

Project report

Manufacturing Data Collection and Analytics

National Research University

Higher School of Economics

Option II



Performed by students

Ajay Mishra

Shipon Nath

Koushik

Project supervisor: KOVALEV ILYA ALEXANDROVICH

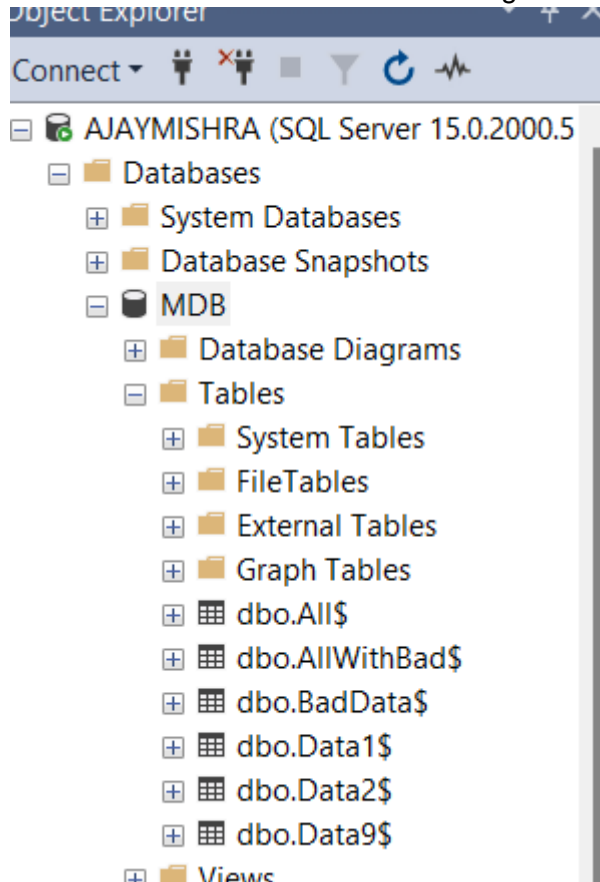
Table OF Content:

Task: -	Done By: -
Create DB- Load All Data- Query 1, 2-	Ajay Mishra
Load xml file in Colab- Visualize the Data- Analysis the data-	Shipon nath
Run the xml file with Python-	Kawshiqe Ahamed

Creation of DB-

-Created a data base including bad and good data, uploaded a file in Microsoft SQL, Management Server DB name is MDB

Created Database with bad file and 3 good files



Data Manipulation to understand about bad data -

- Took the 3 good files and find out how many location data are matching with correct location in bad file and found 308 locations are correct in bad data
- select Count (Distinct BadData\$.Location) as Count bad Data location from
BadData\$
left join All\$ ON BadData\$.customer =All\$.customer and BadData\$.location
=All\$.location

where BadData\$.Location=All\$.location

SQLQuery1.sql - A...YMISHRA\User (56))* ✕

```
select Count (Distinct BadData$.Location) as CountbadDataLocation from BadData$  
left join All$ ON BadData$.customer =All$.customer and BadData$.location =All$.location  
  
where BadData$.Location=All$.location
```

100 %

Results Messages

	CountbadDataLocation
1	308

SQLQuery1.sql - A...YMISHRA\User (56))* ✕

```
select BadData$.Location as LocationNames from BadData$  
left join All$ ON BadData$.customer =All$.customer and BadData$.location =All$.location  
  
where BadData$.Location=All$.location
```

