VIETNAM NATIONAL UNIVERSITY

UNIVERSITY OF INFORMATION TECHNOLOGY

INFORMATION SYSTEMS FACULTY

REPORT LAB 4

SUBJECT: DATA ANALYSIS IN BUSINESS

Lecturer: Assoc. Prof. Nguyen Dinh Thuan

Instructor: TA. Nguyen Minh Nhut

Class: IS403.O22.HTCL

Group 3:

21521049 – Ho Quang Lam

21521586 – Le Thi Le Truc

21521938 – Nguyen Thanh Dat

Ho Chi Minh City, April 2024

ACKNOWLEDGEMENT

First of all, we would like to express our deepest gratitude and appreciation to our lecturers, Mr. Nguyen Dinh Thuan and Mr. Nguyen Minh Nhut, for their teaching and sharing of extensive knowledge as well as practical examples during the lectures. They have guided us in completing our Lab 01 report by providing valuable feedback, suggestions, and assistance with exercises and revisions.

The Data Analysis in Business course is an interesting and highly practical subject. However, due to our limited expertise and initial unfamiliarity with realworld applications, we acknowledge that our Lab 01 report may contain some shortcomings and inaccuracies despite our best efforts. We sincerely hope to receive further guidance and feedback from Mr. Nguyen Dinh Thuan and Mr. Nguyen Minh Nhut to improve our knowledge and equip ourselves for future projects as well as for our academic and professional endeavors.

Once again, we would like to extend our heartfelt and sincere gratitude to our lecturers and peers!

Ho Chi Minh City, April 2024

Group of student performers

Ho Quang Lam

Le Thi Le Truc

Nguyen Thanh Dat

LECTURER'S COMMENTS

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

TABLE OF CONTENTS

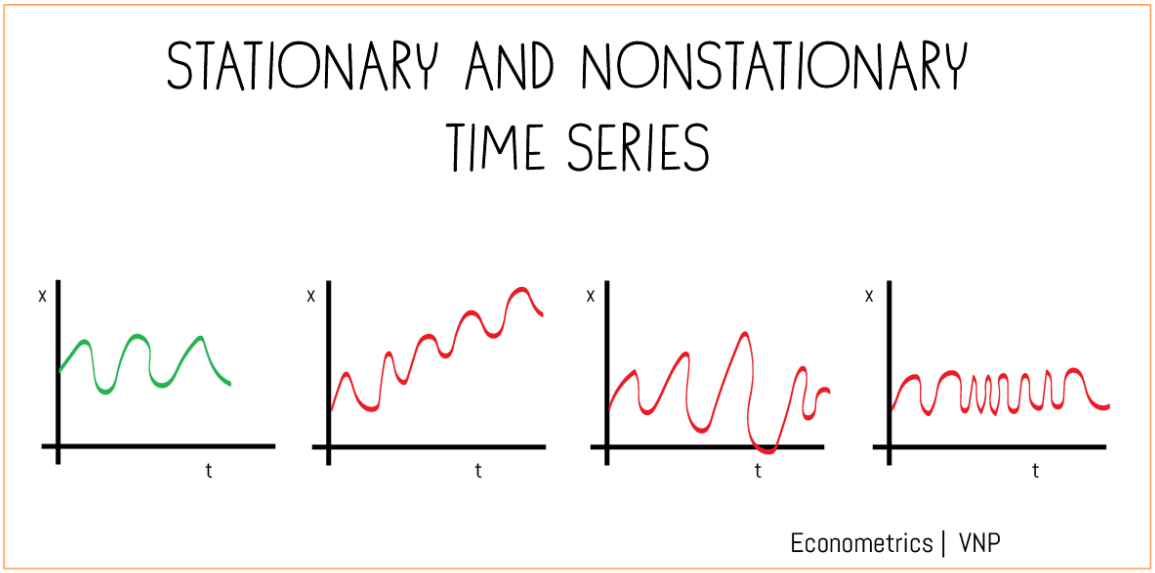
WORK DISTRIBUTION

Members / Works	Le Thi Le Truc (Leader)	Ho Quang Lam	Nguyen Thanh Dat
Problem statement	✓	✓	✓
Build the report template	✓		
Do all exercise with Excel in task 4.1	✓		
Definition	✓	✓	✓
Do all exercise with Python in task 4.1		✓	
Do all exercise with Excel in task 4.2	✓		
Do all exercise with Python in task 4.2		✓	
Do all exercise with Python in task 4.3			✓
Summarize and edit reports	✓	✓	✓
Completion	**100%**	**100%**	**100%**

Chapter 1. DEFINITION

1.1. STATIONARY

A stationary time series is a series of mean, variance, and autocorrelation values that do not change over time and do not include trend elements. stopping the string data because it is important to check the stopping of the calculation

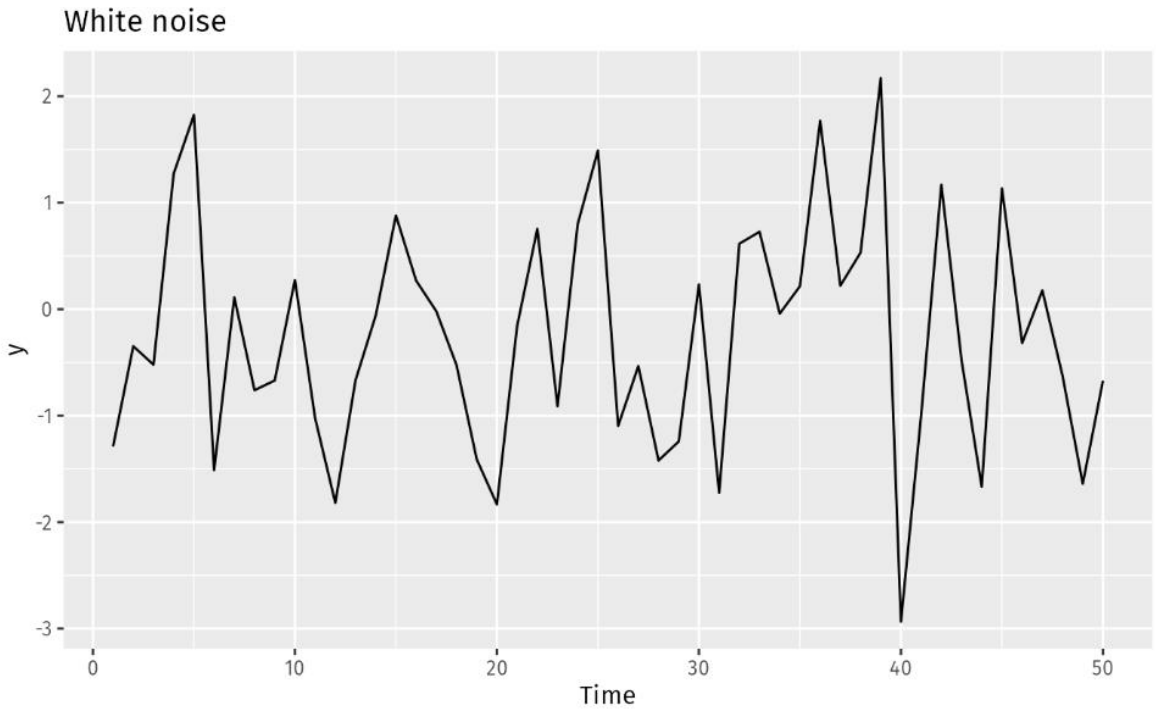


1.2. WHITE NOISE

Time series that show no autocorrelation are called white noise.

The “noise” is because there’s no pattern, just random variation. If you listened to a realization of white noise as an audio file, you would hear a static sound.

The “white” is because all freqencies are equally represented. This will become clear when we do frequency domain analysis of time series.



1.3. ARIMA

ARIMA model is the abbreviation for automatic recovery process (Auto Regression -AR), moving average process (Moving Average – MA) and wrong analysis Integrated - I

1.3.1. AR

AR(p): Autoregression - is the process of finding the relationship between current data and p previous data (lag)

$$y_t = a_0 + a_1 y_{t-1} + a_2 y_{t-2} + ... + a_p y_{t-p} + \varepsilon_t$$

Stop condition of choosing p:

$$\sum_{i=0}^{p} a_i < 1$$

1.3.2. I

I(d): Integrated - Compare the difference between d observations (difference between the current value and d previous values)

Month	Gas Use	First Deffence	Second Deffence	Third Deffence	Fourth Deffence
Jan	244				
Feb	228	-16			
Mar	153	-75	-59		
Apr	140	-13	62	121	
May	55	-85	-72	-134	-255
Jun	34	-21	64	136	270
Jul	30	-4	17	-47	-183
Aug	28	-2	2	-15	32
Sep	29	1	3	1	16
Oct	41	12	11	8	7
Nov	88	47	35	24	16
Dec	199	111	64	29	5

1.3.3. MA

MA(q): Moving Average: is the process of finding a relationship between current data and q past errors

$$y_t = \beta_0 + \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \ldots + \beta_q \varepsilon_{t-q} + \mu_t$$

Stop condition of choosing p:

$$\sum_{i=0}^{q} \beta_i < 1$$

1.4. ACF AND PACF

1.4.1. ACF

Autocorrelation is a mathematical representation of the degree of similarity between a given time series and a lagged version of itself over successive time intervals. It's conceptually like the correlation between two different time series, but autocorrelation uses the same time series twice: once in its original form and once lagged one or more time periods

$$ACF(k) = \frac{\sum_{t=k+1}^{n}(x_t - \mu)(x_{t-k} - \mu)}{\sum_{t=1}^{n}(x_t - \mu)^2}$$

Where:

- **n**: the length of time series.
- **x(t)**: the value at time t.
- **μ**: the mean of time series.

1.4.2. PACF

PACF (Partial Autocorrelation Function) which is a partial correlation between observations at time t and previous times.

