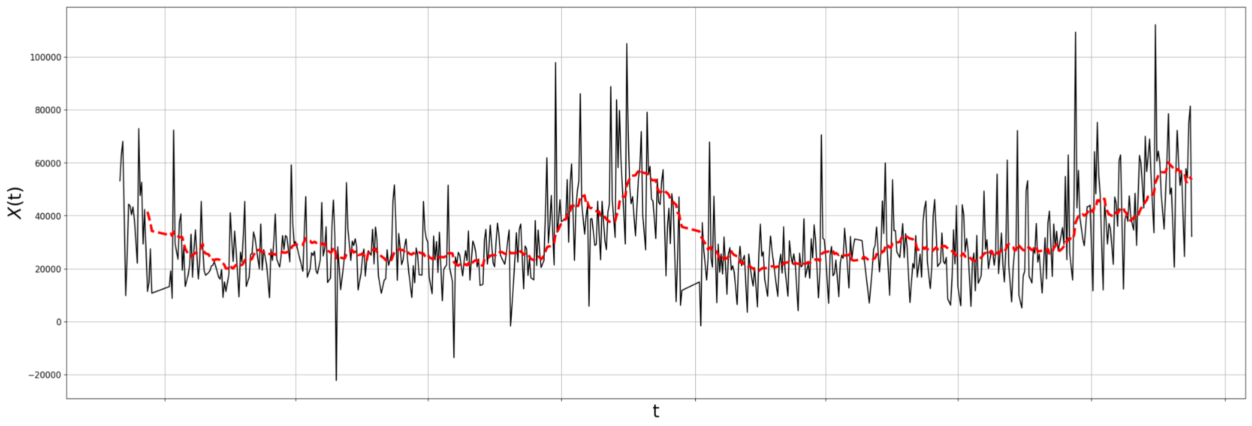


Furthermore, since the second peak in the previous diagram has a frequency of 0.34, sales data might have periodicity in the time scale of 3 days. As a result, by implementing **seasonal_decompose** and setting the parameter of period equal to 3, the following group of graphs is manufactured.



As shown, there are lots of ups and downs in the trend subfigure and residuals of data after teasing seasonality and trend are approximately normally distributed with a mean equal to 0. Thence, seasonal trends, and stationary residuals are deduced.

To validate the theory that there exists a seasonal trend in sales data, the moving average technique is leveraged. The sliding window is defined as 18 days. The result aligns with the theory.



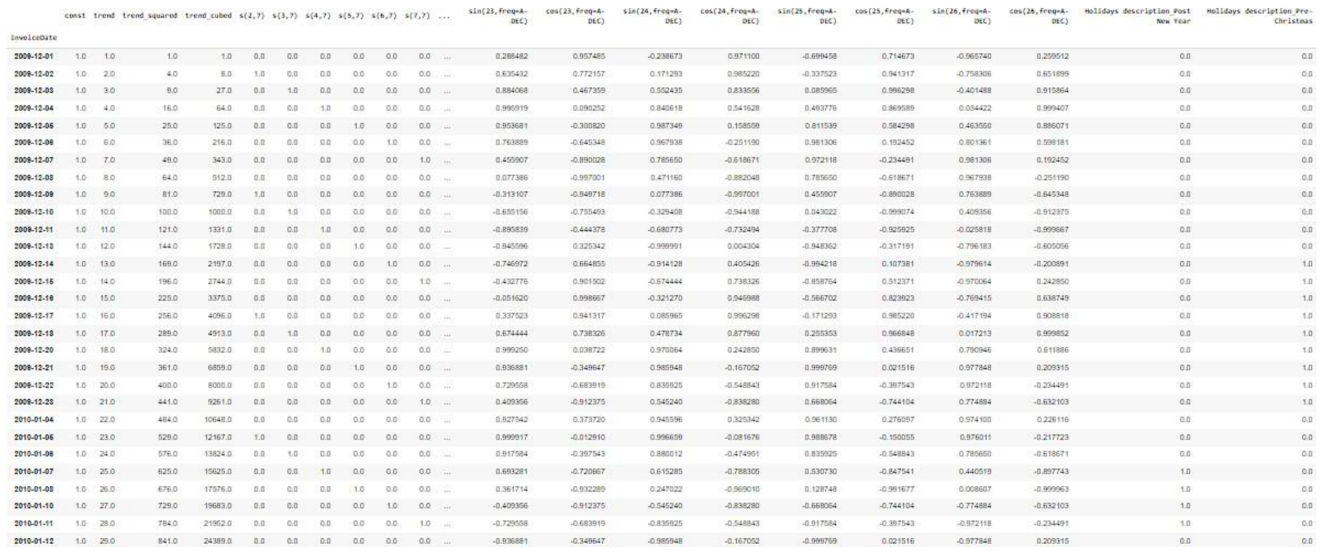
Derived variable creation:

Weekly, annual seasonal feature and trend features:

Source: **DeterministicProcess, CalendarFourier** from **statsmodels.tsa.deterministic**

Holiday features:

Source: Manually noting down **two weeks**' dates **prior to Christmas** and another two weeks' dates **post New Year** and **One-Hot encoding**



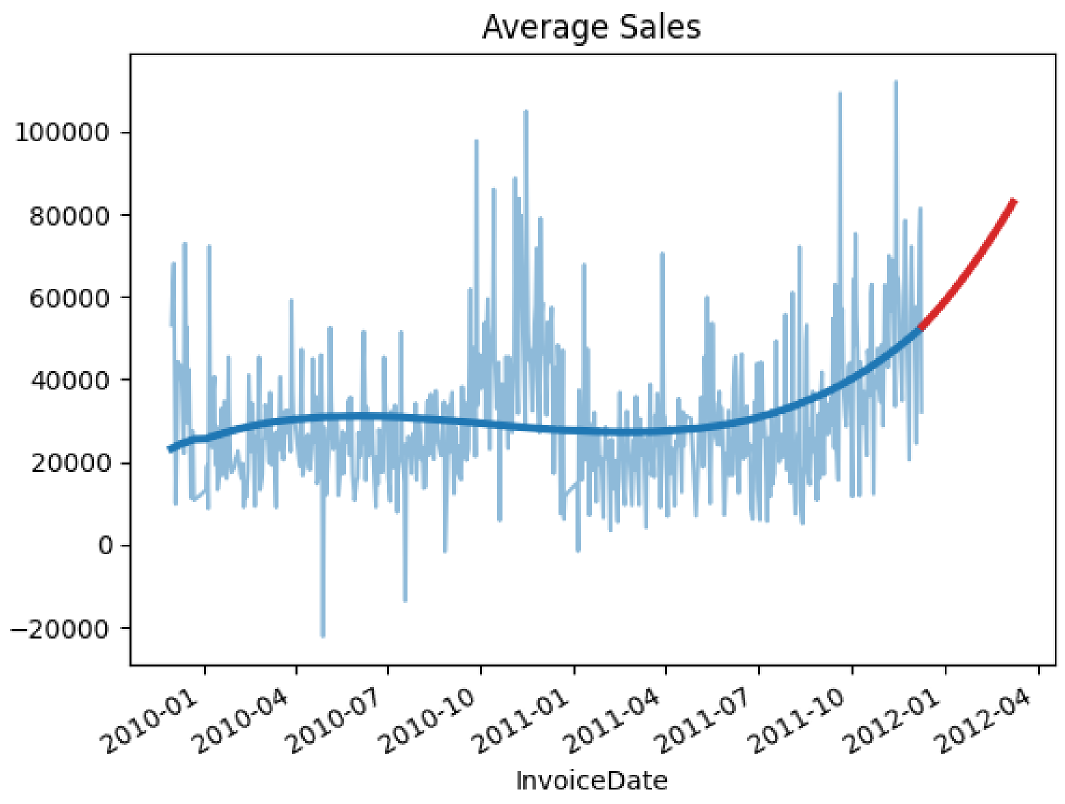
	const	trend	trend_squared	trend_cubed	s(2,7)	s(3,7)	s(4,7)	s(5,7)	s(6,7)	s(7,7)	...	sin(23,freq=A-DEC)	cos(23,freq=A-DEC)	sin(24,freq=A-DEC)	cos(24,freq=A-DEC)	sin(25,freq=A-DEC)	cos(25,freq=A-DEC)	sin(26,freq=A-DEC)	cos(26,freq=A-DEC)	Holidays description_Post New Year	Holidays description_Pre-Christmas
InvoiceDate																					
2009-12-01	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.288482	0.957485	-0.238673	0.971100	-0.699458	0.714673	-0.965740	0.259512	0.0	0.0
2009-12-02	1.0	2.0	4.0	8.0	1.0	0.0	0.0	0.0	0.0	0.0	...	0.635432	0.772157	0.171293	0.985220	-0.337523	0.941317	-0.758306	0.651899	0.0	0.0
2009-12-03	1.0	3.0	9.0	27.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0.884068	0.467359	0.552435	0.833556	0.085965	0.996298	-0.401488	0.915864	0.0	0.0
2009-12-04	1.0	4.0	16.0	64.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.995919	0.090252	0.840618	0.541628	0.493776	0.869589	0.034422	0.999407	0.0	0.0
2009-12-05	1.0	5.0	25.0	125.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.953681	-0.300820	0.987349	0.158559	0.811539	0.584298	0.463550	0.886071	0.0	0.0
2009-12-06	1.0	6.0	36.0	216.0	0.0	0.0	0.0	0.0	1.0	0.0	...	0.763889	-0.645348	0.967938	-0.251190	0.981306	0.192452	0.801361	0.598181	0.0	0.0
2009-12-07	1.0	7.0	49.0	343.0	0.0	0.0	0.0	0.0	0.0	1.0	...	0.455907	-0.890028	0.785650	-0.618671	0.972118	-0.234491	0.981306	0.192452	0.0	0.0
2009-12-08	1.0	8.0	64.0	512.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.077386	-0.997001	0.471160	-0.882048	0.785650	-0.618671	0.967938	-0.251190	0.0	0.0
2009-12-09	1.0	9.0	81.0	729.0	1.0	0.0	0.0	0.0	0.0	0.0	...	-0.313107	-0.949718	0.077386	-0.997001	0.455907	-0.890028	0.763889	-0.645348	0.0	0.0
2009-12-10	1.0	10.0	100.0	1000.0	0.0	1.0	0.0	0.0	0.0	0.0	...	-0.655156	-0.755493	-0.329408	-0.944188	0.043022	-0.999074	0.409356	-0.912375	0.0	0.0
2009-12-11	1.0	11.0	121.0	1331.0	0.0	0.0	1.0	0.0	0.0	0.0	...	-0.895839	-0.444378	-0.680773	-0.732494	-0.377708	-0.925925	-0.025818	-0.999667	0.0	0.0
2009-12-13	1.0	12.0	144.0	1728.0	0.0	0.0	0.0	1.0	0.0	0.0	...	-0.945596	0.325342	-0.999991	0.004304	-0.948362	-0.317191	-0.796183	-0.605056	0.0	0.0
2009-12-14	1.0	13.0	169.0	2197.0	0.0	0.0	0.0	0.0	1.0	0.0	...	-0.746972	0.664855	-0.914128	0.405426	-0.994218	0.107381	-0.979614	-0.200891	0.0	1.0
2009-12-15	1.0	14.0	196.0	2744.0	0.0	0.0	0.0	0.0	0.0	1.0	...	-0.432776	0.901502	-0.674444	0.738326	-0.858764	0.512371	-0.970064	0.242850	0.0	1.0
2009-12-16	1.0	15.0	225.0	3375.0	0.0	0.0	0.0	0.0	0.0	0.0	...	-0.051620	0.998667	-0.321270	0.946988	-0.566702	0.823923	-0.769415	0.638749	0.0	1.0
2009-12-17	1.0	16.0	256.0	4096.0	1.0	0.0	0.0	0.0	0.0	0.0	...	0.337523	0.941317	0.085965	0.996298	-0.171293	0.985220	-0.417194	0.908818	0.0	1.0
2009-12-18	1.0	17.0	289.0	4913.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0.674444	0.738326	0.478734	0.877960	0.255353	0.966848	0.017213	0.999852	0.0	1.0
2009-12-20	1.0	18.0	324.0	5832.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.999250	0.038722	0.970064	0.242850	0.899631	0.436651	0.790946	0.611886	0.0	1.0
2009-12-21	1.0	19.0	361.0	6859.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.936881	-0.349647	0.985948	-0.167052	0.999769	0.021516	0.977848	0.209315	0.0	1.0
2009-12-22	1.0	20.0	400.0	8000.0	0.0	0.0	0.0	0.0	1.0	0.0	...	0.729558	-0.683919	0.835925	-0.548843	0.917584	-0.397543	0.972118	-0.234491	0.0	1.0
2009-12-23	1.0	21.0	441.0	9261.0	0.0	0.0	0.0	0.0	0.0	1.0	...	0.409356	-0.912375	0.545240	-0.838280	0.668064	-0.744104	0.774884	-0.632103	0.0	1.0
2010-01-04	1.0	22.0	484.0	10648.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.927542	0.373720	0.945596	0.325342	0.961130	0.276097	0.974100	0.226116	0.0	0.0
2010-01-05	1.0	23.0	529.0	12167.0	1.0	0.0	0.0	0.0	0.0	0.0	...	0.999917	-0.012910	0.996659	-0.081676	0.988678	-0.150055	0.976011	-0.217723	0.0	0.0
2010-01-06	1.0	24.0	576.0	13824.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0.917584	-0.397543	0.880012	-0.474951	0.835925	-0.548843	0.785650	-0.618671	0.0	0.0
2010-01-07	1.0	25.0	625.0	15625.0	0.0	0.0	1.0	0.0	0.0	0.0	...	0.693281	-0.720667	0.615285	-0.788305	0.530730	-0.847541	0.440519	-0.897743	1.0	0.0
2010-01-08	1.0	26.0	676.0	17576.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.361714	-0.932289	0.247022	-0.969010	0.128748	-0.991677	0.008607	-0.999963	1.0	0.0
2010-01-10	1.0	27.0	729.0	19683.0	0.0	0.0	0.0	0.0	1.0	0.0	...	-0.409356	-0.912375	-0.545240	-0.838280	-0.668064	-0.744104	-0.774884	-0.632103	1.0	0.0
2010-01-11	1.0	28.0	784.0	21952.0	0.0	0.0	0.0	0.0	0.0	1.0	...	-0.729558	-0.683919	-0.835925	-0.548843	-0.917584	-0.397543	-0.972118	-0.234491	1.0	0.0
2010-01-12	1.0	29.0	841.0	24389.0	0.0	0.0	0.0	0.0	0.0	0.0	...	-0.936881	-0.349647	-0.985948	-0.167052	-0.999769	0.021516	-0.977848	0.209315	0.0	0.0