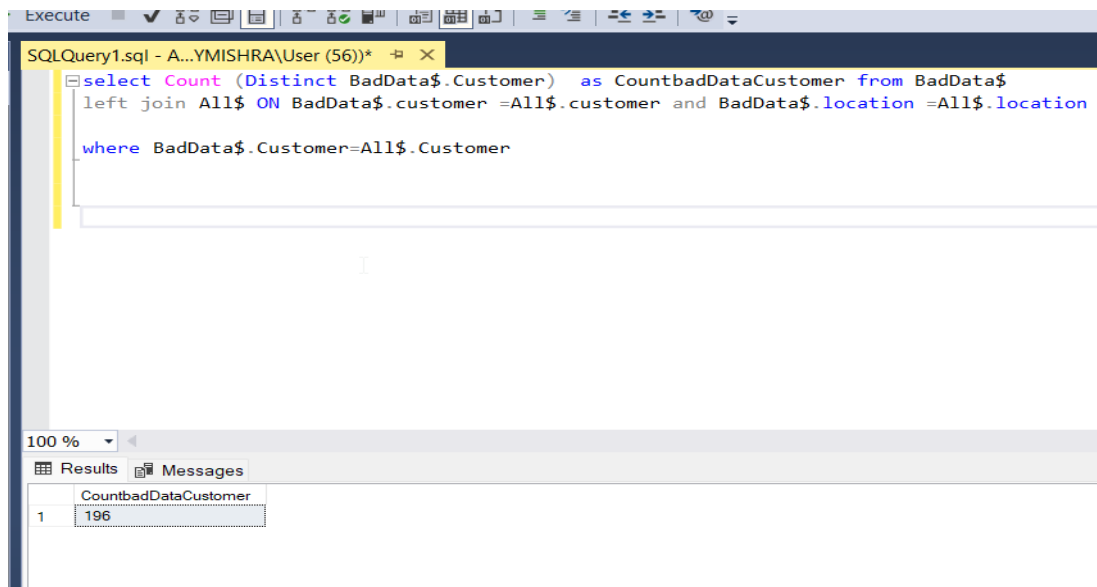
100 %

Results Messages

	LocationNames
1	Bosch#14
2	Bosch#14
3	Bosch#14
4	Bosch#14
5	Bosch#17
6	Bosch#17
7	Bosch#19
8	Bosch#19
9	Bosch#19
10	Bosch#19
11	Bosch#20
12	Bosch#20
13	Bosch#20
14	Bosch#20

Query executed successfully.

196 data are correct in customer column in bad file while comparing with 3 good files



Answers of tasks with respect to SQL and their screenshot

Took 3 tables and based on 3 tables answers these task

Questions1 - Count the number of unique Locations

1-Count of Unique Locations - 1595

The screenshot shows a SQL Server Enterprise Manager window with a results pane. The results pane shows a single row with the column name 'CountOfuniqueLocations' and the value '1595'.

CountOfuniqueLocations
1595

2-List Of Unique customer_- Count- 14134

SQLQuery1.sql - A...YMISHRA\User (56))

100 %

Results Messages

	uniqueCustomer
1	PLCCNC23
2	CustomerNewGenloT_6
3	PLCIloT.18
4	Iot_XDK Bosch:21
5	PLCCNC-23
6	u_Fanuc_15
7	CustomerGPD151_21
8	PLC_Bosch_1
9	Iot_FestoR267/1
10	Iot_GPD151.10
11	machineSiemens/12
12	Iot_Festo_9
13	PLCNewGenloT.11
14	machine-15
15	CustomerNewGenloT.4
16	u_SensXiom:3
17	PLC_FanucloT.14
18	Iot_SensXiom_19
19	CustomerIoTSiemens...
20	Iot_3
21	machine15
22	u_IloT_17
23	Iot_Festo_8
24	PLCFanucloT20
25	PLCFanucloT-20
26	CustomerSensXiom_20
27	u_NewGenloT-18
28	u_NewGenloT#14
29	u_NewGenloT18
30	PLCSensXiom_22
31	Iot_IoTSiemens/16

Query executed successfully.

3- Find all Customers with GyrSen in the range between 22-24

SQLQuery1.sql - A...YMISHRA\User (56))

```
select customer,GyrSen from All$  
where (GyrSen between '22' and '24')
```

100 %

Results Messages

	customer	GyrSen
1	PLC_Bosch19	22
2	PLCNewGenloT20	23
3	u_FestoR267.3	24
4	PLCNewGenloT-1	22
5	PLCFanuc:7	23
6	Iot_VersaloT.2	22
7	u_Festo:15	24
8	u_IloT-5	23
9	u_NewGenloT_5	23
10	u_VersaloT.23	23
11	u_SensXiom:5	22
12	u_VersaloT_6	22
13	CustomerCISS	23
14	CustomerXDK-22	24
15	PLCVersaloT_19	23
16	u_OSISoftloT.10	24
17	CustomerOSISoft...	23
18	machinelot_12	24
19	Iot_FanucloT:14	24
20	PLC_NewGenloT...	24
21	CustomerSieme...	23
22	PLCFestoR267#7	23
23	CustomerSensXi...	24
24	u_FanucloT/2	23
25	Iot_Siemens_16	24
26	PLCOSISoftloT:14	23
27	PLCFestoR267-5	24
28	machineCNC:12	24

Query executed successfully.

AJAYMISHRA (15.0 RTM)

4-Find all ios app Customers that do not support wi-fi and update them to "yes"

```
SQLQuery1.sql - A...YMISHRA\User (56))*  
--  
UPDATE A11$  
SET [Wi-Fi] = 'yes'  
WHERE iOSapp='yes'
```

100 %
Messages
(20002 rows affected)
Completion time: 2022-12-09T12:40:12.9390749+03:00

```
SQLQuery1.sql - A...YMISHRA\User (56))*  
--  
SELECT customer, iOSapp  
FROM A11$  
WHERE [Wi-Fi] = 'yes'
```