In the ACF model, we have variables that are correlated with each other in an indirect form. Additionally, with PACF, for each lag, we can find a direct correlation from the observation at time t to the observation at a specific time t-k. PACF basically eliminates the effects of other delays

$$PACF(k) = r_{kk} = corr(Y_t, Y_{t-k} \mid Y_{t-1}, Y_{t-2}, \dots, Y_{t-k+1})\ [5]$$

$$= \frac{\mathbf{Cov}(y_t, y_{t+k} | y_{t+1}, y_{t+2}, \dots, y_{t+k-1})}{\sqrt{\boldsymbol{Var}(y_t \mid y_{t+1}, y_{t+2}, \dots, y_{t+k-1})\boldsymbol{Var}(y_{t+1} \mid y_{t+1}, y_{t+2}, \dots, y_{t+k-1})}}$$

Chapter 2. GAS & ELECTRIC

4.1. Using MS Excel and Python language to implement the ARIMA algorithm and another machine learning algorithm on the Gas & Electric dataset.

2.1. ARIMA

2.1.1. USING EXCEL

2.1.1.1. GAS

We have ARIMA (4, 0, 0) model. Means that:

$$Y_t = \mu + \varphi_1 Y_{t-1} + \varphi_2 Y_{t-2} + \varphi_3 Y_{t-3} + \varphi_4 Y_{t-4}$$

Calculate the lag to find the AR(q) and delete all rows which have empty value

Month	Gas Use
Jan	244
Feb	228
Mar	153
Apr	140
May	55
Jun	34
Jul	30
Aug	28
Sep	29
Oct	41
Nov	88
Dec	199
Jan	230
Feb	245
Mar	247
Apr	135
May	34
Jun	33
Jul	27
Aug	26
Sep	28
Oct	39
Nov	86
Dec	188

LAG1	LAG2	LAG3	LAG4
244			
228	244		
153	228	244	
140	153	228	244
55	140	153	228
34	55	140	153
30	34	55	140
28	30	34	55
29	28	30	34
41	29	28	30
88	41	29	28
199	88	41	29
230	199	88	41
245	230	199	88
247	245	230	199
135	247	245	230
34	135	247	245
33	34	135	247
27	33	34	135
26	27	33	34
28	26	27	33
39	28	26	27
86	39	28	26

Month	Gas Use	LAG1	LAG2	LAG3	LAG4
May	55	140	153	228	244
Jun	34	55	140	153	228
Jul	30	34	55	140	153
Aug	28	30	34	55	140
Sep	29	28	30	34	55
Oct	41	29	28	30	34
Nov	88	41	29	28	30
Dec	199	88	41	29	28
Jan	230	199	88	41	29
Feb	245	230	199	88	41
Mar	247	245	230	199	88
Apr	135	247	245	230	199
May	34	135	247	245	230
Jun	33	34	135	247	245
Jul	27	33	34	135	247
Aug	26	27	33	34	135
Sep	28	26	27	33	34
Oct	39	28	26	27	33
Nov	86	39	28	26	27
Dec	188	86	39	28	26

We have a coefficient table using Excel tool

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0,9145191
R Square	0,836345184
Adjusted R Squ	0,7927039
Standard Error	37,74794786
Observations	20

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	4	109228,1865	27307,04662	19,16408281	9,25028E-06
Residual	15	21373,61351	1424,907568		
Total	19	130601,8			

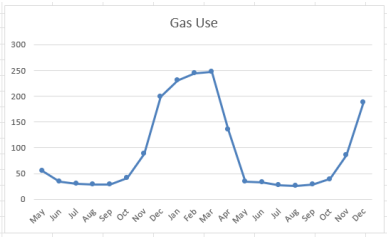
	Coefficients	*Standard Error*	*t Stat*	***P-value***	*Lower 95%*	*Upper 95%*	*Lower 95,0%*	*Upper 95,0%*
Intercept	**61,09284143**	17,9938765	3,395201775	**0,003996942**	22,73980154	99,44588133	22,73980154	99,44588133
LAG1	**1,114772925**	0,244198948	4,565019358	**0,000371928**	0,594275188	1,635270662	0,594275188	1,635270662
LAG2	**-0,474937936**	0,359578343	-1,320819079	**0,206358924**	-1,241361031	0,29148516	-1,241361031	0,29148516
LAG3	**0,1712355**	0,354192116	0,48345373	**0,635754809**	-0,583707126	0,926178125	-0,583707126	0,926178125
LAG4	**-0,378767285**	0,232050045	-1,63226551	**0,123434381**	-0,873370248	0,115835678	-0,873370248	0,115835678

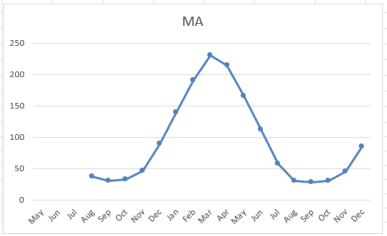
After having all necessary values, we start to predict based on this equation

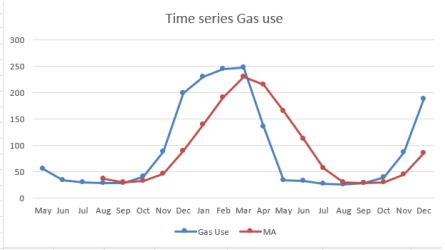
$$Y_t = 61.093 + 1.115 * Y_{t-1} - 0.475 * Y_{t-2} + 0.171 * Y_{t-3} - 0.079 * Y_{t-4}$$

We have a graph showing "Gas Use"

Month	Gas Use	MA
May	55	#N/A
Jun	34	#N/A
Jul	30	#N/A
Aug	28	36,75
Sep	29	30,25
Oct	41	32
Nov	88	46,5
Dec	199	89,25
Jan	230	139,5
Feb	245	190,5
Mar	247	230,25
Apr	135	214,25
May	34	165,25
Jun	33	112,25
Jul	27	57,25
Aug	26	30
Sep	28	28,5
Oct	39	30
Nov	86	44,75
Dec	188	85,25







2.1.1.2. ELECTRIC

We have ARIMA (2, 0, 0) model. Means that:

$$Y_t = \mu + \varphi_1 Y_{t-1} + \varphi_2 Y_{t-2}$$

Calculate the lag to find the AR(q) and delete all rows which have empty value

Month	Electric Use
Jan	967
Feb	795
Mar	820
Apr	672
May	722
Jun	820
Jul	1326
Aug	1262
Sep	1126
Oct	814
Nov	821
Dec	918
Jan	950
Feb	878
Mar	785
Apr	690
May	794
Jun	802
Jul	1445
Aug	1357
Sep	1268
Oct	889
Nov	830
Dec	935

LAG1	LAG2
967	
795	967
820	795
672	820
722	672
820	722
1326	820
1262	1326
1126	1262
814	1126
821	814
918	821
950	918
878	950
785	878
690	785
794	690
802	794
1445	802
1357	1445
1268	1357
889	1268
830	889
935	830

Month	Electric Use	LAG1	LAG2
Mar	820	795	967
Apr	672	820	795
May	722	672	820
Jun	820	722	672
Jul	1326	820	722
Aug	1262	1326	820
Sep	1126	1262	1326
Oct	814	1126	1262
Nov	821	814	1126
Dec	918	821	814
Jan	950	918	821
Feb	878	950	918
Mar	785	878	950
Apr	690	785	878
May	794	690	785
Jun	802	794	690
Jul	1445	802	794
Aug	1357	1445	802
Sep	1268	1357	1445
Oct	889	1268	1357
Nov	830	889	1268
Dec	935	830	889

We have a coefficient table using Excel tool

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0,639684851
R Square	0,409196709
Adjusted R Square	0,347006889
Standard Error	188,7133967
Observations	22

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	2	468649,6427	234324,8213	6,579802097	0,00674074
Residual	19	676642,1755	35612,74608		
Total	21	1145291,818			

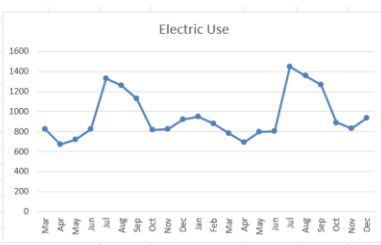
	Coefficients	*Standard Error*	*t Stat*	***P-value***	*Lower 95%*	*Upper 95%*	*Lower 95,0%*	*Upper 95,0%*
Intercept	**607,210134**	193,1351865	3,143964313	**0,005344147**	202,9735429	1011,446725	202,9735429	1011,447
LAG1	**0,754103218**	0,208038607	3,624823436	**0,001803621**	0,31867341	1,189533027	0,31867341	1,189533
LAG2	**-0,387548599**	0,209252637	-1,852060769	**0,079620161**	-0,825519401	0,050422203	-0,825519401	0,050422

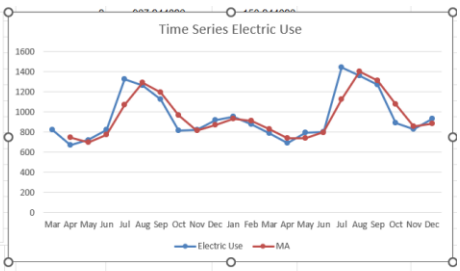
After having all necessary values, we start to predict based on this equation

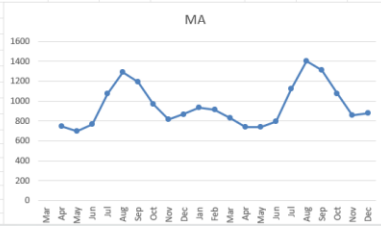
$$Y_t = 607.21 + 0.754 * Y_{t-1} - 0.387 * Y_{t-2}$$

We have a graph showing "Electric Use"

Month	Electric Use	MA
Mar	820	#N/A
Apr	672	746
May	722	697
Jun	820	771
Jul	1326	1073
Aug	1262	1294
Sep	1126	1194
Oct	814	970
Nov	821	817,5
Dec	918	869,5
Jan	950	934
Feb	878	914
Mar	785	831,5
Apr	690	737,5
May	794	742
Jun	802	798
Jul	1445	1123,5
Aug	1357	1401
Sep	1268	1312,5
Oct	889	1078,5
Nov	830	859,5
Dec	935	882,5