Modeling

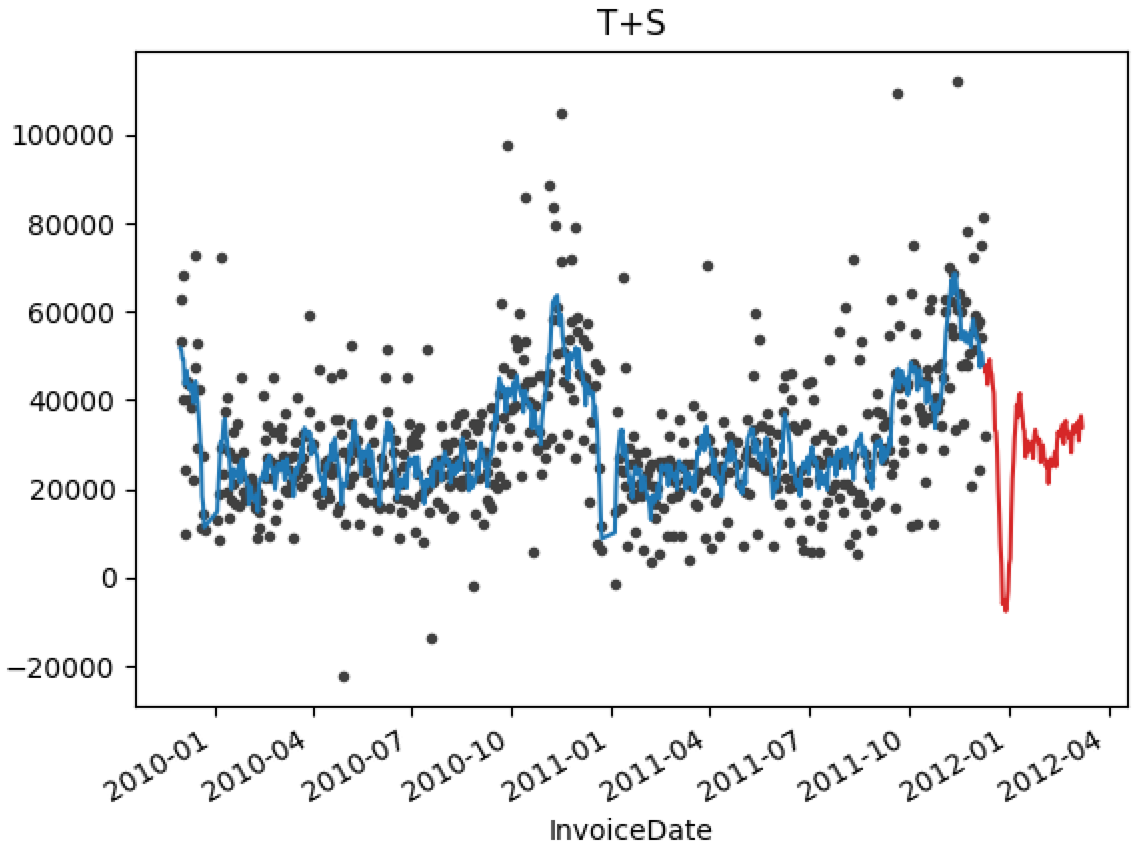
Linear Regression and **SARIMAX** are built to predict the UK market daily sale data during this phase.

Linear Regression:

The first Linear Regression model is set up to forecasting trends with high-order polynomials. Because neither underfitting nor overfitting is desirable, the order of polynomials is defined as 3. As the following graph demonstrates, the blue line captures significant seasonal trends while the red one represents the model's prediction on an out-of-sample time frame.

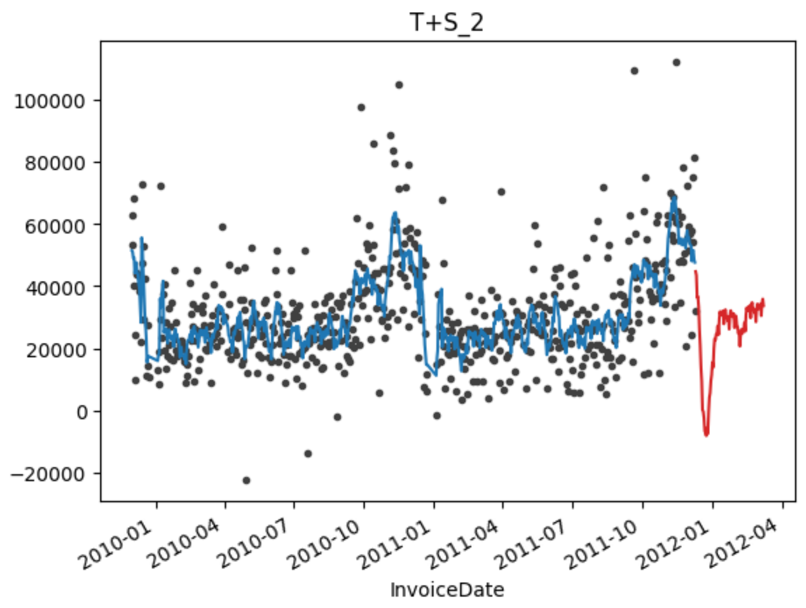


Then, to take account of weekly and annual seasonality, the second Linear Regression model is created and trained on weekly, annual seasonal features and trend features.



The figure above shows this model is capable of learning daily changes in sales data and predicting granularly.

Then with Holiday features coming in, the third Linear Regression model is implemented. This model yields the same result as the second model except for a few nuances.



SARIMAX:

The parameters of SARIMAX are hard to determine depending on explanatory data analysis. Accordingly, it is necessary to proceed by trial and error.

To begin with, all combinations of differencing orders as needed are created to obtain proper d&D indicating the integration order of the process.

