Online Retail II

– Sales Forecasting

Yunpeng Wang

Agenda

I. Introduction
II. Phase I
III. Phase II
IV. Phase III
V. Summary




Introduction

online retail transaction data from UCI Machine Learning Repository

Time span of data set

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.

Objective

Perform a time series analysis and Predict the future sales data

Phase Ⅰ models:
Linear Regression;
SARIMAX

Phase Ⅱ models:
K-means clustering
Facebook Prophet
Random Forest Regression

Phase Ⅲ models:
GPT
Google Bard
XGBoost




Phase I

plain linear regression/ SARIMAX model

Data Processing

Check missing data and outliers
prior to data cleaning process

Visualize time series

Data Diagnostics | Data Cleaning | Data Visualization

Unify unit of InvoiceDate
Only retain records in the UK
Delete duplicate records

Target Variable & Predictive Variable

- Total_amount in sterling (Â£) per InvoiceDate is the target 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. 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 test

Data seasonality and trend analysis

Autocorrelation
Power Spectral Density
Seasonality decomposition

Moving average

Derived variable creation

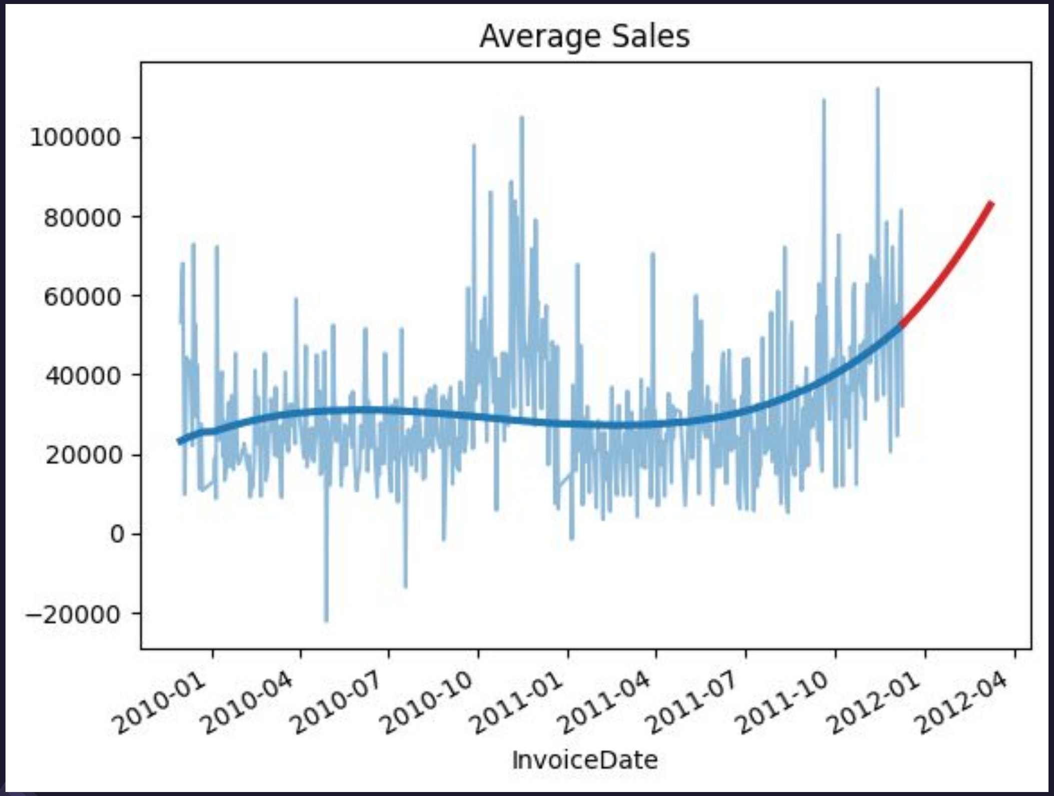
DeterministicProcess,

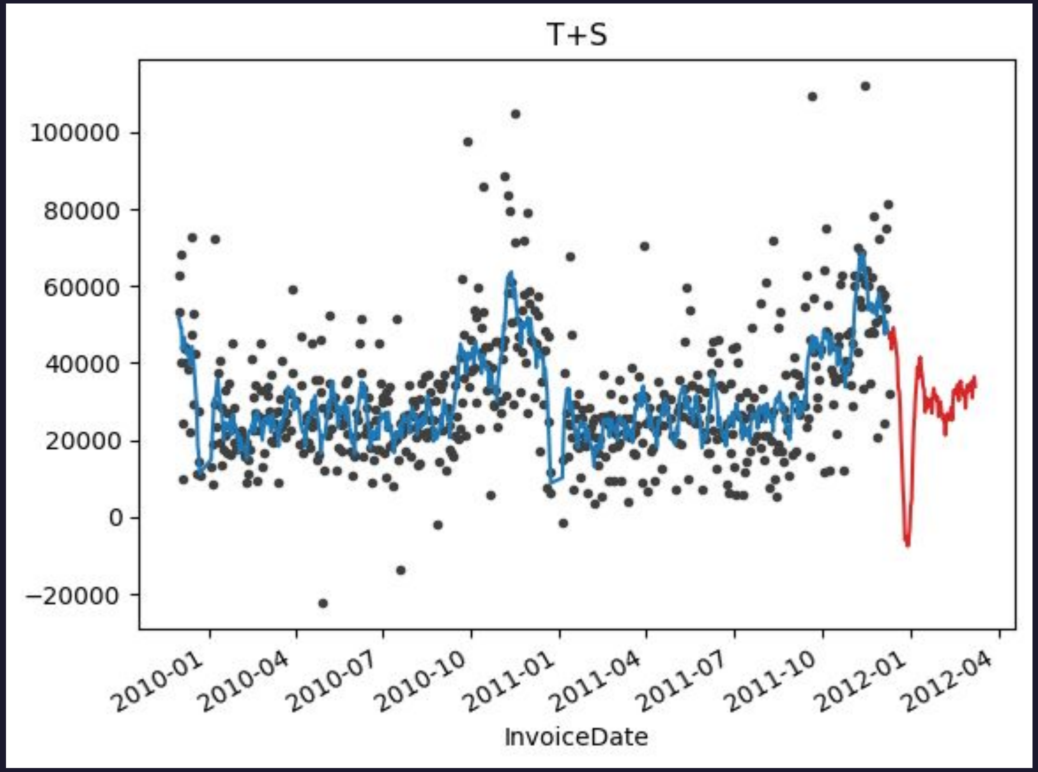
CalendarFourier from

statsmodels.tsa.deterministic

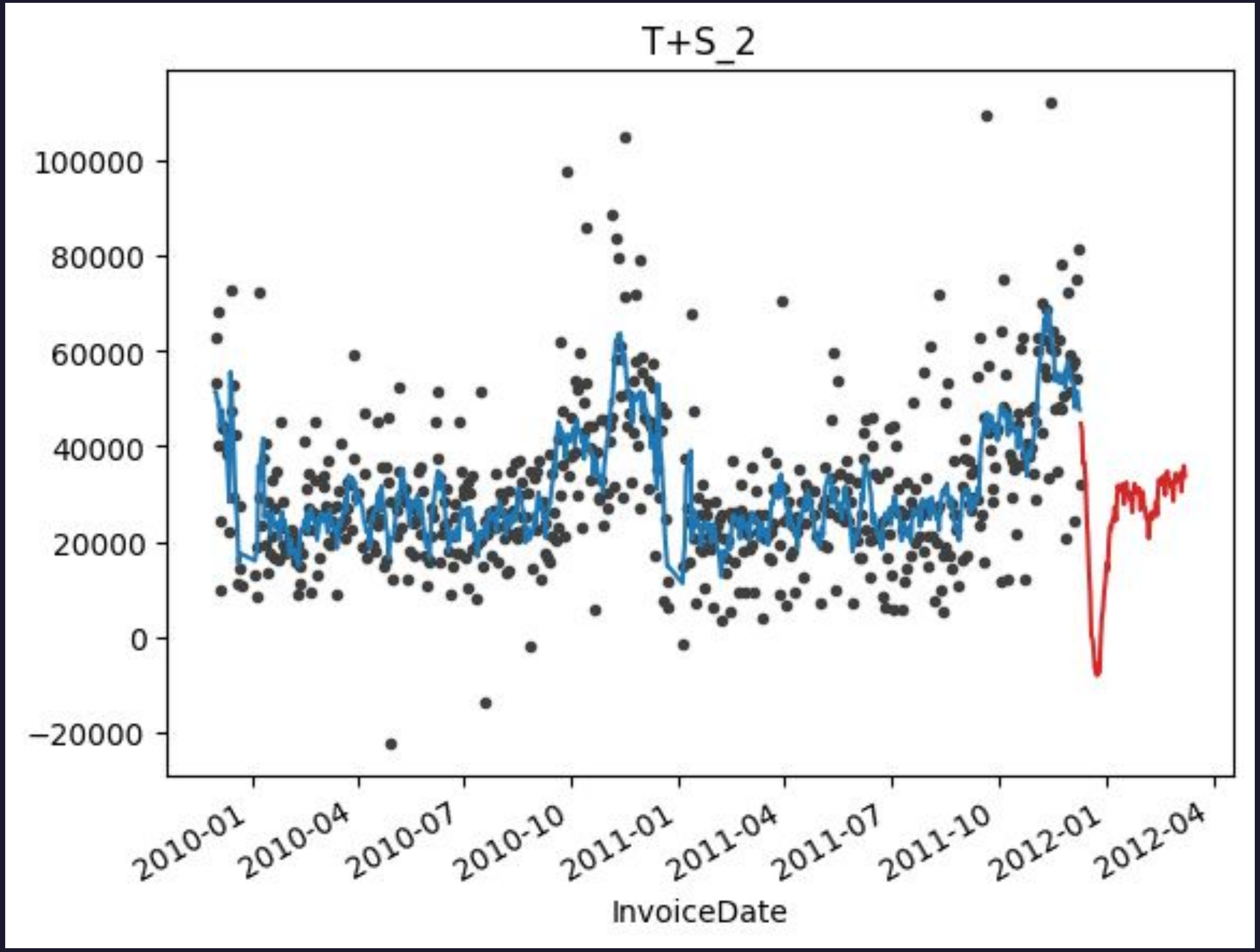
One-Hot encoding holiday features

Modeling – Linear Regression





Modeling – Linear Regression



Modeling -SARIMAX

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

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

	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




Phase II

KMeans_cluster + Facebook Prophet/ Random Forest Regression

Data Processing

Check out missing data and outliers in Description column

Visualize time series

Data Diagnostics | Data Cleaning | Data Visualization

Sanitize Description
Vectorize Description
K-means Clustering
Interpolate()
IsolationForest()

Target Variable & Predictive Variable

- Total_amount in sterling (Â£) per InvoiceDate/ transaction week is the target 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 (Facebook Prophet)

	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

Pre-modeling (Random Forest Regression)