2.1.2. USING PYTHON

2.1.2.1. GAS

First, we find the best ARIMA model

```
[ ]  # Training process
     x_train = np.array(train_data.index).reshape(-1, 1)
     y_train = np.array(train_data['Gas Use '])

     # Find the best ARIMA model using auto_arima
     from pmdarima.arima import auto_arima
     model = auto_arima(y_train, trace=True, error_action='ignore', suppress_warnings=True)

     # Fit the model
     model.fit(y_train)
```

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.26 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=192.159, Time=0.01 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=177.668, Time=0.05 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=180.683, Time=0.08 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=209.529, Time=0.01 sec
 ARIMA(2,0,0)(0,0,0)[0] intercept   : AIC=173.876, Time=0.07 sec
 ARIMA(3,0,0)(0,0,0)[0] intercept   : AIC=172.021, Time=0.12 sec
 ARIMA(4,0,0)(0,0,0)[0] intercept   : AIC=171.477, Time=0.21 sec
 ARIMA(5,0,0)(0,0,0)[0] intercept   : AIC=173.422, Time=0.21 sec
 ARIMA(4,0,1)(0,0,0)[0] intercept   : AIC=173.232, Time=0.23 sec
 ARIMA(3,0,1)(0,0,0)[0] intercept   : AIC=inf, Time=0.14 sec
 ARIMA(5,0,1)(0,0,0)[0] intercept   : AIC=175.246, Time=0.27 sec
 ARIMA(4,0,0)(0,0,0)[0]             : AIC=179.717, Time=0.06 sec

Best model:  ARIMA(4,0,0)(0,0,0)[0] intercept
Total fit time: 1.788 seconds
```

ARIMA
ARIMA(4,0,0)(0,0,0)[0] intercept

We found the best model is ARMIA(4,0,0). After that, we calculate the model parameters

```
[ ]  model.summary()
```

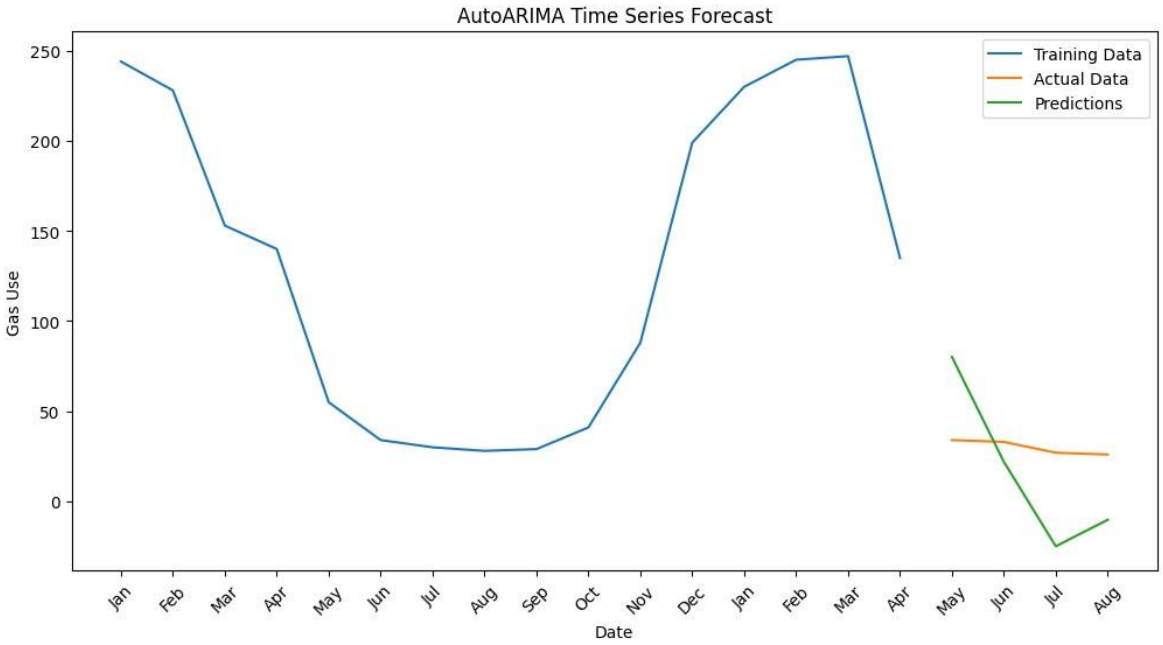
SARIMAX Results

Dep. Variable:	y	**No. Observations:**	16
Model:	SARIMAX(4, 0, 0)	**Log Likelihood**	-79.739
Date:	Sun, 28 Apr 2024	**AIC**	171.477
Time:	10:30:10	**BIC**	176.113
Sample:	0	**HQIC**	171.715
	- 16		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	74.4257	20.290	3.668	0.000	34.658	114.193
ar.L1	0.7719	0.412	1.873	0.061	-0.036	1.580
ar.L2	0.0765	0.810	0.094	0.925	-1.511	1.664
ar.L3	-0.0287	0.823	-0.035	0.972	-1.641	1.584
ar.L4	-0.4799	0.502	-0.956	0.339	-1.464	0.504
sigma2	958.5288	458.323	2.091	0.036	60.232	1856.826

Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	0.35
Prob(Q):	0.98	**Prob(JB):**	0.84
Heteroskedasticity (H):	1.61	**Skew:**	0.29
Prob(H) (two-sided):	0.61	**Kurtosis:**	2.57

We draw a chart



2.1.2.2. ELECTRIC

First, we find the best ARIMA model

```
# Training process
x_train = np.array(train_data.index).reshape(-1, 1)
y_train = np.array(train_data['Electric Use'])

# Find the best ARIMA model using auto_arima
from pmdarima.arima import auto_arima
model = auto_arima(y_train, trace=True, error_action='ignore', suppress_warnings=True)

# Fit the model
model.fit(y_train)
```

```
Performing stepwise search to minimize aic
 ARIMA(2,0,2)(0,0,0)[0] intercept   : AIC=inf, Time=0.52 sec
 ARIMA(0,0,0)(0,0,0)[0] intercept   : AIC=216.521, Time=0.03 sec
 ARIMA(1,0,0)(0,0,0)[0] intercept   : AIC=213.152, Time=0.09 sec
 ARIMA(0,0,1)(0,0,0)[0] intercept   : AIC=213.171, Time=0.17 sec
 ARIMA(0,0,0)(0,0,0)[0]             : AIC=265.675, Time=0.02 sec
 ARIMA(2,0,0)(0,0,0)[0] intercept   : AIC=211.509, Time=0.26 sec
 ARIMA(3,0,0)(0,0,0)[0] intercept   : AIC=211.543, Time=0.36 sec
 ARIMA(2,0,1)(0,0,0)[0] intercept   : AIC=213.946, Time=0.50 sec
 ARIMA(1,0,1)(0,0,0)[0] intercept   : AIC=213.838, Time=0.26 sec
 ARIMA(3,0,1)(0,0,0)[0] intercept   : AIC=212.177, Time=0.70 sec
 ARIMA(2,0,0)(0,0,0)[0]             : AIC=219.742, Time=0.04 sec

Best model:  ARIMA(2,0,0)(0,0,0)[0] intercept
Total fit time: 3.001 seconds
```

ARIMA
ARIMA(2,0,0)(0,0,0)[0] intercept

We found the best model is ARMIA(2,0,0). After that, we calculate the model parameters