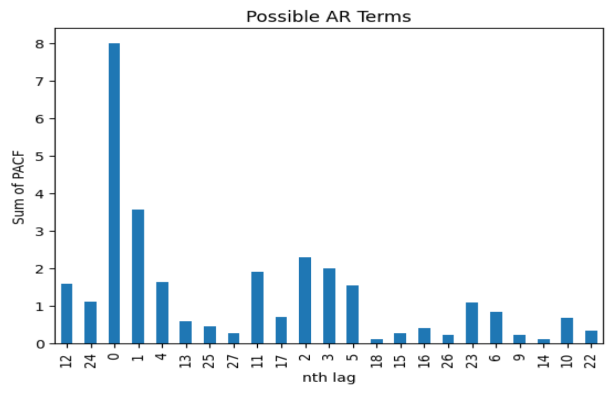
	d0_D0_m0	d1_D0_m0	d2_D0_m0	d0_D1_m12	d1_D1_m12	d2_D1_m12	d0_D2_m12	d1_D2_m12	d2_D2_m12
InvoiceDate									
2009-12-01	53173.03	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2009-12-02	62763.59	9590.56	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2009-12-03	68093.05	5329.46	-4261.10	NaN	NaN	NaN	NaN	NaN	NaN
2009-12-04	40346.40	-27746.65	-33076.11	NaN	NaN	NaN	NaN	NaN	NaN
2009-12-05	9803.05	-30543.35	-2796.70	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...
2011-12-05	57751.32	33185.54	65705.82	9448.82	19785.05	39175.54	31147.40	33473.24	56903.30
2011-12-06	54228.37	-3522.95	-36708.49	-8078.95	-17527.77	-37312.82	-13738.61	-44886.01	-78359.25
2011-12-07	75076.22	20847.85	24370.80	-3404.48	4674.47	22202.24	-19285.75	-5547.14	39338.87
2011-12-08	81417.78	6341.56	-14506.29	33337.50	36741.98	32067.51	54213.46	73499.21	79046.35
2011-12-09	32149.53	-49268.25	-55609.81	-18293.19	-51630.69	-88372.67	-13900.40	-68113.86	-141613.07

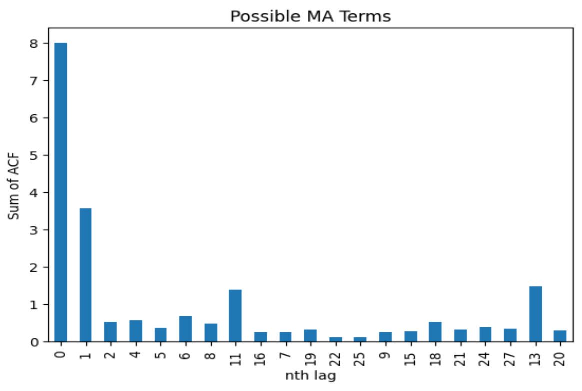
604 rows × 9 columns

Next, the Augmented Dickey-Fuller test is used to filter away results that are not stationary and retain results that are stationary.

	Test Statistic	p-value	#Lags Used	No. of Obs. Used	Critical Value (1%)	Critical Value (5%)	Critical Value (10%)
d0_D0_m0	-2.075961	2.542900e-01	18	585	-3.441578	-2.866493	-2.569408
d1_D0_m0	-7.706146	1.300056e-11	17	585	-3.441578	-2.866493	-2.569408
d2_D0_m0	-12.086310	2.173512e-22	19	582	-3.441636	-2.866519	-2.569422
d0_D1_m12	-5.380110	3.728728e-06	18	573	-3.441814	-2.866597	-2.569463
d1_D1_m12	-11.223171	1.990167e-20	16	574	-3.441794	-2.866588	-2.569459
d2_D1_m12	-11.483270	4.935113e-21	19	570	-3.441875	-2.866624	-2.569478
d0_D2_m12	-6.527891	1.003911e-08	18	561	-3.442060	-2.866706	-2.569521
d1_D2_m12	-12.363571	5.477534e-23	16	562	-3.442039	-2.866697	-2.569516
d2_D2_m12	-11.678219	1.766994e-21	19	558	-3.442124	-2.866734	-2.569536

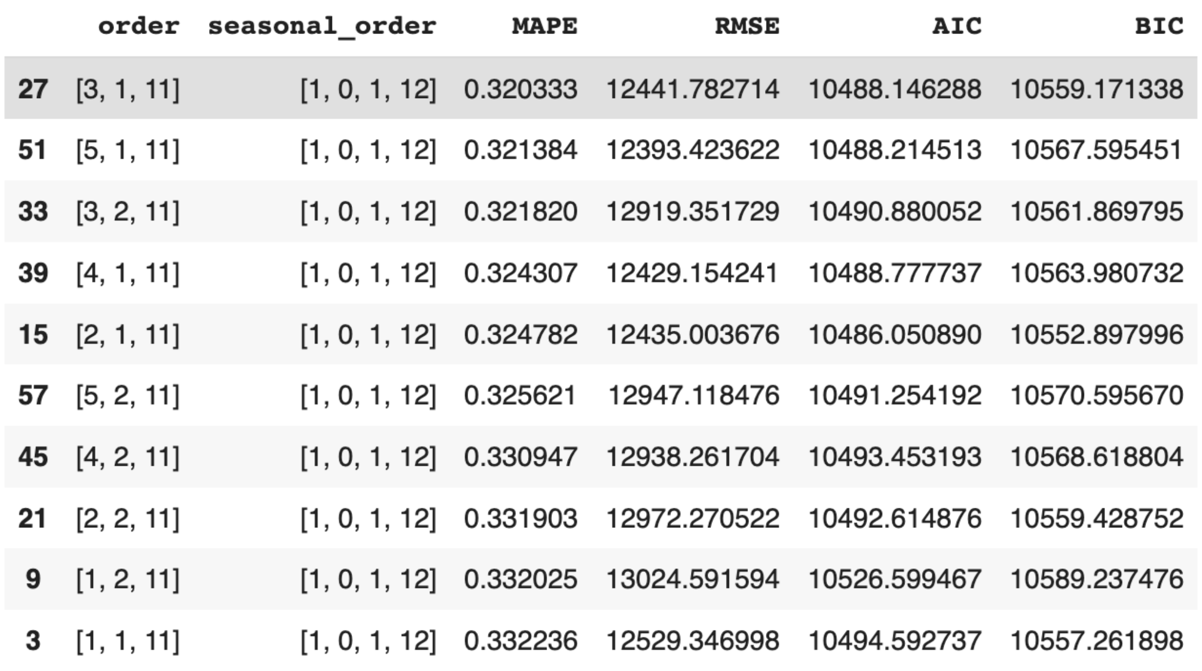
Then, the partial autocorrelation function plays an important role in finding all potential MA and AR terms of the SARIMAX model. All we need is to spot on significant spikes by n-lags in ACF plots and the corresponding horizontal coordinates of spikes are appropriate parameters.





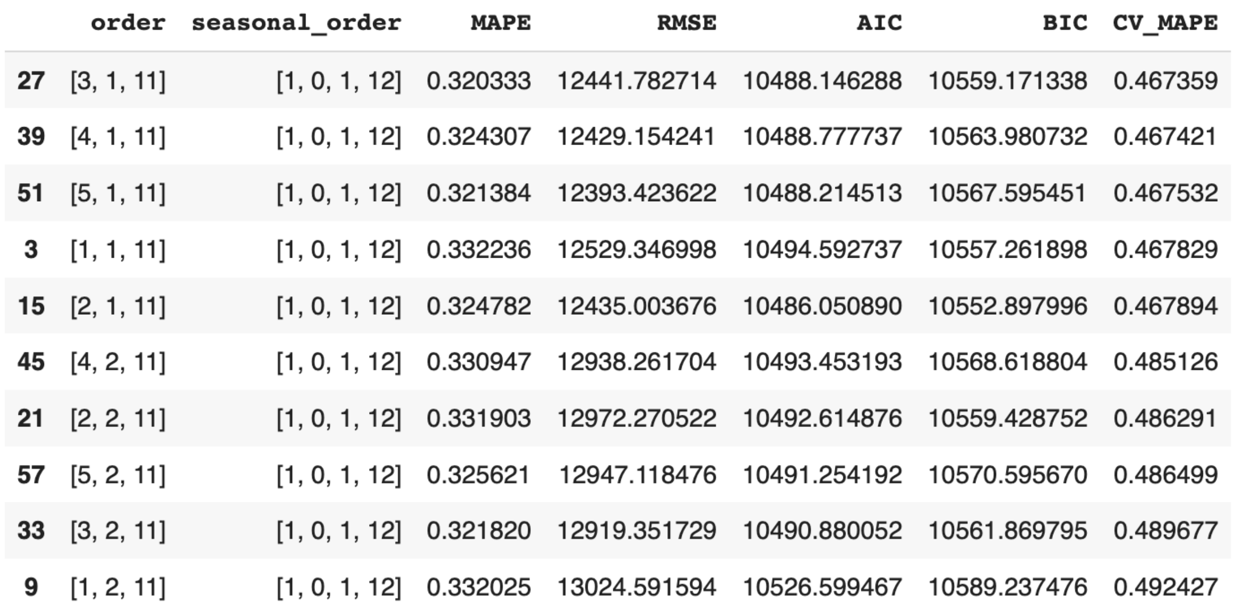
Therefore, p = [1, 2, 3, 4, 5], where P is the AR term and the list consists of all viable values of AR terms. q = [1, 11], where Q is the MA term and the list consists of all viable values of MA terms.