DeterministicProcess, CalendarFourier

from statsmodels.tsa.deterministic

One-Hot encoding holiday features

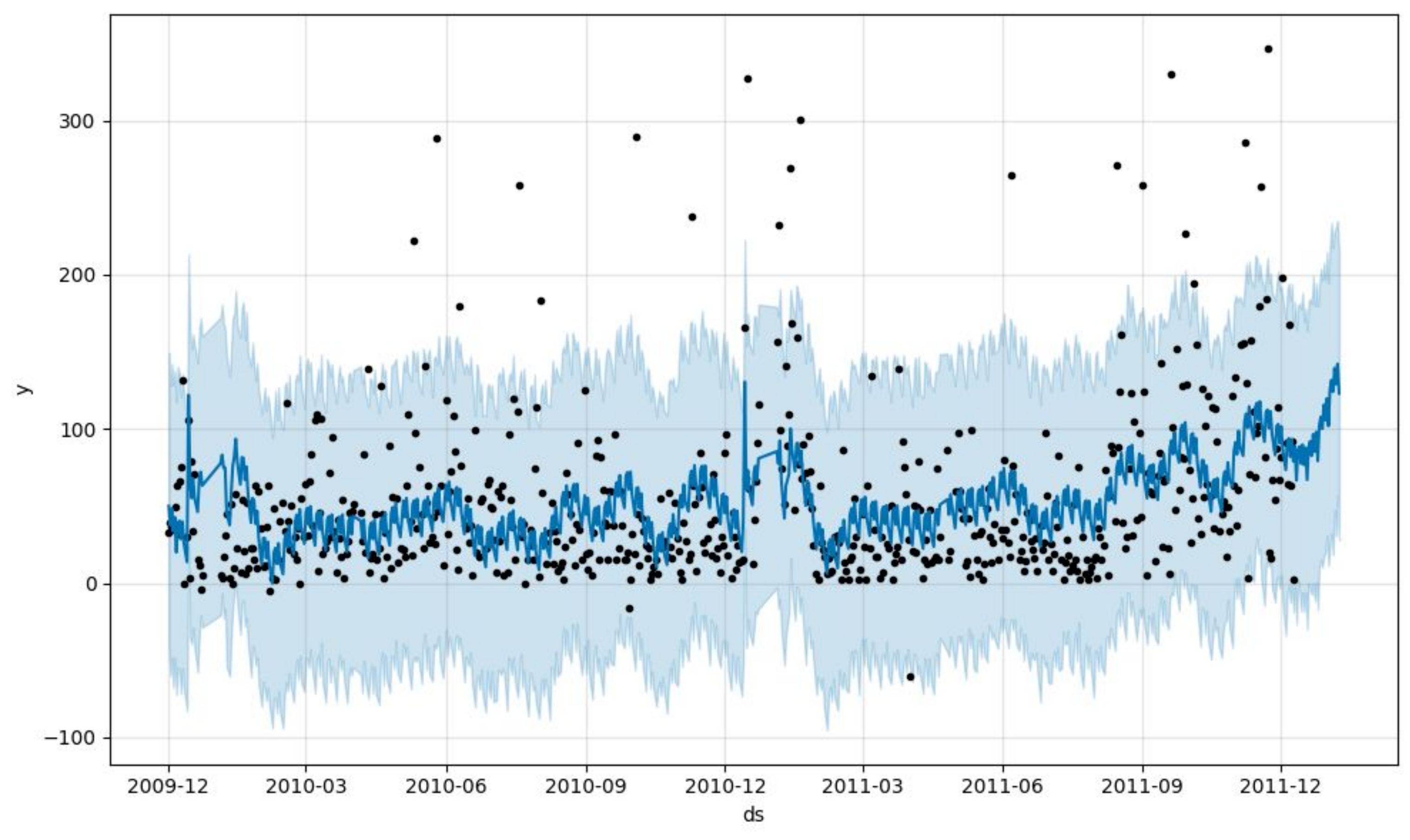
Prophet.predict() from prophet

Data seasonality and trend analysis

Derived variable creation

Autocorrelation
Power Spectral Density
Seasonality decomposition

Modeling – Facebook Prophet




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

```
1.4589423709169071
```

Modeling – Random Forest Regression

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

Screenshot of features

M_A_P_E
0.08707794143455443




Phase Ⅲ

GPT-cluster+ Google Bard + XGBoost

Data Processing

Check out missing data and outliers in Description and Price column

Data Diagnostics

Data Cleaning

Sanitize Description
Drop off records with Price = 0
Merge two worksheets together

Target Variable & Predictive Variable

- Total_amount in sterling (Â£) per transaction week is the target 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. 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.
7. clusters

Pre-modeling – GPT-cluster

ChatGPT

```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
```

Pre-modeling– Google Bard further cluster

Bard AI

	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

Pre-modeling –Semi-supervised classification

```python
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)
```

```python
c.cluster.value_counts()
```