```
[ ]  model.summary()
```

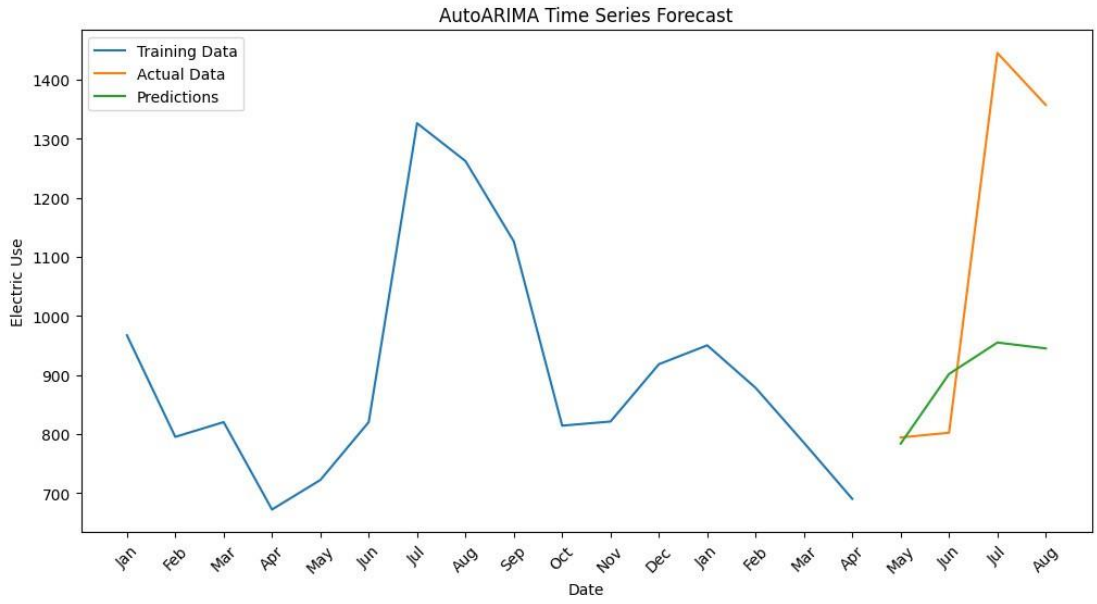
SARIMAX Results

Dep. Variable:	y	**No. Observations:**	16
Model:	SARIMAX(2, 0, 0)	**Log Likelihood**	-101.754
Date:	Sun, 28 Apr 2024	**AIC**	211.509
Time:	10:49:50	**BIC**	214.599
Sample:	0	**HQIC**	211.667
	- 16		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	578.8288	272.955	2.121	0.034	43.846	1113.811
ar.L1	0.8064	0.629	1.281	0.200	-0.427	2.040
ar.L2	-0.4480	0.491	-0.913	0.361	-1.410	0.513
sigma2	1.859e+04	7934.966	2.342	0.019	3033.272	3.41e+04

Ljung-Box (L1) (Q):	0.37	**Jarque-Bera (JB):**	10.74
Prob(Q):	0.55	**Prob(JB):**	0.00
Heteroskedasticity (H):	0.41	**Skew:**	1.45
Prob(H) (two-sided):	0.35	**Kurtosis:**	5.78

We draw a chart



2.2. LINEAR REGRESSION

2.2.1. USING EXCEL

2.2.1.1. GAS

We have a coefficient table using Excel tool

Month	Month_num	Gas Use
Jan	1	244
Feb	2	228
Mar	3	153
Apr	4	140
May	5	55
Jun	6	34
Jul	7	30
Aug	8	28
Sep	9	29
Oct	10	41
Nov	11	88
Dec	12	199
Jan	1	230
Feb	2	245
Mar	3	247
Apr	4	135
May	5	34

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0,505274406
R Square	0,255302225
Adjusted R Squar	0,221452326
Standard Error	76,36042475
Observations	24

ANOVA

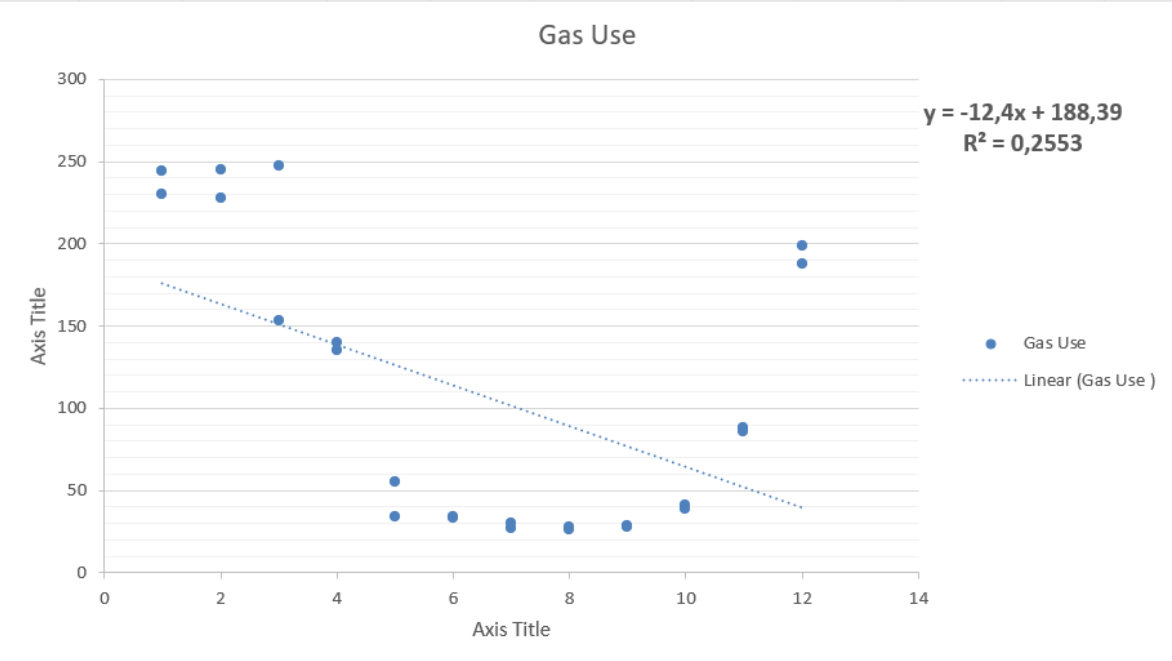
	df	*SS*	*MS*	*F*	*Significance F*
Regression	1	43977,84	43977,84	7,542186	0,011784375
Residual	22	128280,1	5830,914		
Total	23	172258			

	Coefficients	*andard Err*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*ower 95,0%*	*Upper 95,0%*
Intercept	**188,3939394**	33,23161	5,669119	1,06E-05	119,4757997	257,3121	119,4758	257,3120791
Month_num	**-12,40034965**	4,515287	-2,7463	0,011784	-21,76448087	-3,03622	-21,7645	-3,036218435

After having all necessary values, we start to predict based on this equation

Gas use = 188.393 – 12.4*Month_num

We have a graph showing "Gas Use"



2.2.1.2. ELECTRIC

We have a coefficient table using Excel tool

Month	Month_num	Electric Use
Jan	1	967
Feb	2	795
Mar	3	820
Apr	4	672
May	5	722
Jun	6	820
Jul	7	1326
Aug	8	1262
Sep	9	1126
Oct	10	814
Nov	11	821
Dec	12	918
Jan	1	950
Feb	2	878
Mar	3	785
Apr	4	690
May	5	794
Jun	6	802

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0,25708881
R Square	0,06609465
Adjusted R Square	0,02364441
Standard Error	222,773822
Observations	24

ANOVA

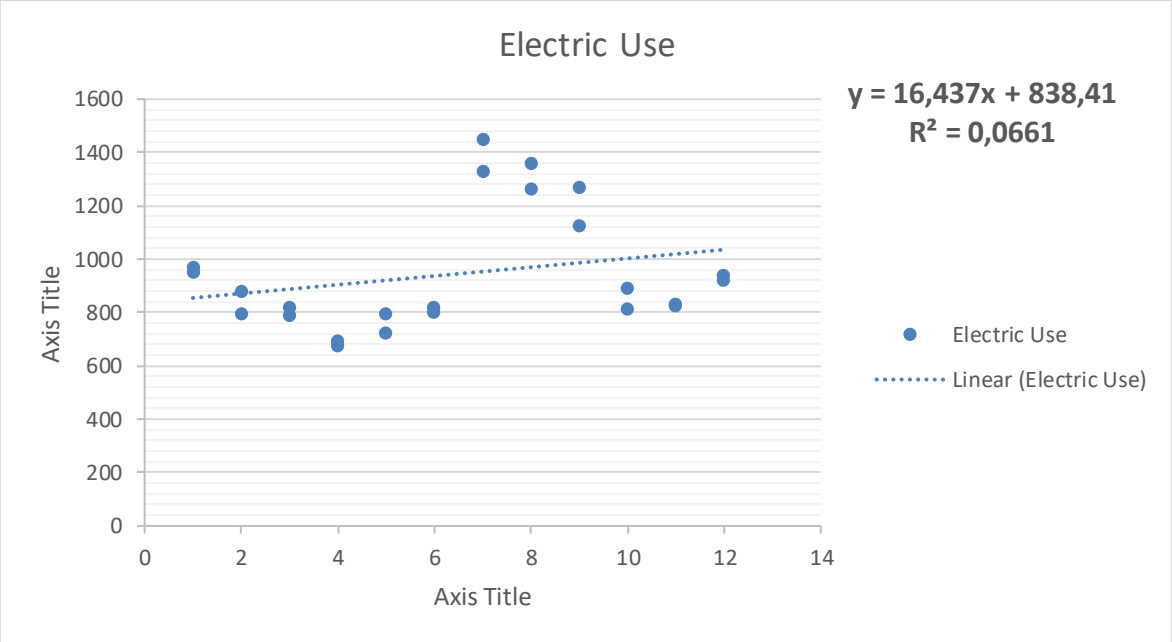
	df	*SS*	*MS*	*F*	*ignificance F*
Regression	1	77270,63	77270,63	1,556991	0,225225
Residual	22	1091820	49628,18		
Total	23	1169091			

	Coefficients	*andard Err*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*ower 95,0%*	*pper 95,0%*
Intercept	**838,409091**	96,94986	8,647863	1,58E-08	637,3474	1039,471	637,3474	1039,471
Month_num	**16,4370629**	13,17289	1,247795	0,225225	-10,8818	43,75597	-10,8818	43,75597

After having all necessary values, we start to predict based on this equation

Electric use = 838.4 – 16.44*Month_num

We have a graph showing "Electric Use"



2.2.2. USING PYTHON

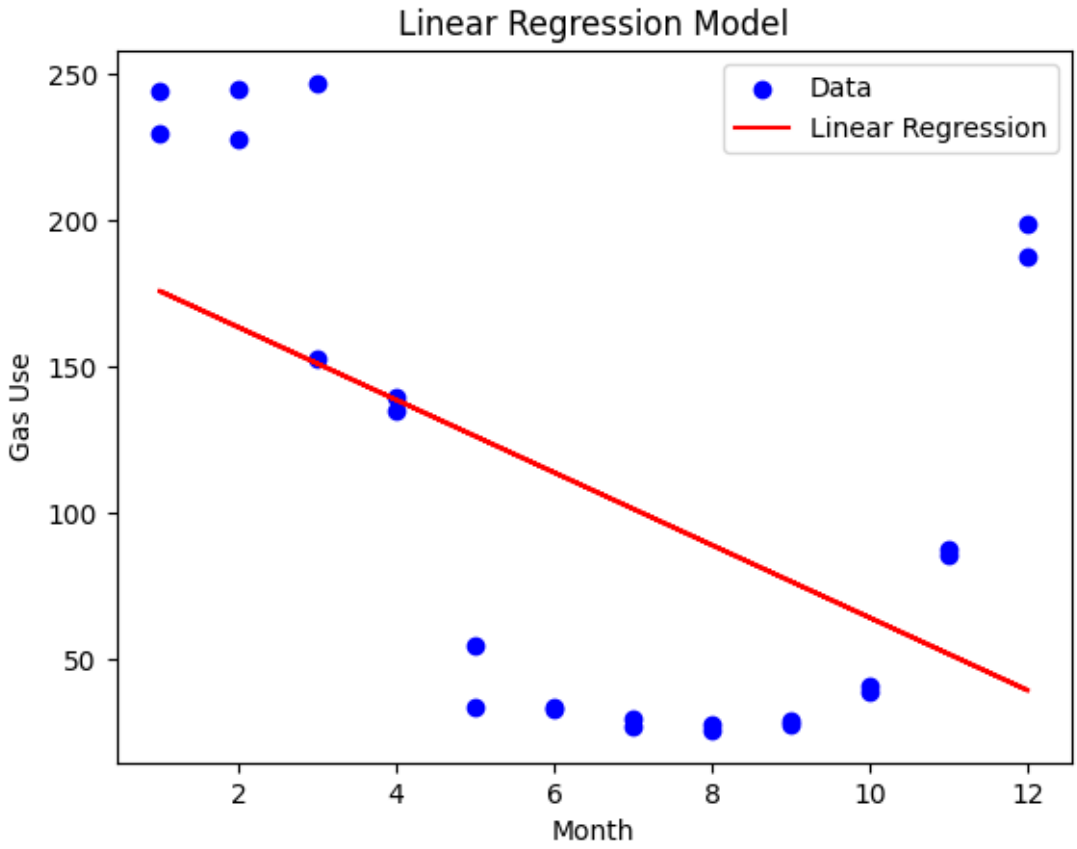
2.2.2.1. GAS

Using Linear Regression() to gain the model follow X and Y variables and taking result