Finally, different SARIMAX models with distinct parameter combinations are founded and trained on the entire sale data.



	order	seasonal_order	MAPE	RMSE	AIC	BIC
27	[3, 1, 11]	[1, 0, 1, 12]	0.320333	12441.782714	10488.146288	10559.171338
51	[5, 1, 11]	[1, 0, 1, 12]	0.321384	12393.423622	10488.214513	10567.595451
33	[3, 2, 11]	[1, 0, 1, 12]	0.321820	12919.351729	10490.880052	10561.869795
39	[4, 1, 11]	[1, 0, 1, 12]	0.324307	12429.154241	10488.777737	10563.980732
15	[2, 1, 11]	[1, 0, 1, 12]	0.324782	12435.003676	10486.050890	10552.897996
57	[5, 2, 11]	[1, 0, 1, 12]	0.325621	12947.118476	10491.254192	10570.595670
45	[4, 2, 11]	[1, 0, 1, 12]	0.330947	12938.261704	10493.453193	10568.618804
21	[2, 2, 11]	[1, 0, 1, 12]	0.331903	12972.270522	10492.614876	10559.428752
9	[1, 2, 11]	[1, 0, 1, 12]	0.332025	13024.591594	10526.599467	10589.237476
3	[1, 1, 11]	[1, 0, 1, 12]	0.332236	12529.346998	10494.592737	10557.261898

10 least MAPE models are selected and cross-validated. Ultimately, the results are shown in the table below.



	order	seasonal_order	MAPE	RMSE	AIC	BIC	CV_MAPE
27	[3, 1, 11]	[1, 0, 1, 12]	0.320333	12441.782714	10488.146288	10559.171338	0.467359
39	[4, 1, 11]	[1, 0, 1, 12]	0.324307	12429.154241	10488.777737	10563.980732	0.467421
51	[5, 1, 11]	[1, 0, 1, 12]	0.321384	12393.423622	10488.214513	10567.595451	0.467532
3	[1, 1, 11]	[1, 0, 1, 12]	0.332236	12529.346998	10494.592737	10557.261898	0.467829
15	[2, 1, 11]	[1, 0, 1, 12]	0.324782	12435.003676	10486.050890	10552.897996	0.467894
45	[4, 2, 11]	[1, 0, 1, 12]	0.330947	12938.261704	10493.453193	10568.618804	0.485126
21	[2, 2, 11]	[1, 0, 1, 12]	0.331903	12972.270522	10492.614876	10559.428752	0.486291
57	[5, 2, 11]	[1, 0, 1, 12]	0.325621	12947.118476	10491.254192	10570.595670	0.486499
33	[3, 2, 11]	[1, 0, 1, 12]	0.321820	12919.351729	10490.880052	10561.869795	0.489677
9	[1, 2, 11]	[1, 0, 1, 12]	0.332025	13024.591594	10526.599467	10589.237476	0.492427

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

In conclusion, the lowest MAPE out of SARIMAX is 0.32, which is not quite favorable. For the purpose of improving the accuracy of predictions and reducing the Mean Absolute Percentage Error (MAPE) value, the clustering model is involved to group up products within similar categories, and Facebook Prophet/ Random Forest Regression model is considered in the next phase.

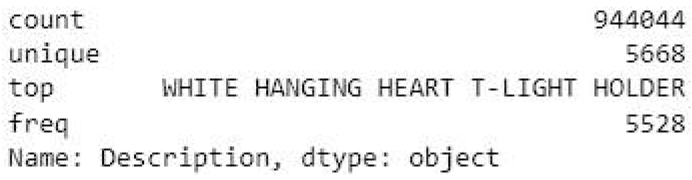
Phase II: More Data Preprocessing & Advanced Model

This section illustrates the intermediate stage of project workflow, where we already have experience as we go through phase I, and we are grinding to promote our forecasting accuracy.

Data Processing

Data Diagnostics

Customers have different needs for different commodities per day. Demands for goods vary. However, some products have similar patterns in demand. So, it makes sense that products are bucketed and each bucket is studied separately according to its own characteristics. Additionally, because the **Description** offers a lot of literal information about products and thence could be perfectly used to compute the degree of similarity of two products based on the similarity, this attribute should be checked if they are applicable.

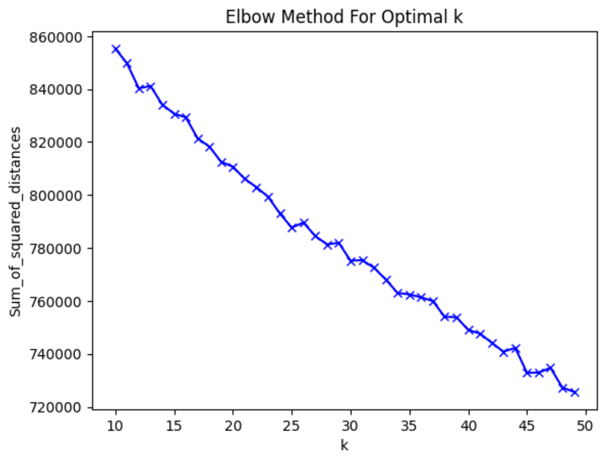


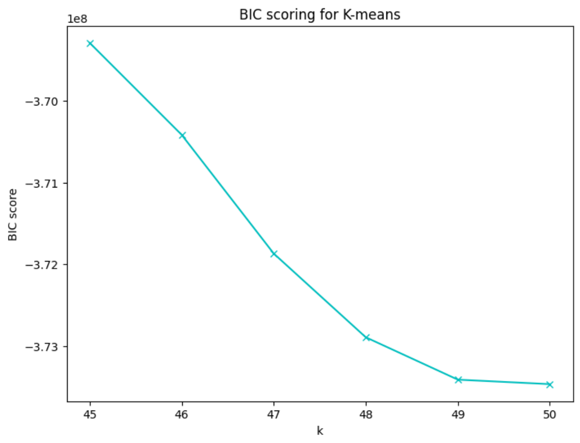
```
count                                   944044
unique                                    5668
top         WHITE HANGING HEART T-LIGHT HOLDER
freq                                      5528
Name: Description, dtype: object
```

After breezing through the **Description,** we find that a lot of strings, which contain special redundant characters, need sanitizing.

Data Cleaning

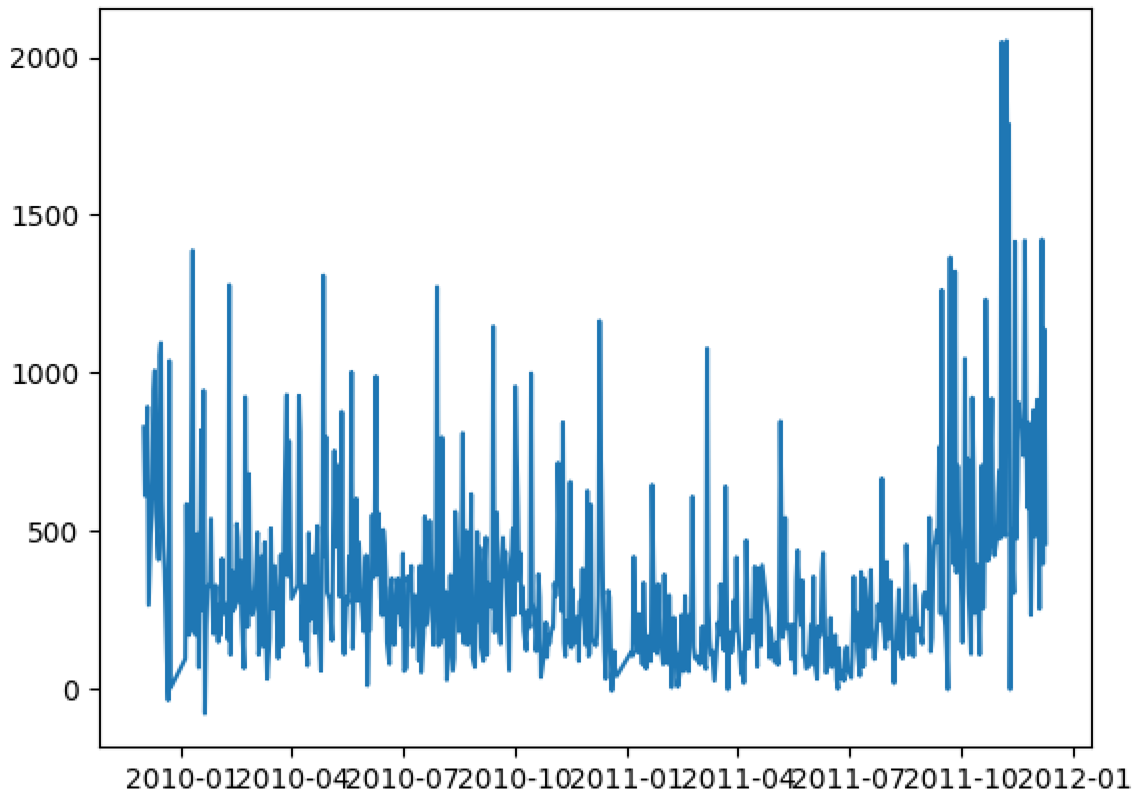
Procedure in Phase I is replicated excluding data aggregation. Prior to data aggregation, each product's **Description** is winnowed by the function preprocess_text() so that the **Description** is literal and useful in differentiating products. For the sake of clustering, transactions with blank under **Description** are deleted. What's more, the **Description** is vectorized through the TF-IDF calculation and the KMeans Clustering model takes vectorized **Description** as input and produces 47 categories. The metrics to determine why 47 is the number of buckets we need are Elbow Method and BIC scoring.





Furthermore, sales data is aggregated by **cluster** and **InvoiceDate.** But missing data and outliers are still popping up. Thus, **interpolate()** is called to fix missing data in the time series, and **IsolationForest()** catches outliers and replaces them with reasonable data.

Sample Data Visualization



Target Variable

Total_amount in sterling (Â£) per InvoiceDate/ Week is the **target variable**.