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

Pre-modeling

DeterministicProcess, CalendarFourier

from statsmodels.tsa.deterministic

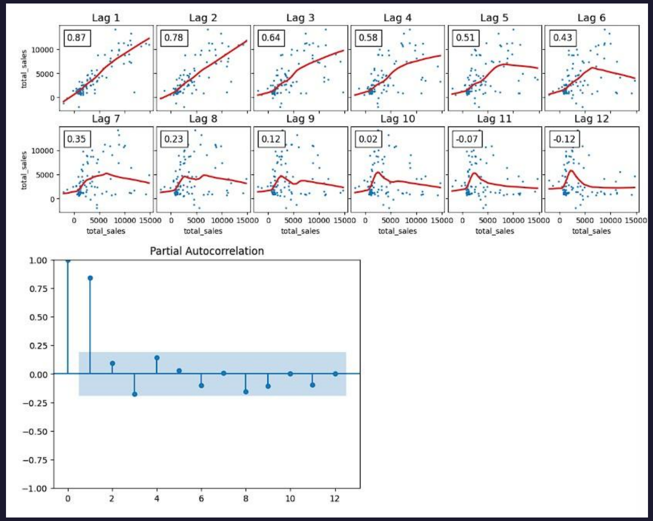
One-Hot encoding holiday features

plot_pacf from statsmodels.graphics.tsaplots

Data seasonality and trend analysis

Derived variable creation

Autocorrelation
Power Spectral Density
Seasonality decomposition



modeling

```python
def TimeSplit_ModBuild(model, paramGrid, splits, X, y):
    #Loop over each time split and for each
    for train_index, val_index in splits.split(X):
        _X_train_ = X.iloc[train_index]
        _y_train_ = y.iloc[train_index]
        _X_val_ = X.iloc[val_index]
        _y_val_ = y.iloc[val_index]

        train_scores = []
        val_scores = []

        # Loop through the parameter grid, set the hyperparameters, and save the scores
        for g in paramGrid:
            model.set_params(**g)
            model.fit(_X_train_, _y_train_)
            p_train = model.predict(_X_train_)
            p_val = model.predict(_X_val_)
            score_train = mean_absolute_percentage_error(_y_train_, p_train)
            score_val = mean_absolute_percentage_error(_y_val_, p_val)
            train_scores.append(score_train)
            val_scores.append(score_val)
            #models.append(model)
            best_idx = np.argmin(val_scores)

        print("Best-Fold HyperParams:: ", paramGrid[best_idx])
        print("Best-Fold Train MAPE: ", train_scores[best_idx])
        print("Best-Fold Val MAPE: ",val_scores[best_idx])
        print("\n")

    #Return most recent model
    return train_scores, val_scores, best_idx
```

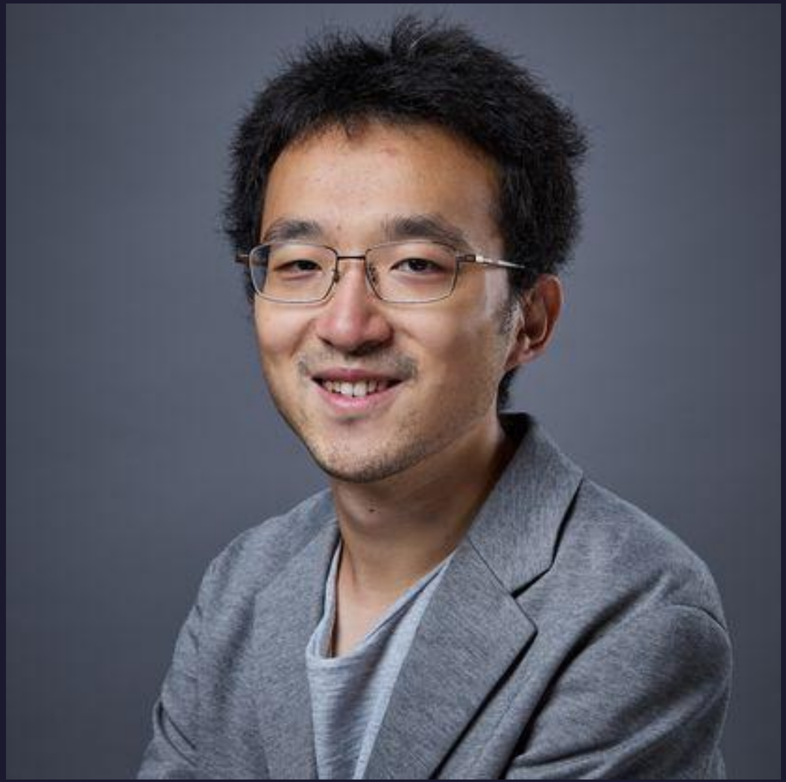
```python
M_A_P_E = sum(test_score) / len(test_score)
M_A_P_E
```

```
0.07617333266357508
```

Next Step

- Pipeline automation would be a research direction worthy of digging deeper into.
- Another potential research direction is the trade-off between clusters' explainability and the speed of getting reliable forecasting models.

One man gang



Yunpeng Wang