```
[ ]  print("Intercept term: ",model.intercept_)
     print("R_coef: ", model.coef_)
     print("R Square: ", model.score(x, y))
```

```
Intercept term:  [188.39393939]
R_coef:  [[-12.40034965]]
R Square:  0.25530222499133703
```

We have a graph showing "Gas Use"



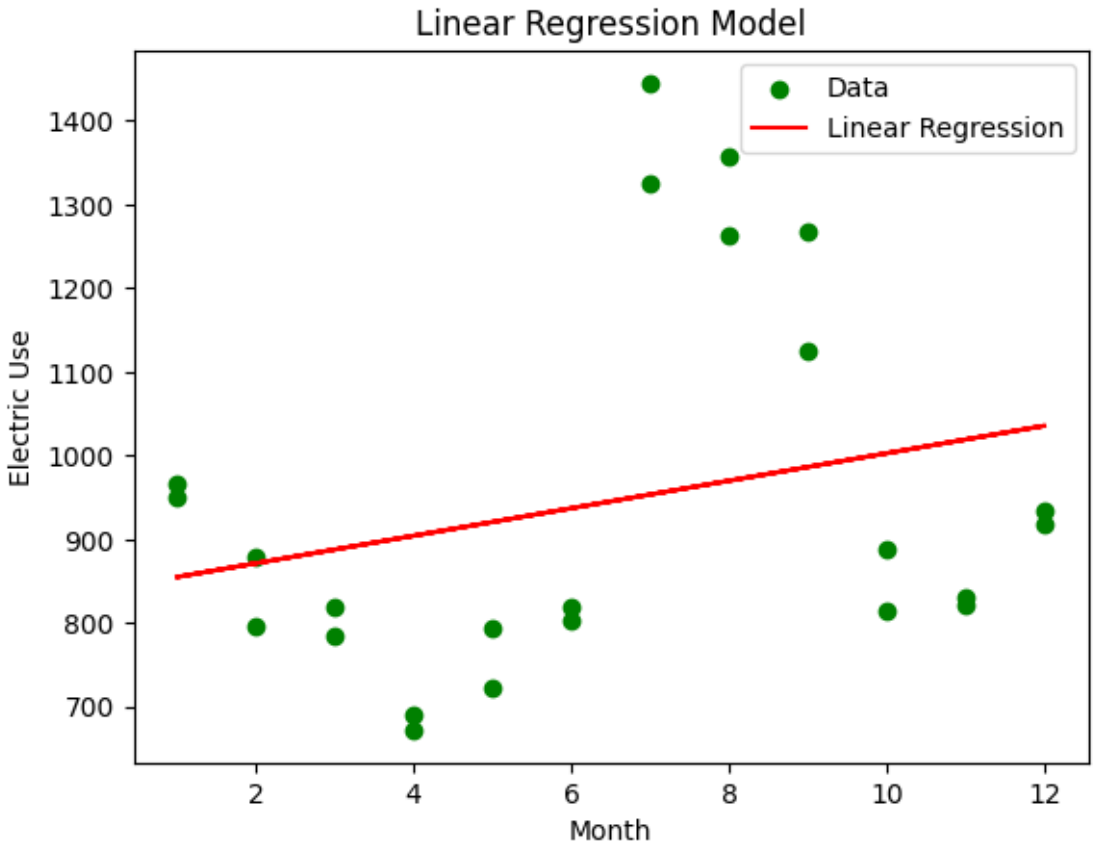
2.2.2.2. ELECTRIC

Using Linear Regression() to gain the model follow X and Y variables and taking result

```
print("Intercept term: ",model.intercept_)
print("R_coef: ", model.coef_)
print("R Square: ", model.score(x, y))
```

```
Intercept term:  [838.40909091]
R_coef:  [[16.43706294]]
R Square:  0.06609465466286213
```

We have a graph showing "Electric Use"



Chapter 3. VIETNAM GOLD PRICE

4.2. Using MS Excel and Python language to implement the ARIMA algorithm and another machine learning algorithm on the optional dataset about/ of Vietnam

Data sources: VIETNAM GOLD PRICE

We will be interested in 2 columns:

Date: observation time.

Price: Vietnam gold price (VND)

Statement of the problem: Based on **VIETNAMESE GOLD PRICE** data with dummies option group theory, using MS Excel and Python language to analyze and forecast ARIMA and Linear Regression

3.1. ARIMA

3.1.1. USING EXCEL

We have ARIMA (5, 2, 0) model. Means that:

$$Y_t = \mu + \varphi_1 Y_{t-1} + \varphi_2 Y_{t-2} + \varphi_3 Y_{t-3} + \varphi_4 Y_{t-4} + \varphi_5 Y_{t-5}$$

Calculate the lag to find the AR(q) and delete all rows which have empty value

Date	Price	LAG1	LAG 2	LAG3	LAG4	LAG5
03/01/2017	3638000					
04/01/2017	3645000	3638000				
05/01/2017	3665000	3645000	3638000			
06/01/2017	3650000	3665000	3645000	3638000		
07/01/2017	3660000	3650000	3665000	3645000	3638000	
09/01/2017	3650000	3660000	3650000	3665000	3645000	3638000
10/01/2017	3655000	3650000	3660000	3650000	3665000	3645000
11/01/2017	3658000	3655000	3650000	3660000	3650000	3665000
12/01/2017	3675000	3658000	3655000	3650000	3660000	3650000
13/01/2017	3668000	3675000	3658000	3655000	3650000	3660000
14/01/2017	3661000	3668000	3675000	3658000	3655000	3650000
16/01/2017	3672000	3661000	3668000	3675000	3658000	3655000
17/01/2017	3665000	3672000	3661000	3668000	3675000	3658000
18/01/2017	3670000	3665000	3672000	3661000	3668000	3675000

Date	Price	LAG1	LAG 2	LAG3	LAG4	LAG5
09/01/2017	3650000	3660000	3650000	3665000	3645000	3638000
10/01/2017	3655000	3650000	3660000	3650000	3665000	3645000
11/01/2017	3658000	3655000	3650000	3660000	3650000	3665000
12/01/2017	3675000	3658000	3655000	3650000	3660000	3650000
13/01/2017	3668000	3675000	3658000	3655000	3650000	3660000
14/01/2017	3661000	3668000	3675000	3658000	3655000	3650000
16/01/2017	3672000	3661000	3668000	3675000	3658000	3655000
17/01/2017	3665000	3672000	3661000	3668000	3675000	3658000
18/01/2017	3670000	3665000	3672000	3661000	3668000	3675000
19/01/2017	3658000	3670000	3665000	3672000	3661000	3668000
20/01/2017	3654000	3658000	3670000	3665000	3672000	3661000
21/01/2017	3665000	3654000	3658000	3670000	3665000	3672000
23/01/2017	3675000	3665000	3654000	3658000	3670000	3665000
24/01/2017	3686000	3675000	3665000	3654000	3658000	3670000

We have a coefficient table using Excel tool

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0,999465945
R Square	0,998932175
Adjusted R Square	0,998928568
Standard Error	29151,13628
Observations	1486

ANOVA

	df	*SS*	*MS*	*F*	*ignificance F*
Regression	5	1,18E+15	2,35E+14	276903,039	0
Residual	1480	1,26E+12	8,5E+08		
Total	1485	1,18E+15			

	Coefficients	*andard Erro*	*t Stat*	***P-value***	*Lower 95%*	*Upper 95%*	*ower 95,0%*	*Upper 95,0%*
Intercept	**-1183,816726**	3847,153	-0,30771	**0,75834447**	-8730,27	6362,636	-8730,27	6362,636
LAG1	**0,992521776**	0,025956	38,23878	**4,406E-223**	0,941608	1,043436	0,941608	1,043436
LAG 2	**0,002026048**	0,036569	0,055403	**0,95582492**	-0,06971	0,073759	-0,06971	0,073759
LAG3	**-0,098708239**	0,036476	-2,70609	**0,00688616**	-0,17026	-0,02716	-0,17026	-0,02716
LAG4	**0,050455595**	0,03657	1,379693	**0,16788962**	-0,02128	0,12219	-0,02128	0,12219
LAG5	**0,054414533**	0,025983	2,094217	**0,03641038**	0,003447	0,105382	0,003447	0,105382

After having all necessary values, we start to predict based on this equation

$$Y_t = -1183{,}82 + 0.99 * Y_{t-1} + 0.002 * Y_{t-2} - 0.099 * Y_{t-3} + 0.05 * Y_{t-4} + 0.054 * Y_{t-5}$$

We have a graph showing "Gold Price"

Date	Price	MA
03/01/2017	3638000	#N/A
04/01/2017	3645000	#N/A
05/01/2017	3665000	#N/A
06/01/2017	3650000	#N/A
07/01/2017	3660000	3651600
09/01/2017	3650000	3654000
10/01/2017	3655000	3656000
11/01/2017	3658000	3654600
12/01/2017	3675000	3659600
13/01/2017	3668000	3661200
14/01/2017	3661000	3663400
16/01/2017	3672000	3666800
17/01/2017	3665000	3668200
18/01/2017	3670000	3667200
19/01/2017	3658000	3665200
20/01/2017	3654000	3663800
21/01/2017	3665000	3662400
23/01/2017	3675000	3664400
24/01/2017	3686000	3667600
25/01/2017	3690000	3674000
02/02/2017	3780000	3699200
03/02/2017	3760000	3718200
04/02/2017	3760000	3735200
06/02/2017	3700000	3738000

3.1.2. USING PYTHON

First, we find the best ARIMA model