Predictive Variable

Direct Variable List:

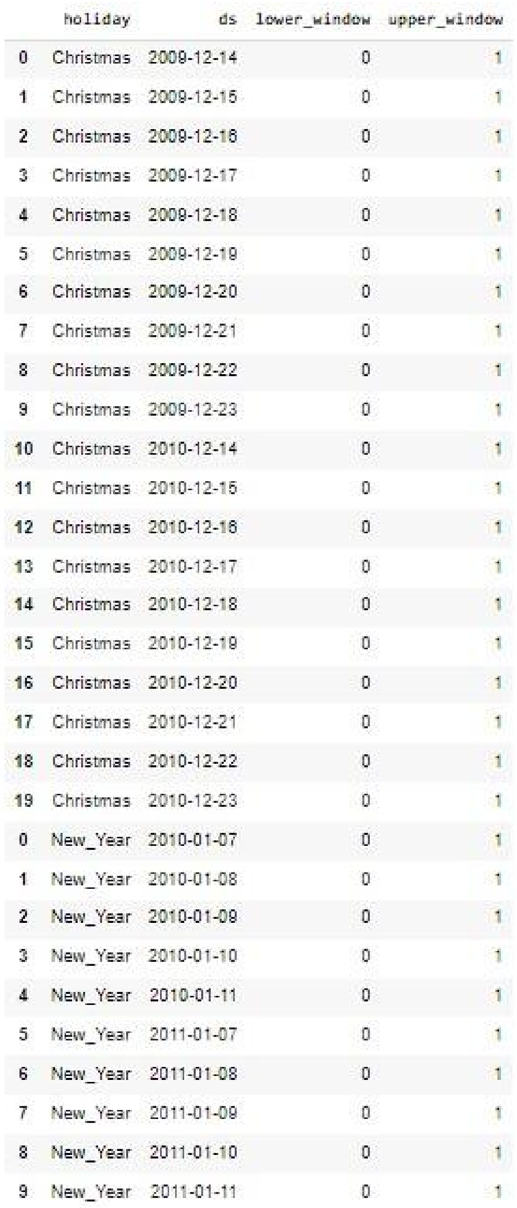
1. InvoiceDate: Features of time are critical in the modeling because they capture seasonality

Derived Variable List:

1. const: constant dummies
2. trend, trend_squared, trend_cubed: time trends
3. s(2,7), s(3, 7) …: weekly seasonality
4. sin(23,freq=A-DEC), cos(23,freq=A-DEC) …: annual seasonality
5. Holiday: US national holidays. The holiday has an effect on people's purchasing patterns.
6. clusters

Pre-modeling

1. Facebook Prophet: simply create a holiday feature by hand

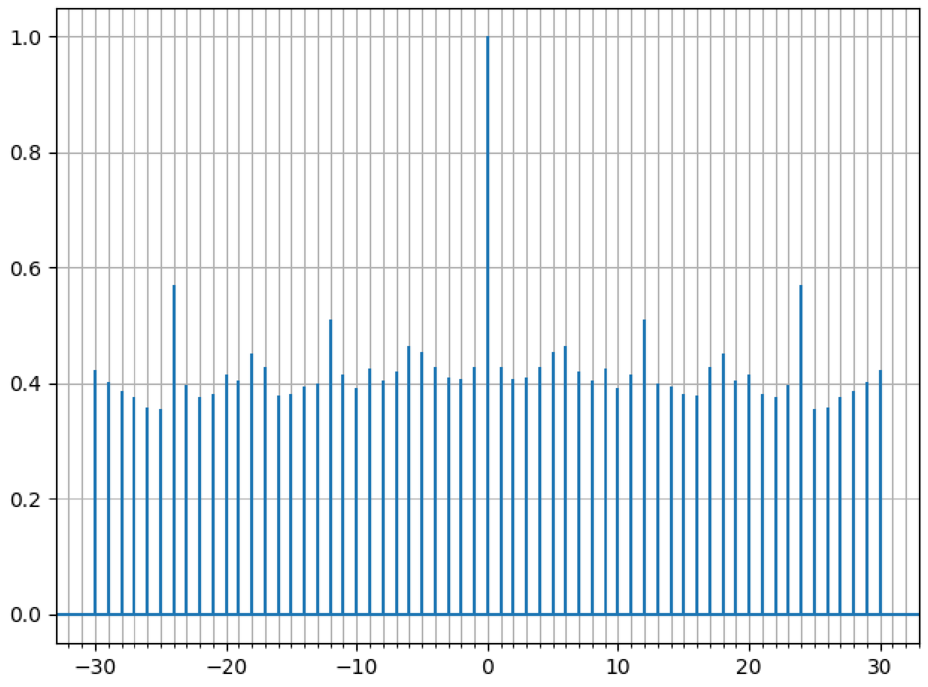


	holiday	ds	lower_window	upper_window
0	Christmas	2009-12-14	0	1
1	Christmas	2009-12-15	0	1
2	Christmas	2009-12-16	0	1
3	Christmas	2009-12-17	0	1
4	Christmas	2009-12-18	0	1
5	Christmas	2009-12-19	0	1
6	Christmas	2009-12-20	0	1
7	Christmas	2009-12-21	0	1
8	Christmas	2009-12-22	0	1
9	Christmas	2009-12-23	0	1
10	Christmas	2010-12-14	0	1
11	Christmas	2010-12-15	0	1
12	Christmas	2010-12-16	0	1
13	Christmas	2010-12-17	0	1
14	Christmas	2010-12-18	0	1
15	Christmas	2010-12-19	0	1
16	Christmas	2010-12-20	0	1
17	Christmas	2010-12-21	0	1
18	Christmas	2010-12-22	0	1
19	Christmas	2010-12-23	0	1
0	New_Year	2010-01-07	0	1
1	New_Year	2010-01-08	0	1
2	New_Year	2010-01-09	0	1
3	New_Year	2010-01-10	0	1
4	New_Year	2010-01-11	0	1
5	New_Year	2011-01-07	0	1
6	New_Year	2011-01-08	0	1
7	New_Year	2011-01-09	0	1
8	New_Year	2011-01-10	0	1
9	New_Year	2011-01-11	0	1

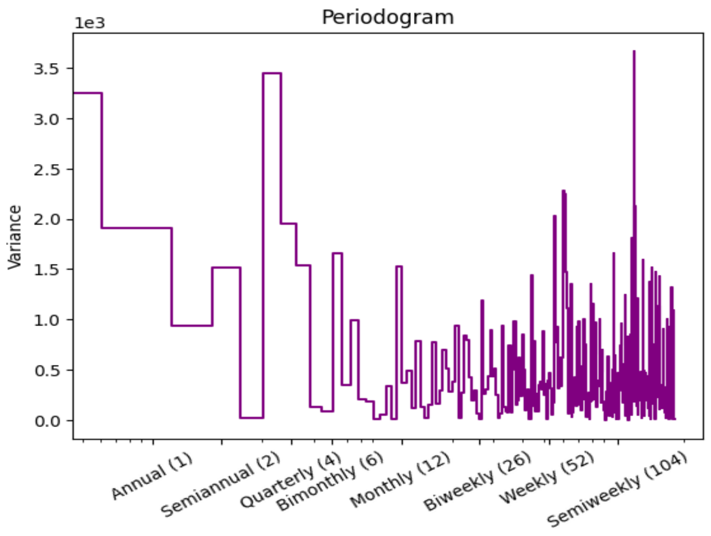
2. Random Forest Regression:

Data seasonality and trend analysis:

Since non-stationary factors disturb sales data, the following data seasonality and trend analysis proceeds. Following are the auto-correlation graphs with a maximum lag of 30 and 600 days, As lag increases, the auto-correlation gradually decreases and they perform a wave shape with a frequency of 7 days. That makes sense, since the sales may perform periodicity in the time scale of a single week.



The periodogram gives out the strength of the frequencies in a time series. Obviously, the periodogram below drops off between monthly and biweekly. Therefore, let’s use 18 Fourier pairs.



Derived variable creation:

Weekly, annual seasonal feature and trend features:

Source: **DeterministicProcess, CalendarFourier from statsmodels.tsa.deterministic**

Holiday features:

Source: Manually noting down **two weeks**' dates **prior to Christmas** and another two weeks' dates **post New Year** and **One-Hot encoding**

Facebook Prophet prediction features:

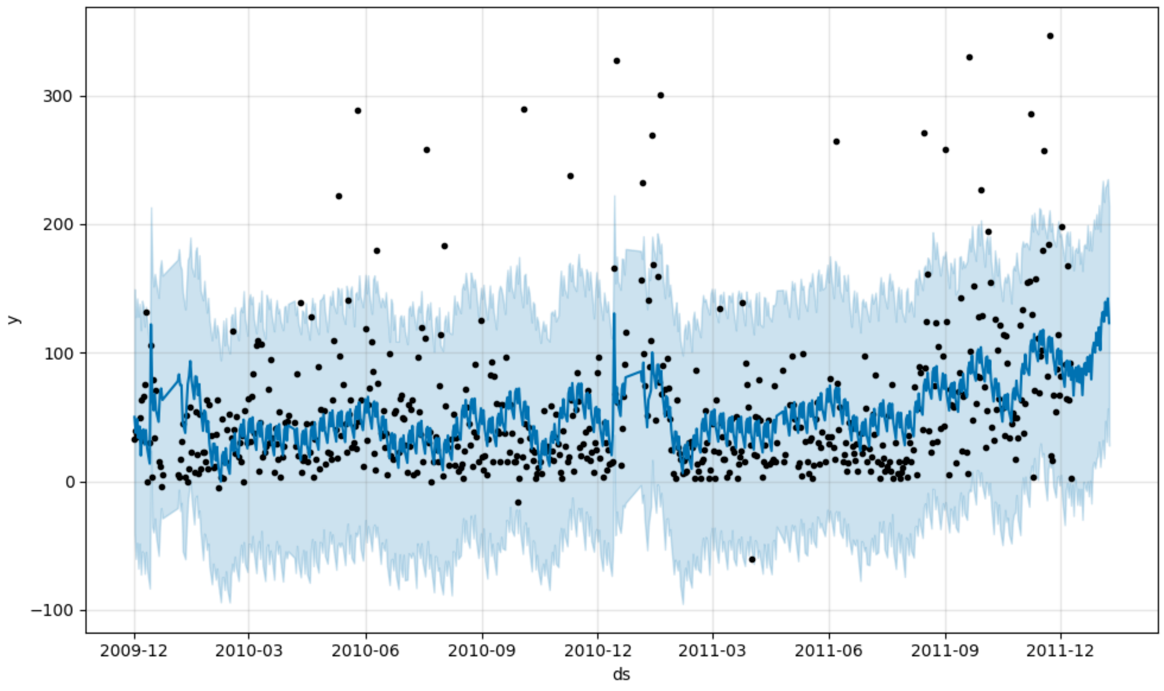
Source: **Prophet.predict() from prophet**

Modeling

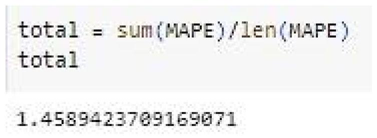
Firstly, we take advantage of **Facebook Prophet** model to make predictions. Secondly, the **Facebook Prophet** model is considered as a baseline and its predictions are treated as input in training the **Random Forest Regression** model.

Facebook Prophet:

Entire sales data in each cluster is chopped up into three different parts: train, valid, and test. For each cluster, we build a specific Facebook Prophet model with minimum MAPE after hyperparameter tuning and cross-validation. Further, FB has the functionality to generate future time frames so that we could predict the future even though there is no sales data available yet.



Despite all the effort in clustering and hyperparameters tunning, the final MAPE is far from acceptable.