100 %
Results Messages

	customer	iOSapp	Wi-Fi
1	machineFesto-17	yes	yes
2	PLC_Bosch19	yes	yes
3	u_NewGenIoT-3	yes	yes
4	lot_IoTSiemens#3	yes	yes
5	PLC_Siemens11	yes	yes
6	machineSensXiom#20	yes	yes
7	u_OSISoftIoT21	yes	yes
8	PLC_CNC20	yes	yes
9	lot_23	yes	yes
10	PLC_CNC18	yes	yes
11	machine10	yes	yes
12	CustomerVersaloT13	yes	yes
13	machine21	yes	yes
14	u_Bosch_16	yes	yes
15	lot_Festo6	yes	yes
16	PLC_XDK3	yes	yes
17	PLC_12	yes	yes
18	PLC_SensXiom-12	yes	yes
19	lot_OSISoftIoT23	yes	yes
20	CustomerIoT-19	yes	yes
21	CustomerFanucIoT-8	yes	yes
22	machineGPD1513	yes	yes
23	machineBosch12	yes	yes
24	u_OSISoftIoT21	yes	yes
25	CustomerIoT#9	yes	yes
26	lot_GPD1516	yes	yes
27	PLC_Fanuc#9	yes	yes
28	CustomerCNC6	yes	yes
29	PLCNewGenIoT-1	yes	yes

Query executed successfully. AJAYMISHRA (15.0 RTM) AJAYMISHRA\User (56) MDB 00:00:09 20,002 rows

5-Find all Customers with a Number of sensors in the range between 25-27, and for all devices with STM or ESP, regardless of digits further instead of Yes in the iOS App column display "1", and for No - "0" (Count of Customers- 14,442)

```
SQLQuery1.sql - A...YMISHRA\User (63))*  
--  
SELECT customer, [Number of sensors], Board, iOSapp  
FROM A11$  
WHERE ([Number of sensors] >= '25' and '27' <= [Number of sensors]) and (Board LIKE '%STM%' or Board LIKE '%ESP%')
```

100 %
Results Messages

	customer	Number of sensors	Board	iOSapp
1	PLC_Bosch19	27	ESP32	1
2	u_NewGenIoT-3	35	STM32	1
3	machineSensXiom#20	39	ESP32	1
4	PLC_CNC20	36	ESP8266	1
5	PLC_FanucIoT_22	40	ESP8266	0
6	CustomerIoT5	37	STM	0
7	machine10	35	STM32	1
8	machineSensXiom_4	43	ESP8266	0
9	u_NewGenIoT10	44	ESP8266	0
10	CustomerVersaloT13	33	ESP8266	1
11	u_Bosch_16	40	ESP8266	1
12	u_IoT1	37	ESP8266	0
13	lot_Festo6	43	ESP8266	1
14	PLC_XDK3	49	STM32	1

Query executed successfully. AJAYMISHRA (15.0 RTM) AJAYMISHRA\User (63) MDB 00:00:00 14,442 rows

Run xml files with python

```
[3] import pandas_read_xml as pdx
    from pandas_read_xml import flatten, fully_flatten, auto_separate_tables
```

```
df = pdx.read_xml(text_xml, ['root'])
df
```

	@xmlns:xsi	@Comment	@IndraWorksVersion
0	http://www.w3.org/2001/XMLSchema-instance	Persistence file of IndraWorks Oscilloscope	14.22.1035.0

```
[9] df = df.pipe(flatten)
df
```

	@xmlns:xsi	@Comment	@IndraWorksVersion	@CompatibilityVersion	@xmlns
0	http://www.w3.org/2001/XMLSchema-instance	Persistence file of IndraWorks Oscilloscope	14.22.1035.0	10.6	http://www.boschrexroth.de/indraworks.textform...

1 rows x 29 columns

```
[14] df = df.pipe(flatten)
df
```

	@xmlns:xsi	@Comment	@IndraWorksVersion	@CompatibilityVersion	@xmlns
0	http://www.w3.org/2001/XMLSchema-instance	Persistence file of IndraWorks Oscilloscope	14.22.1035.0	10.6	http://www.boschrexroth.de/indraworks.textform...
1	http://www.w3.org/2001/XMLSchema-instance	Persistence file of IndraWorks Oscilloscope	14.22.1035.0	10.6	http://www.boschrexroth.de/indraworks.textform...
2	http://www.w3.org/2001/XMLSchema-instance	Persistence file of IndraWorks Oscilloscope	14.22.1035.0	10.6	http://www.boschrexroth.de/indraworks.textform...
3	http://www.w3.org/2001/XMLSchema-instance	Persistence file of IndraWorks Oscilloscope	14.22.1035.0	10.6	http://www.boschrexroth.de/indraworks.textform...