```
# Training process
x_train = np.array(train_data.index).reshape(-1, 1)
y_train = np.array(train_data['Price'])

# Find the best ARIMA model using auto_arima
from pmdarima.arima import auto_arima
model = auto_arima(y_train, trace=True, error_action='ignore', suppress_warnings=True)

# Fit the model
model.fit(y_train)
```

```
Performing stepwise search to minimize aic
 ARIMA(2,2,2)(0,0,0)[0]             : AIC=inf, Time=3.07 sec
 ARIMA(0,2,0)(0,0,0)[0]             : AIC=28767.921, Time=0.08 sec
 ARIMA(1,2,0)(0,0,0)[0]             : AIC=28762.891, Time=0.23 sec
 ARIMA(0,2,1)(0,0,0)[0]             : AIC=28807.910, Time=0.42 sec
 ARIMA(2,2,0)(0,0,0)[0]             : AIC=28761.771, Time=0.46 sec
 ARIMA(3,2,0)(0,0,0)[0]             : AIC=28737.347, Time=1.62 sec
 ARIMA(4,2,0)(0,0,0)[0]             : AIC=28735.362, Time=2.19 sec
 ARIMA(5,2,0)(0,0,0)[0]             : AIC=28721.782, Time=2.23 sec
 ARIMA(5,2,1)(0,0,0)[0]             : AIC=inf, Time=3.66 sec
 ARIMA(4,2,1)(0,0,0)[0]             : AIC=inf, Time=2.59 sec
 ARIMA(5,2,0)(0,0,0)[0] intercept   : AIC=28723.819, Time=1.49 sec

Best model:  ARIMA(5,2,0)(0,0,0)[0]
Total fit time: 18.102 seconds
```

ARIMA
ARIMA(5,2,0)(0,0,0)[0]

We found the best model is ARMIA(2,0,0). After that, we calculate the model parameters

```
model.summary()
```

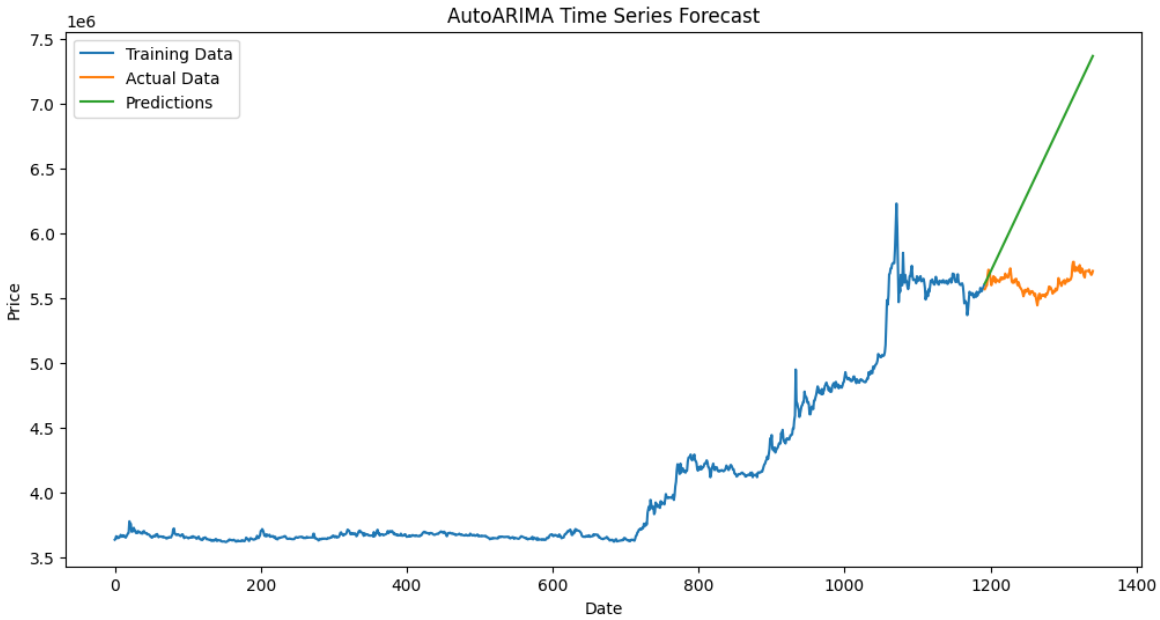
SARIMAX Results

Dep. Variable:	y	**No. Observations:**	1192
Model:	SARIMAX(5, 2, 0)	**Log Likelihood**	-14354.891
Date:	Sun, 28 Apr 2024	**AIC**	28721.782
Time:	10:18:54	**BIC**	28752.272
Sample:	0 - 1192	**HQIC**	28733.272
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.2290	0.003	-87.394	0.000	-0.234	-0.224
ar.L2	-0.1899	0.003	-70.058	0.000	-0.195	-0.185
ar.L3	-0.1590	0.002	-66.912	0.000	-0.164	-0.154
ar.L4	-0.0996	0.002	-46.274	0.000	-0.104	-0.095
ar.L5	-0.0483	0.002	-27.768	0.000	-0.052	-0.045
sigma2	1.106e+09	1.03e-12	1.07e+21	0.000	1.11e+09	1.11e+09

Ljung-Box (L1) (Q):	141.87	**Jarque-Bera (JB):**	114327.46
Prob(Q):	0.00	**Prob(JB):**	0.00
Heteroskedasticity (H):	6.25	**Skew:**	-2.43
Prob(H) (two-sided):	0.00	**Kurtosis:**	50.77

We draw a chart



3.2. LINEAR REGRESSION

3.2.1. USING EXCEL

We have a coefficient table using Excel tool

Date	Date_Num	Price
03/01/2017	42738,00	3638000
04/01/2017	42739,00	3645000
05/01/2017	42740,00	3665000
06/01/2017	42741,00	3650000
07/01/2017	42742,00	3660000
09/01/2017	42744,00	3650000
10/01/2017	42745,00	3655000
11/01/2017	42746,00	3658000
12/01/2017	42747,00	3675000
13/01/2017	42748,00	3668000
14/01/2017	42749,00	3661000
16/01/2017	42751,00	3672000
17/01/2017	42752,00	3665000
18/01/2017	42753,00	3670000
19/01/2017	42754,00	3658000
20/01/2017	42755,00	3654000
21/01/2017	42756,00	3665000

SUMMARY OUTPUT

Regression Statistics	
Multiple R	0,909977741
R Square	0,828059489
Adjusted R Squar	0,827944015
Standard Error	369258,1577
Observations	1491

ANOVA

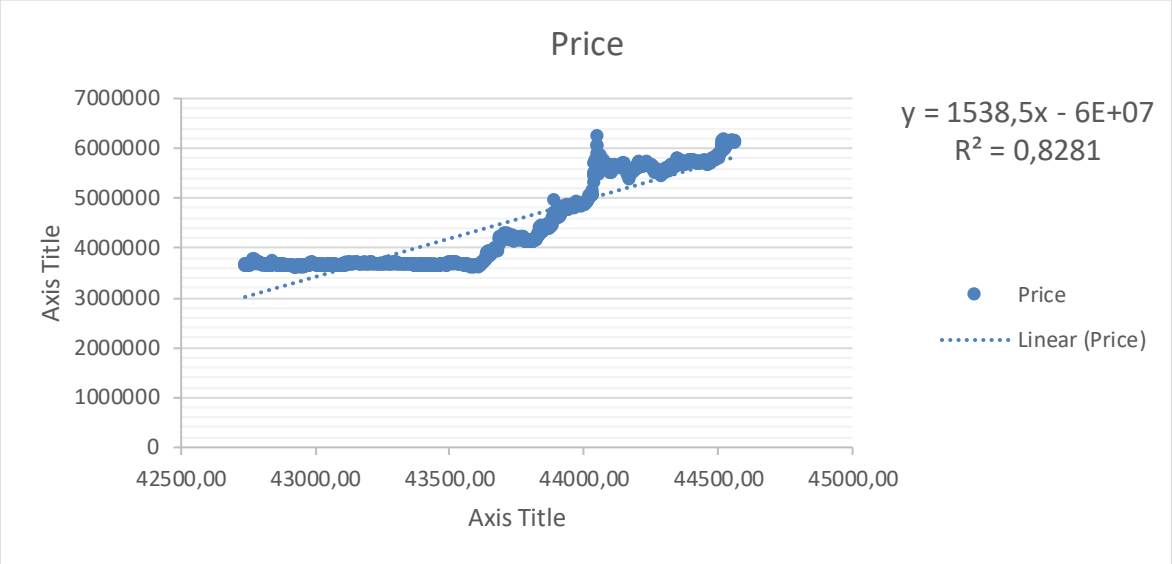
	df	*SS*	*MS*	*F*	*ignificance F*
Regression	1	9,78E+14	9,78E+14	7170,972	0
Residual	1489	2,03E+14	1,36E+11		
Total	1490	1,18E+15			

	Coefficients	*andard Erro*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*ower 95,0%*	*Upper 95,0%*
Intercept	**-62731809,69**	793103,2	-79,0967	0	-6,4E+07	-6,1E+07	-6,4E+07	-61176091,37
Date_Num	**1538,454556**	18,16752	84,68159	0	1502,818	1574,091	1502,818	1574,09121

After having all necessary values, we start to predict based on this equation

Price = -62732809.69 – 1538.45*Date_num

We have a graph showing "Price Gold"



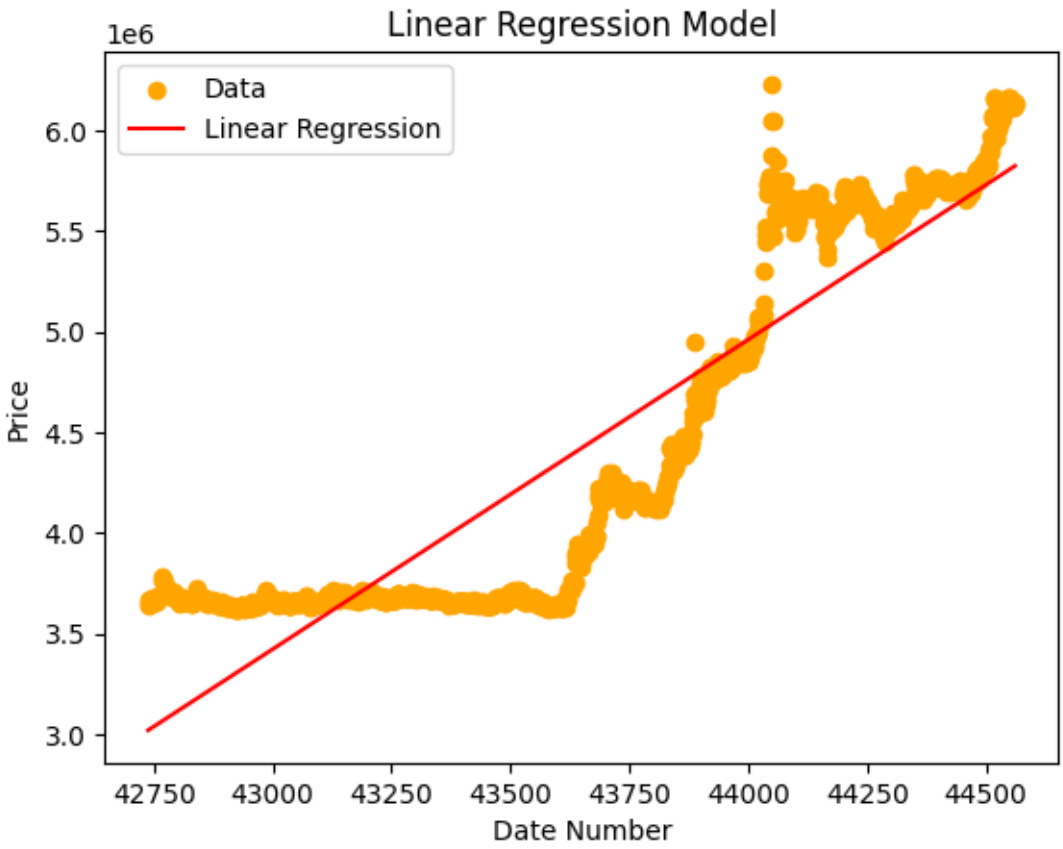
3.2.2. USING PYTHON

Using Linear Regression() to gain the model follow X and Y variables and taking result