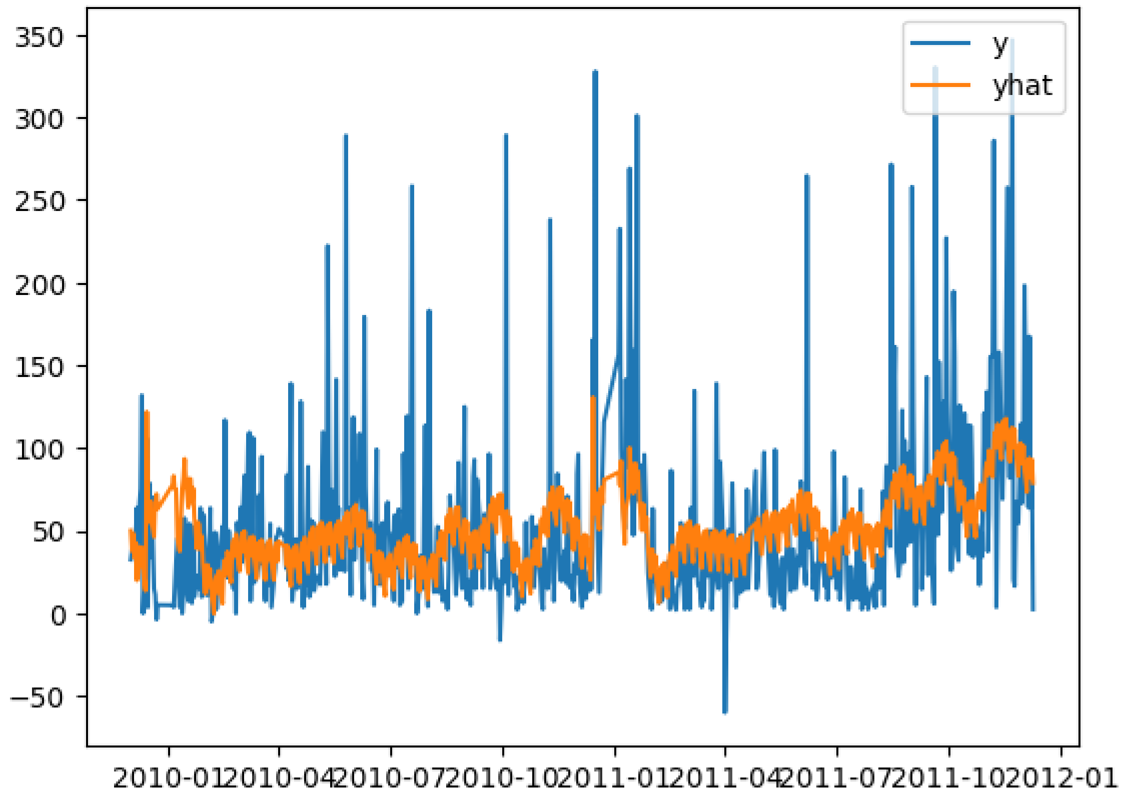


```
total = sum(MAPE)/len(MAPE)
total
```

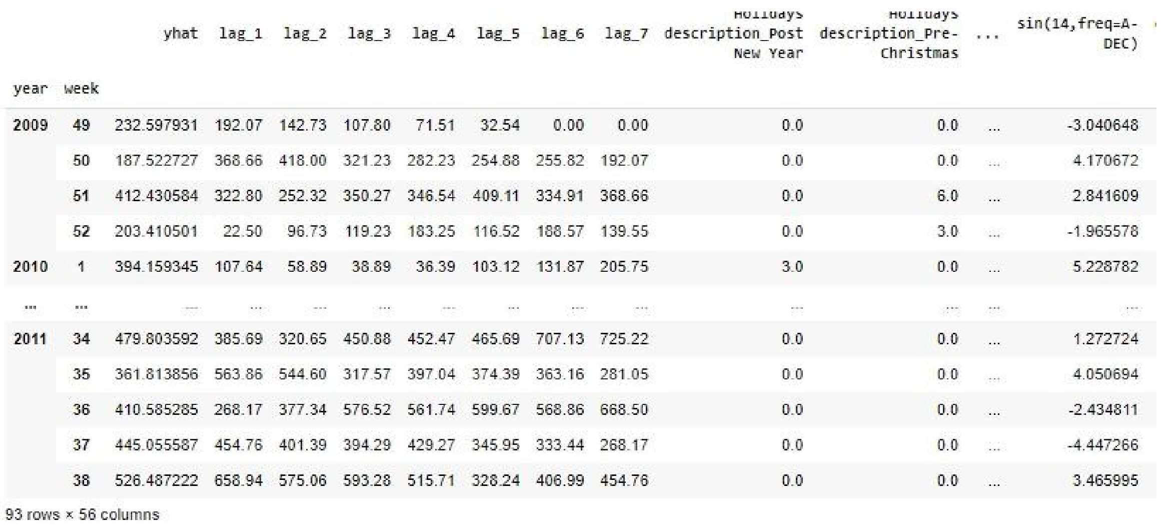
```
1.4589423709169071
```

Random Forest Regression:

The unsatisfied outcome from Facebook Prophet get me to think if Facebook Prophet is no help in forecasting sales data. Yet, a closer look at FB's prediction and actual data graph makes me realize that FB is good but not good enough. Thus, we decide to use FB's predictions as one of the features in training the Random Forest Regression model.



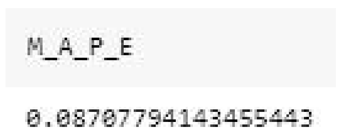
One more little detail before hopping into training models, this time instead of concentrating on daily sales data in each cluster, we shift our focus to weekly sales data created by summing up all features in daily sales data within one week.



year	week	yhat	lag_1	lag_2	lag_3	lag_4	lag_5	lag_6	lag_7	Holidays description_Post New Year	Holidays description_Pre-Christmas	...	sin(14,freq=A-DEC)
2009	49	232.597931	192.07	142.73	107.80	71.51	32.54	0.00	0.00	0.0	0.0	...	-3.040648
	50	187.522727	368.66	418.00	321.23	282.23	254.88	255.82	192.07	0.0	0.0	...	4.170672
	51	412.430584	322.80	252.32	350.27	346.54	409.11	334.91	368.66	0.0	6.0	...	2.841609
	52	203.410501	22.50	96.73	119.23	183.25	116.52	188.57	139.55	0.0	3.0	...	-1.965578
2010	1	394.159345	107.64	58.89	38.89	36.39	103.12	131.87	205.75	3.0	0.0	...	5.228782
...	...	...	...	...	...	...	...	...	...	...	...	...	...
2011	34	479.803592	385.69	320.65	450.88	452.47	465.69	707.13	725.22	0.0	0.0	...	1.272724
	35	361.813856	563.86	544.60	317.57	397.04	374.39	363.16	281.05	0.0	0.0	...	4.050694
	36	410.585285	268.17	377.34	576.52	561.74	599.67	568.86	668.50	0.0	0.0	...	-2.434811
	37	445.055587	454.76	401.39	394.29	429.27	345.95	333.44	268.17	0.0	0.0	...	-4.447266
	38	526.487222	658.94	575.06	593.28	515.71	328.24	406.99	454.76	0.0	0.0	...	3.465995

93 rows × 56 columns

Surprisingly, MAPE out of multiple Random Forest Regression models is extremely low.



```
M_A_P_E
```

```
0.08707794143455443
```

In hindsight, clustering really works out in terms of promoting models' prediction accuracy. However, what each cluster stands for is still a disturbing puzzle. Also, 47 clusters may be considered too many for classification purposes. Hence, we decide to give it a try on GPT and Google Bard when clustering in the next phase.

Phase III: Embrace GPT & Improve Forecasting

This section illustrates the ultimate stage of project workflow, where we already have experience as we go through Phase I and Phase II, and we are grinding to promote our forecasting accuracy. To be clear, this phase is called ultimate due to the fact that I already pushed it to my limit. Admittedly, there is a lot to improve, such as pipeline automation.

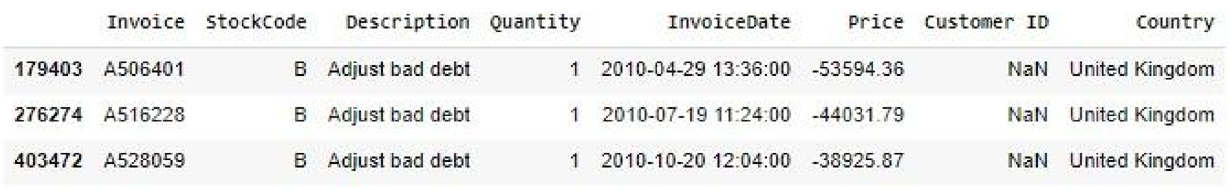
Data Processing

Data Diagnostics & Data Cleaning:

Unlike diagnostics before, this time's diagnostics is more comprehensive.

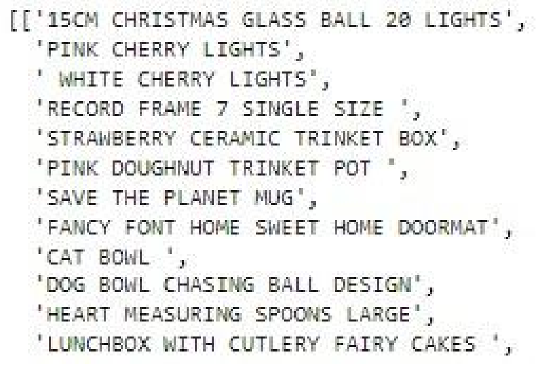
Firstly, after inspection, the **Price** of the null **Description** in the records is all zero. As a result, all these records are meaningless and removable.

Secondly, there are several codes starting with the letter '**A**' in the **Invoice.** We pull the records containing special codes up and figure out they represent adjustments on bad debt, which is needed in bookkeeping. Thus, we retain them.



	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
179403	A506401	B	Adjust bad debt	1	2010-04-29 13:36:00	-53594.36	NaN	United Kingdom
276274	A516228	B	Adjust bad debt	1	2010-07-19 11:24:00	-44031.79	NaN	United Kingdom
403472	A528059	B	Adjust bad debt	1	2010-10-20 12:04:00	-38925.87	NaN	United Kingdom

Thirdly, nonsense records are identified by looking at the **Description** and dropped off. Moreover, **the Description** is further cleansed by my getting rid of special characters.



```
[['15CM CHRISTMAS GLASS BALL 20 LIGHTS',
  'PINK CHERRY LIGHTS',
  ' WHITE CHERRY LIGHTS',
  'RECORD FRAME 7 SINGLE SIZE ',
  'STRAWBERRY CERAMIC TRINKET BOX',
  'PINK DOUGHNUT TRINKET POT ',
  'SAVE THE PLANET MUG',
  'FANCY FONT HOME SWEET HOME DOORMAT',
  'CAT BOWL ',
  'DOG BOWL CHASING BALL DESIGN',
  'HEART MEASURING SPOONS LARGE',
  'LUNCHBOX WITH CUTLERY FAIRY CAKES ',
```

Fourthly, we check if there is abnormal data in **Price.** After analyzing the attribute **Price**, I understand that because price = 0 has no impact on statistics for sales, we could delete all records with unit price = 0.

Eventually, we concatenate two worksheets into one.

Target Variable

total_sales in sterling (Â£) per transaction week is the **target variable**.

Predictive Variable

Direct Variable List:

1. InvoiceDate: Features of time are critical in the modeling because they capture seasonality

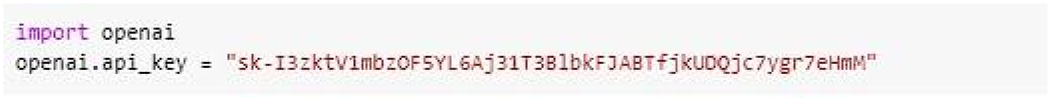
Derived Variable List:

1. const: constant dummies
2. trend, trend_squared, trend_cubed: time trends
3. s(2,7), s(3, 7) …: weekly seasonality
4. sin(23,freq=A-DEC), cos(23,freq=A-DEC) …: annual seasonality
5. Holiday: US national holidays. The holiday has an effect on people's purchasing patterns.
6. clusters

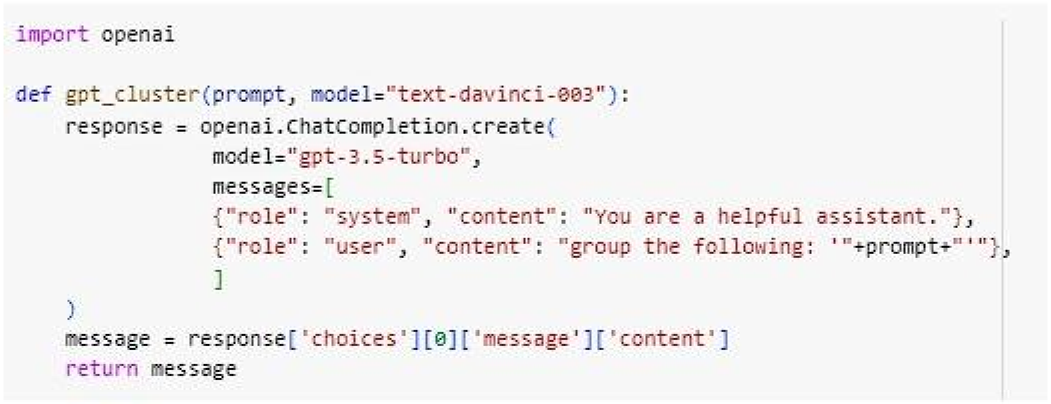
Pre-modeling

GPT cluster:

Thanks to Professor Syed Waseem Haider, I am motivated to apply GPT in the work of clustering sales data. Below is a snippet of codes when using the openai API and building up the GPT clustering model.



```python
import openai
openai.api_key = "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"
```



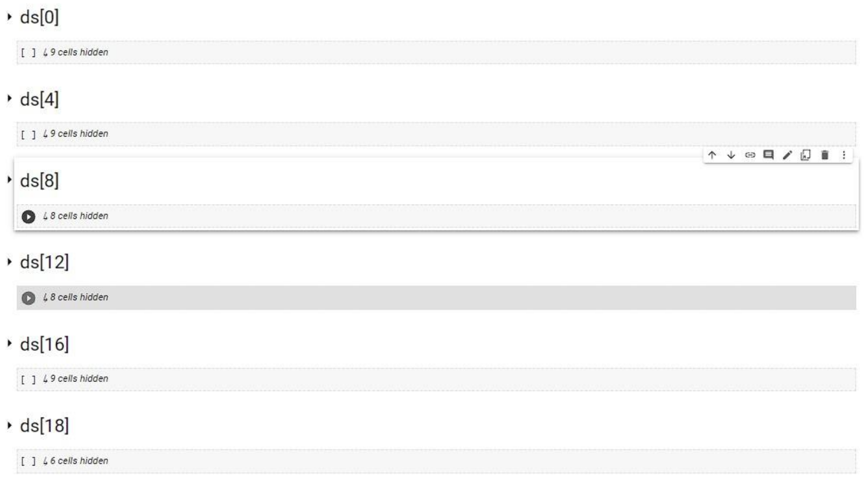
```python
import openai

def gpt_cluster(prompt, model="text-davinci-003"):
    response = openai.ChatCompletion.create(
                model="gpt-3.5-turbo",
                messages=[
                {"role": "system", "content": "You are a helpful assistant."},
                {"role": "user", "content": "group the following: '"+prompt+"'"},
                ]
    )
    message = response['choices'][0]['message']['content']
    return message
```

First and foremost, 5385 unique descriptions are extracted from the original sales data and transformed into the format accepted by the GPT-cluster model.

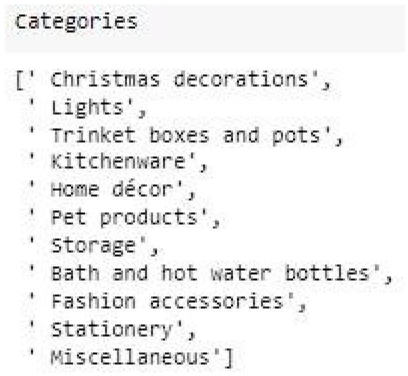
Then, it does not take me a long time to realize that over 5000 messages are way too large for GPT to digest and understand. Consequently, 5385 phrases are divided into 21 groups so that each group could fit into the GPT-cluster model.

Next, to shorten the time spent on clustering and at the same time make clusters as much comprehensive as possible, I sample the first group in every four groups.



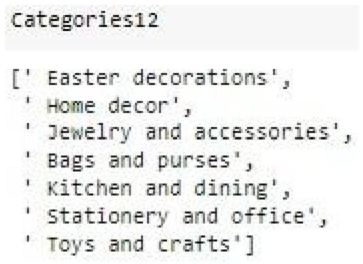
```
ds[0]
[ ] 9 cells hidden
ds[4]
[ ] 9 cells hidden
ds[8]
8 cells hidden
ds[12]
8 cells hidden
ds[16]
[ ] 9 cells hidden
ds[18]
[ ] 6 cells hidden
```

For each sampled group, the GPT-cluster model yields different bunches of categories with meaning. **However, input and output tokens are limited to GPT policies, which means only a small portion of descriptive phrases are clustered.**



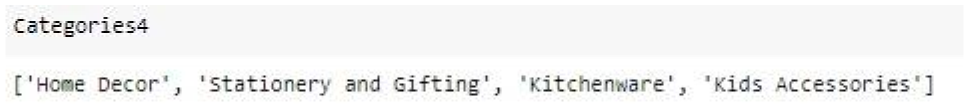
```
Categories

[' Christmas decorations',
 ' Lights',
 ' Trinket boxes and pots',
 ' Kitchenware',
 ' Home décor',
 ' Pet products',
 ' Storage',
 ' Bath and hot water bottles',
 ' Fashion accessories',
 ' Stationery',
 ' Miscellaneous']
```



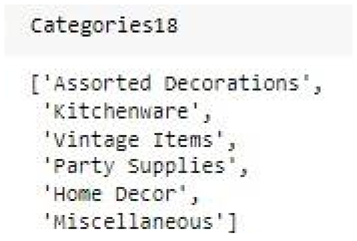
```
Categories12

[' Easter decorations',
 ' Home decor',
 ' Jewelry and accessories',
 ' Bags and purses',
 ' Kitchen and dining',
 ' Stationery and office',
 ' Toys and crafts']
```



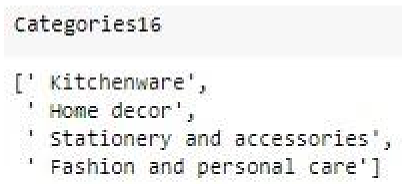
```
Categories4

['Home Decor', 'Stationery and Gifting', 'Kitchenware', 'Kids Accessories']
```



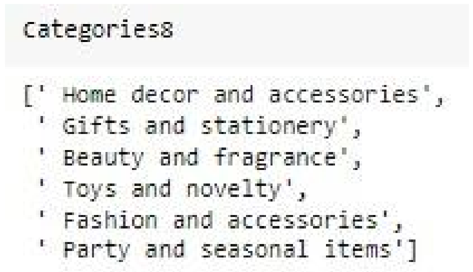
```
Categories18

['Assorted Decorations',
 'Kitchenware',
 'Vintage Items',
 'Party Supplies',
 'Home Decor',
 'Miscellaneous']
```



```
Categories16

[' Kitchenware',
 ' Home decor',
 ' Stationery and accessories',
 ' Fashion and personal care']
```



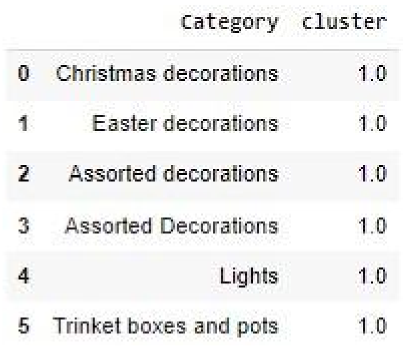
```
Categories8

[' Home decor and accessories',
 ' Gifts and stationery',
 ' Beauty and fragrance',
 ' Toys and novelty',
 ' Fashion and accessories',
 ' Party and seasonal items']
```

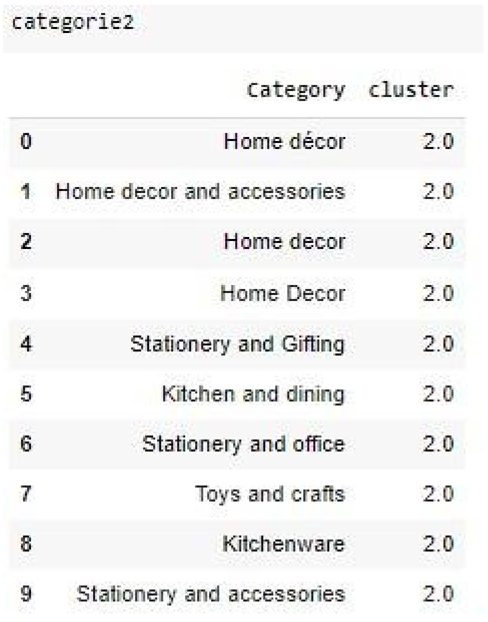
Since the literal meaning of these categories is well extended to cover all product descriptions, it is safe to conclude that **Description** could be classified under the categories listed above. But there are so many overlapping between obtained categories, ergo further cluster is in need.