```
[ ]  print("Intercept term: ",model.intercept_)
     print("R_coef: ", model.coef_)
     print("R Square: ", model.score(x, y))
```

```
Intercept term:  [-62731809.69288268]
R_coef:  [[1538.45455619]]
R Square:  0.8280594886866178
```

We have a graph showing “Price Gold”



Chapter 4. EVALUATE THE ACCURACY

4.3. Methods to evaluate the accuracy of an algorithm or prediction model? Apply the above methods to labs 2, 3, 4.1, 4.2

4.1. DEFINITIONS

4.1.1. MEAN ABSOLUTE ERROR (MAE)

Absolute Error is the amount of error in your measurements. It is the difference between the measured value and "true" value

This can be caused by your scale not measuring the exact amount you are trying to measure

The formula for the absolute error (Δx) is:

$$(\Delta x) = xi - x$$

Where:

xi is the measurement,

x is the true value.

Mean Absolute Error (MAE) is a statistical measure of errors between paired observations expressing the same phenomenon. It is calculated by taking the average of the absolute differences between the predicted and actual values

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|x_i - x|$$

Where:

n = the number of errors,

|xi – x| = the absolute errors.

4.1.2. MEAN ABSOLUTE PERCENT AGE ERROR (MAPE)

The mean absolute percentage error (MAPE) — also called the mean absolute percentage deviation (MAPD) — measures accuracy of a forecast system. It measures this accuracy as a percentage, and can be calculated as the average

absolute percent error for each time period minus actual values divided by actual values.

The mean absolute percentage error (MAPE) is the most common measure used to forecast error, probably because the variable's units are scaled to percentage units, which makes it easier to understand [3]. It works best if there are no extremes to the data (and no zeros). It is often used as a loss function in regression analysis and model evaluation.

The formula for MAPE is:

$$MAPE = \frac{1}{n}\left|\frac{A_t - F_t}{A_t}\right|$$

Where:

n is the number of fitted points

At is the actual value

Ft is the forecast value

MAE is a scale-dependent accuracy measure and therefore cannot be used to make comparisons between predicted values that use different scales . It is a common measure of forecast error in time series analysis.

4.1.3. MEAN SQUARED ERROR (MSE)

Mean Squared Error (MSE) is a common metric used to measure the average squared difference between the actual (observed) values and the predicted values in a regression analysis. It is a way to quantify the accuracy of a model's predictions

The formula for Mean Squared Error is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where:

n is the number of data points

yi is the actual (observed) value of the dependent variable for the i-th data point

y^i is the predicted value of the dependent variable for the i-th data point

4.1.4. ROOT MEAN SQUARED ERROR (RMSE)

RMSE stands for Root Mean Squared Error, and it is another metric commonly used to evaluate the performance of a regression model, similar to Mean Squared Error (MSE). The key difference is that RMSE takes the square root of the average of the squared differences between the predicted and actual values. This is done to make the metric more interpretable in the same units as the target variable

The formula for RMSE is as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Where:

n is the number of data points

yi is the actual (observed) value of the dependent variable for the i-th data point

y^i is the predicted value of the dependent variable for the i-th data point

4.1.5. COEFFICENT OF DETERMINATION (R-SQUARED)

The coefficient of determination, or R 2 is a measure that provides information about the goodness of fit of a model. In the context of regression, it is a statistical measure of how well the regression line approximates the actual data. It is

therefore important when a statistical model is used either to predict future outcomes or in the testing of hypotheses.

$$R^2 = 1 - \frac{SSE}{SST} = \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y}_i)^2}$$

4.1.6. ADJUST R-SQUARED

The R-Squared (R2) value is commonly reported when performing multiple linear regression. It quantifies the proportion of variance of the dependent variable that can be accounted for by the regression model in the sample, which is commonly abbreviated as the proportion of variance explained. It is well known that the R2 value systematically overestimates the amount of variance explained in the population, which is arguably the more relevant quantity. To estimate the amount of variance explained in the population, the so-called adjusted R^2 is typically used

$$Adjusted\ R^2 = R_a^2 = 1 - \frac{SSE(n-2)}{SST(v)}$$

Where:

v= n−m

v is the residual degrees of freedom which indicate the number of independent pieces of information involving the n data points thatare required to calculate the sum of squares

n is the number of response values estimated from the response values

m is the number of fitted coefficients estimated from the response values

4.2. USING PYTHON

4.2.1. LAB 2

```
# Get the R-squared
r_squared = results.rsquared
print(f'R-squared: {r_squared}')
```

```
R-squared: 0.8377310230323395
```

```
# Get the Mean Squared Error
mse = results.mse_resid
print(f'Mean Squared Error: {mse}')
```

```
Mean Squared Error: 1.5889335938992273
```

❖ **Conclusion:**

R-squared, also known as the coefficient of determination, is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model. If the R-squared is 0.8377310230323395, it means that approximately 83.77% of the variance in the dependent variable can be explained by the model.

Mean Squared Error (MSE) is a measure of how well a regression line fits the data points. A smaller MSE means a closer fit to the data. In this case, the MSE is 1.5889335938992273, which is relatively low, indicating a good fit.

4.2.2. LAB 3

```
# Creating a regression table using MLR Multiple Linear Regression through OLS Ordinary Least Squares.
x = sm.add_constant(x)

reg = sm.OLS(y, x).fit()
print(reg.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.534
Model:                            OLS   Adj. R-squared:                  0.492
Method:                 Least Squares   F-statistic:                     12.63
Date:                Sun, 28 Apr 2024   Prob (F-statistic):           6.33e-07
Time:                        22:54:41   Log-Likelihood:                -148.69
No. Observations:                  49   AIC:                             307.4
Df Residuals:                      44   BIC:                             316.8
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         17.9210     24.557      0.730      0.469     -31.571      67.413
x1             0.0720      0.018      4.004      0.000       0.036       0.108
x2            -0.2486      0.083     -2.990      0.005      -0.416      -0.081
x3            -0.1356      0.066     -2.057      0.046      -0.269      -0.003
x4            -0.1628      0.079     -2.051      0.046      -0.323      -0.003
==============================================================================
Omnibus:                        1.954   Durbin-Watson:                   2.010
Prob(Omnibus):                  0.376   Jarque-Bera (JB):                1.833
Skew:                          -0.450   Prob(JB):                        0.400
Kurtosis:                       2.706   Cond. No.                     4.11e+04
==============================================================================
```

Based on the results of the regression analysis, the p-values for the independent variables are all less than 0.05, indicating that they are statistically significant in predicting the graduation rate. The R-squared value of 0.534 indicates that the model explains 53.4% of the variance in the graduation rate. The regression equation is as follows:

%graduation = 17.921 + 0.072*median_sat – 24.8592*accept – 0.0001*expend – 0.1628*top 10% HS