Google Bard further cluster:

Since Google Bard API is not open to the public yet, I have to manually type input into the Google Bard online platform, retrieve the outcome documentation, and reverse engineer **Categories** into **cluster.**

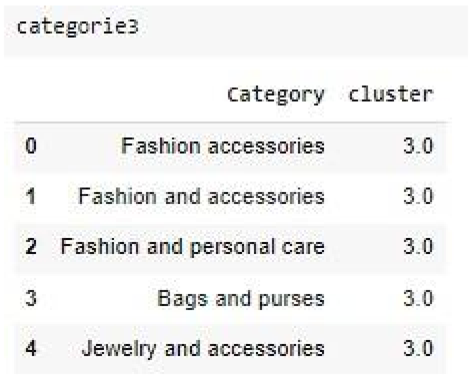


	Category	cluster
0	Christmas decorations	1.0
1	Easter decorations	1.0
2	Assorted decorations	1.0
3	Assorted Decorations	1.0
4	Lights	1.0
5	Trinket boxes and pots	1.0



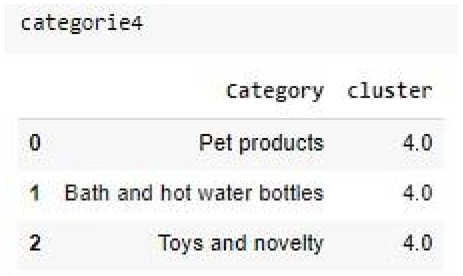
categorie2

	Category	cluster
0	Home décor	2.0
1	Home decor and accessories	2.0
2	Home decor	2.0
3	Home Decor	2.0
4	Stationery and Gifting	2.0
5	Kitchen and dining	2.0
6	Stationery and office	2.0
7	Toys and crafts	2.0
8	Kitchenware	2.0
9	Stationery and accessories	2.0



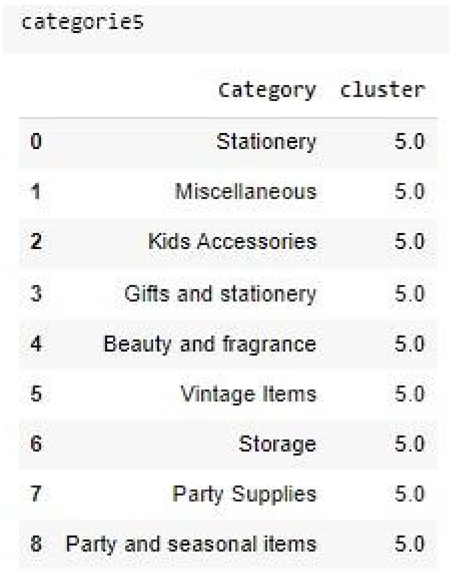
categorie3

	Category	cluster
0	Fashion accessories	3.0
1	Fashion and accessories	3.0
2	Fashion and personal care	3.0
3	Bags and purses	3.0
4	Jewelry and accessories	3.0



categorie4

	Category	cluster
0	Pet products	4.0
1	Bath and hot water bottles	4.0
2	Toys and novelty	4.0



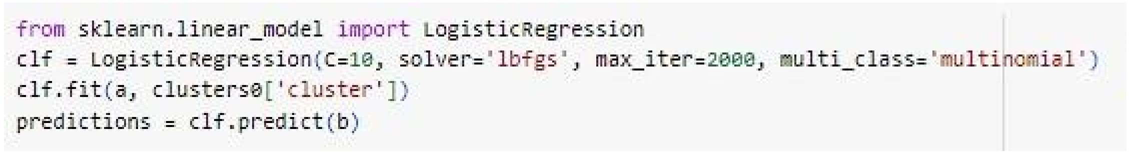
categorie5

	Category	cluster
0	Stationery	5.0
1	Miscellaneous	5.0
2	Kids Accessories	5.0
3	Gifts and stationery	5.0
4	Beauty and fragrance	5.0
5	Vintage Items	5.0
6	Storage	5.0
7	Party Supplies	5.0
8	Party and seasonal items	5.0

Semi-supervised classification:

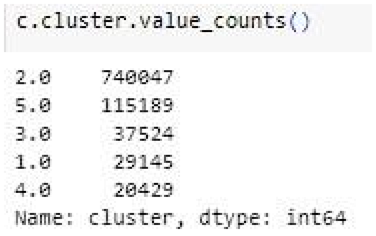
With the help of GPT-cluster and Google Bard, approximately 500 unique descriptive phrases are labeled with **Category** and **cluster**. However, compared to 5385, 500 is just a tiny fraction. Now the problem is evolved into a classification problem and semi-supervised machine learning is the only way out.

Firstly, literal data are vectorized through the TF-IDF calculation, and Logistic Regression is trained on labeled records, then able to recognize similarities among different descriptive phrases under **Description**. Its predictions on unlabeled data fill up blanks under **cluster**.



```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(C=10, solver='lbfgs', max_iter=2000, multi_class='multinomial')
clf.fit(a, clusters0['cluster'])
predictions = clf.predict(b)
```

The figure beneath shows the distribution of each **cluster**.



```
c.cluster.value_counts()

2.0    740047
5.0    115189
3.0     37524
1.0     29145
4.0     20429
Name: cluster, dtype: int64
```

Data aggregation:

Firstly, sales data is aggregated by **cluster** and **week** which is derived from **InvoiceDate.**

Data seasonality and trend analysis:

Seasonality and trend are analyzed by autocorrelation, PSD, seasonal decomposition, and periodogram graphs following the same logic in previous phases. For more details, check out the source code in Trends + seasonality on Jupyter Notebook called GPT/Bard_Clustering/XGBoost.ipynb.

Derived variable creation:

Seasonal features and trend features:

Source: **DeterministicProcess, CalendarFourier from statsmodels.tsa.deterministic**

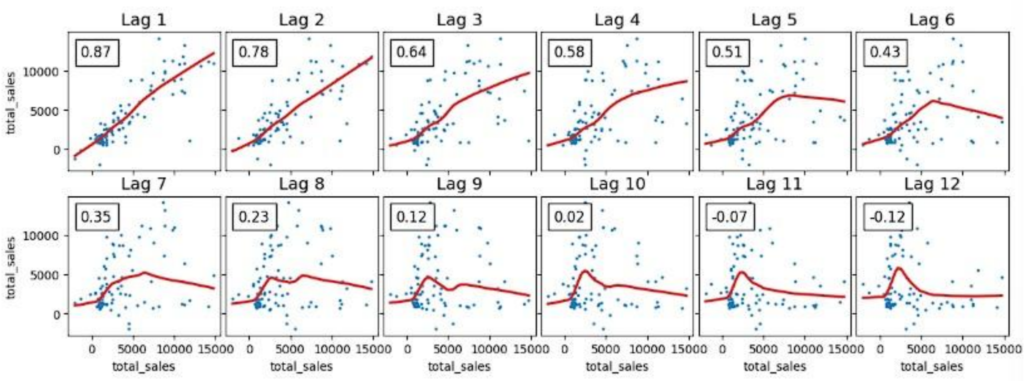
Holiday features:

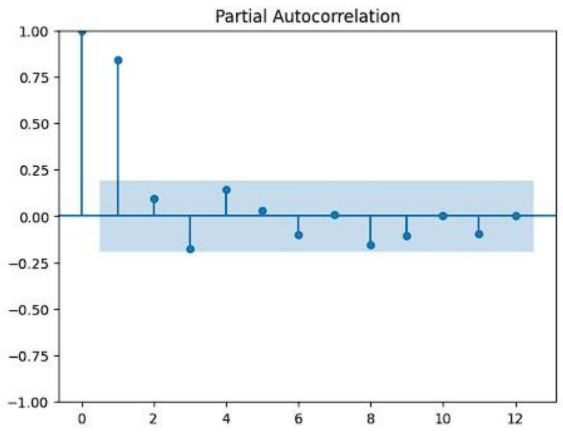
Source: Manually noting down **two weeks**' dates **prior to Christmas** and another two weeks' dates **post New Year** and **One-Hot encoding**

Lag features:

Source:**plot_pacf from statsmodels.graphics.tsaplots**

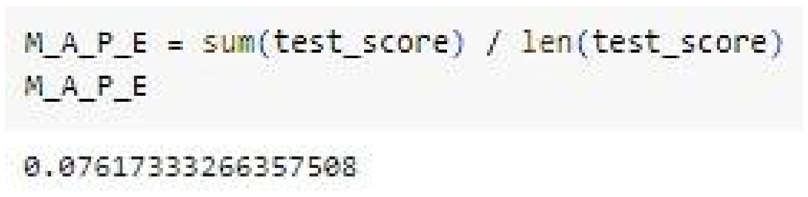
Directly read off from sequential graphs below.





Modeling

Since each cluster's weekly sales data has distinct features, it is legitimate to create different models and have them trained on a specific cluster accordingly. For each cluster, we build an XGBoost model with minimum MAPE after hyperparameter tuning and cross-validation. Finally, after all these efforts put in, a little bit of reduction in MAPE is acquired.



```
M_A_P_E = sum(test_score) / len(test_score)
M_A_P_E
```

```
0.07617333266357508
```

Next Step

Pipeline automation would be a research direction worthy of digging deeper into. It would save me a lot of time if GPT and Google Bard are integrated with some handy API so that I could just write Python scripts to implement functions instead of going back and forth between different platforms and programming languages to search for answers.

Another potential research direction is the trade-off between clusters' explainability and the speed of getting reliable forecasting models.