```
# Perform ANOVA to test the significance of the regression model
from scipy.stats import f_oneway
f_value, p_value = f_oneway(data['Graduation %'], data['Median SAT'], data['Acceptance Rate'], data['Expenditures/Student'], data['Top 10% HS'])
print(f_value, p_value)
```

```
15587.9846115108 1.3312865590772317e-288
```

```
# Perform Chi-Square test to test the independence of the variables
from scipy.stats import chi2_contingency
chi2, p, dof, expected = chi2_contingency(pd.crosstab(data['Graduation %'], data['Median SAT']))
print(chi2, p)
```

```
793.9361111111112 0.3287297687139225
```

```
# Calculate the Mean Squared Error (MSE) between the actual and predicted values
y_pred = reg.predict(x)
mse = mean_squared_error(y, y_pred)
print(mse)
```

```
25.30310118943153
```

```
# Calculate the Fisher's F-Test
f_test = mse / (np.mean(y) * (1 - np.mean(y)))
print(f_test)
```

```
-0.0036957889350236733
```

```
# Calculate the Root Mean Squared Error (RMSE), MAPE
rmse = np.sqrt(mse)
print(rmse)

mape = mean_absolute_percentage_error(y, y_pred)
print(mape)
```

```
5.030218801347664
0.04954012750825252
```

❖ **Conclusion:**

The regression analysis indicates that the median SAT score, acceptance rate, expenditure per student, and the percentage of students from the top 10% of their high school class are all statistically significant predictors of the graduation rate. The model explains 53.4% of the variance in the graduation rate, and the regression equation can be used to predict the graduation rate based on these variables.

4.2.3. LAB 4.1

```
# Creating a regression table using MLR Multiple Linear Regression through OLS Ordinary Least Squares.
x = sm.add_constant(x)

reg_electric = sm.OLS(y_electric, x).fit()
print(reg_electric.summary())
```

[34]

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.066
Model:                            OLS   Adj. R-squared:                  0.024
Method:                 Least Squares   F-statistic:                     1.557
Date:                Sun, 28 Apr 2024   Prob (F-statistic):              0.225
Time:                        23:33:16   Log-Likelihood:                -162.76
No. Observations:                  24   AIC:                             329.5
Df Residuals:                      22   BIC:                             331.9
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        838.4091     96.950      8.648      0.000     637.347    1039.471
x1            16.4371     13.173      1.248      0.225     -10.882      43.756
==============================================================================
Omnibus:                        4.265   Durbin-Watson:                   1.029
Prob(Omnibus):                  0.119   Jarque-Bera (JB):                3.688
Skew:                           0.933   Prob(JB):                        0.158
Kurtosis:                       2.544   Cond. No.                         15.9
==============================================================================
```

R-squared (R^2) indicates that the model can only explain about 6.6% of the variation in the dependent variable, meaning the model cannot explain a large part of this variation.

The p-value (P>|t|) for the independent variable x1 is 0.225.

In conclusion, the model does not have good performance, with low R-squared and high p-value, implying that the model cannot explain the relationship between the dependent variable and the independent variable significantly

```
# ANOVA, Chi-Square, MSE, Fisher's F-Test, RMSE, MAPE
# Perform ANOVA
from scipy.stats import f_oneway
f_value1, p_value1 = f_oneway(data['Gas Use '], data['Month'])
print(f_value1, p_value1)

f_value2, p_value2 = f_oneway(data['Electric Use'], data['Month'])
print(f_value2, p_value2)
```

```
32.82364107697193 7.35985613614859e-07
415.9916523891151 1.113847916383789e-24
```

The low p-value suggests a significant difference in electric usage across months. Analyses indicate significant differences in gas and electric usage across months.

```
# Perform Chi-Square
from scipy.stats import chi2_contingency
chi2, p, dof, ex = chi2_contingency(data[['Gas Use ', 'Month']])
print(chi2, p)

chi2, p, dof, ex = chi2_contingency(data[['Electric Use', 'Month']])
print(chi2, p)
```

```
242.19477265153503 1.7558480699313437e-38
44.305736282625496 0.00482949180267721
```

The high p-value suggests no significant association between electric usage and the month.

In summary, the Chi-Square test shows a significant association between gas usage and the month, but not between electric usage and the month

```
# Calculate the Mean Squared Error (MSE) between the actual and predicted values
from sklearn.metrics import mean_squared_error
y_pred_gas = reg_gas.predict(x)
mse_gas = mean_squared_error(y_gas, y_pred_gas)
print(mse_gas)

y_pred_electric = reg_electric.predict(x)
mse_electric = mean_squared_error(y_electric, y_pred_electric)
print(mse_electric)
```

```
5345.004929098679
45492.49446386946
```

The MSE represents the average squared difference between the actual and predicted values.

A lower MSE indicates better accuracy of the model in predicting the target variable.

In this case, the MSE for gas use is considerably lower than for electric use, suggesting better predictive performance for gas usage

```
# Perform Fisher's F-Test
from scipy.stats import f

f_value1 = mse_gas / mse_electric
f_value2 = mse_electric / mse_gas

p_value1 = f.cdf(f_value1, len(y_gas)-1, len(y_electric)-1)
p_value2 = f.cdf(f_value2, len(y_electric)-1, len(y_gas)-1)

print(f_value1, p_value1)
print(f_value2, p_value2)
```

```
0.11749201691597125 1.3398594589953418e-06
8.511216559633892 0.999998660140541
```

The high p-value suggests no significant difference in the variances of electric use and gas use.

In summary, Fisher's F-Test confirms that there is a significant difference in the variances between gas use and electric use, with gas use showing lower variance

```
# Calculate the Root Mean Squared Error (RMSE) between the actual and predicted values
rmse_gas = np.sqrt(mse_gas)
print(rmse_gas)

rmse_electric = np.sqrt(mse_electric)
print(rmse_electric)

# Calculate the Mean Absolute Percentage Error (MAPE) between the actual and predicted values
mape_gas = np.mean(np.abs((y_gas - y_pred_gas) / y_gas)) * 100
print(mape_gas)

mape_electric = np.mean(np.abs((y_electric - y_pred_electric) / y_electric)) * 100
print(mape_electric)
```

```
73.10954061611028
213.2896961033736
113.16321683409056
18.68214897758175
```

RMSE measures the square root of the average of the squared differences between the actual and predicted values. A lower RMSE indicates better accuracy of the model in predicting the target variable. In this case, the RMSE for gas use is considerably lower than for electric use, suggesting better predictive performance for gas usage.

MAPE measures the average percentage difference between the actual and predicted values relative to the actual values. A lower MAPE indicates better accuracy of the model. Interestingly, the MAPE for electric use is significantly lower than for gas use, suggesting better predictive performance for electric usage in terms of percentage error

4.2.4. LAB 4.2

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the linear regression model
model = LinearRegression()

# Train the model
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Print the coefficients and intercept
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

```
Mean Squared Error (MSE): 134360806126.19327
Coefficients: [  1836.23286785  44065.88112575 560265.05142754]
Intercept: -1127074577.3574367
```

```python
#  MSE, Fisher's F-Test, RMSE, MAPE
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score

# Calculate the Mean Squared Error (MSE)
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error (MSE):", mse)

# Calculate the Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, y_pred)
print("Mean Absolute Error (MAE):", mae)

# Calculate the Root Mean Squared Error (RMSE)
rmse = np.sqrt(mse)
print("Root Mean Squared Error (RMSE):", rmse)

# Calculate the Mean Absolute Percentage Error (MAPE)
mape = np.mean(np.abs((y_test - y_pred) / y_test)) * 100
print("Mean Absolute Percentage Error (MAPE):", mape)

# Calculate the R-squared value
r2 = r2_score(y_test, y_pred)
print("R-Squared:", r2)
```

```
Mean Squared Error (MSE): 134360806126.19327
Mean Absolute Error (MAE): 311535.3047196028
Root Mean Squared Error (RMSE): 366552.5966709188
Mean Absolute Percentage Error (MAPE): 7.3571371690090235
R-Squared: 0.8294997582717581
```

❖ **Conclusion:**

Based on the OLS regression results provided:

The model has an R-squared value of 0.827, indicating that approximately 82.7% of the variance in the dependent variable (Price) is explained by the independent variables (Day, Month, Year) in the model.

The coefficients of the independent variables (Day, Month, Year) indicate their impact on the dependent variable (Price). For example, for each

unit increase in Day, the Price is estimated to increase by 1836.23 units, holding other variables constant.

The p-values associated with the coefficients indicate the statistical significance of each independent variable. In this case, all three independent variables have p-values less than 0.05, suggesting that they are statistically significant predictors of Price.

The Mean Squared Error (MSE) is 134360806126.19327, the Mean Absolute Error (MAE) is 311535.3047196028, the Root Mean Squared Error (RMSE) is 366552.5966709188, and the Mean Absolute Percentage Error (MAPE) is 7.36%.

The overall model fit is assessed using various diagnostic measures, such as the Omnibus, Durbin-Watson, and Jarque-Bera tests, which provide insights into the normality, autocorrelation, and homoscedasticity assumptions of the regression model.

Based on these results, it appears that the model has a good fit and the independent variables are significant predictors of the dependent variable.

REFERENCES

[1] https://otexts.com/fpp2/wn.html

[2] N.M.Nhut - “LAB04_HuongDan”

[3] N.M.Nhut - “Lecture-1-Giới-thiệu-mô-hình-dữ-liệu-chuỗi-thời-gian”

[4] “Forecasting ethanol demand in India to meet future blending targets: A comparison of ARIMA and various regression models,” Energy Rep., vol. 9, pp. 411– 418, Mar. 2023, doi: 10.1016/j.egyr.2022.11.038

[5] MAPE - Mean Absolute Percentage Error. Working with Planning. Retrieved May 27, 2022

[6] Mean Squared Error (MSE), By Jim Frost