4 rows x 85 columns

```
[18] key_columns = ['@xmlns:xsi']
     data1 = df.pipe(auto_separate_tables, key_columns)
```

```
[19] data1.keys()
dict_keys(['Devices|Device|Measurements|Measurement|Signals|Signal|GraphicProperties|GraphicSignalProperties|PredefinedBitDescriptions|BitDescriptions',
'Devices|Device|Measurements|Measurement|Signals|Signal|GraphicProperties|GraphicSignalProperties|UserdefinedBitDescriptions|BitDescriptions', 'OscilloscopeData'])
```

```
[20] data1['Devices|Device|Measurements|Measurement|Signals|Signal|GraphicProperties|GraphicSignalProperties|PredefinedBitDescriptions|BitDescriptions']
```

	@xmlns:xsi	@id
0	http://www.w3.org/2001/XMLSchema-instance	ref-9
1	http://www.w3.org/2001/XMLSchema-instance	ref-13
2	http://www.w3.org/2001/XMLSchema-instance	ref-17
3	http://www.w3.org/2001/XMLSchema-instance	ref-21

[illegible]

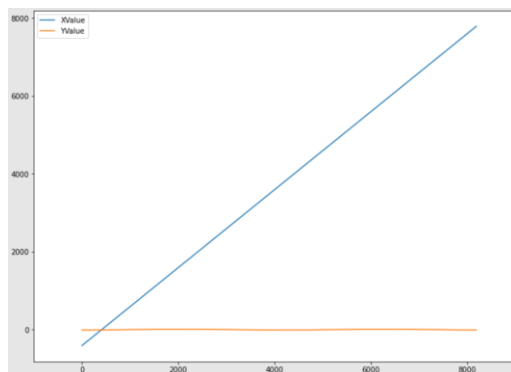
Analysis The Data:

For Analysis of the xml files, we use python. And we did this work on a Colab platform. In these 5 xml files there was the data received from the motor. There we got four types of parameters. 1) Position command value 2) Position feedback value 3) Velocity feedback value 4) Actual output current value (Absolute value). And every file we got 2 values. Data of X- Time Ms and Y - value of parameter.

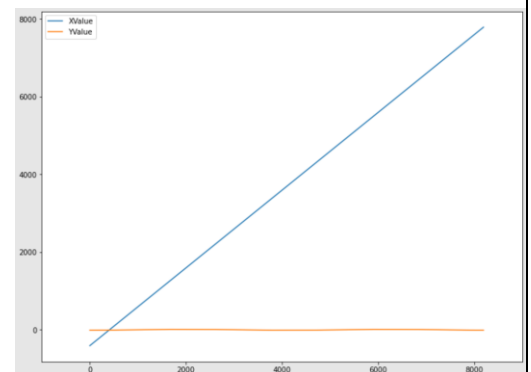
Using python tried to understand the values of X and Y through a graph. Here are all the Graphs that we got by using python.

Position Command Value:

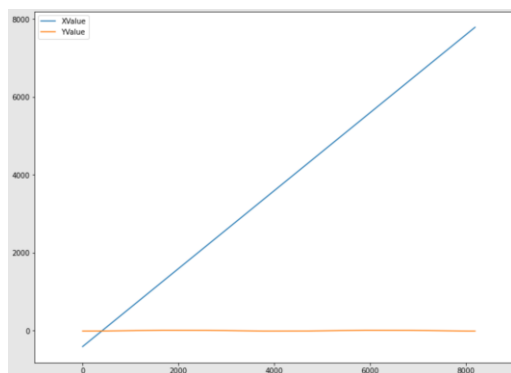
Xml file 1



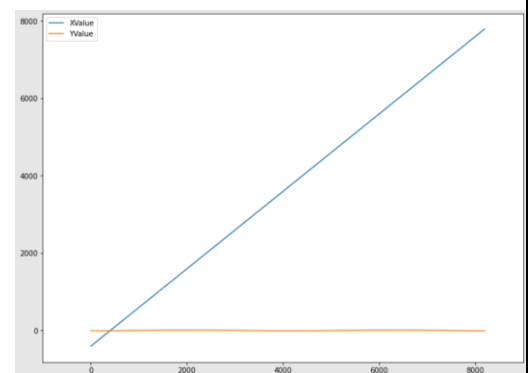
Xml file 3



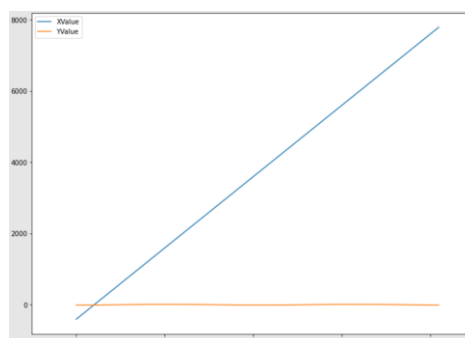
Xml file 2:



Xml file 4:

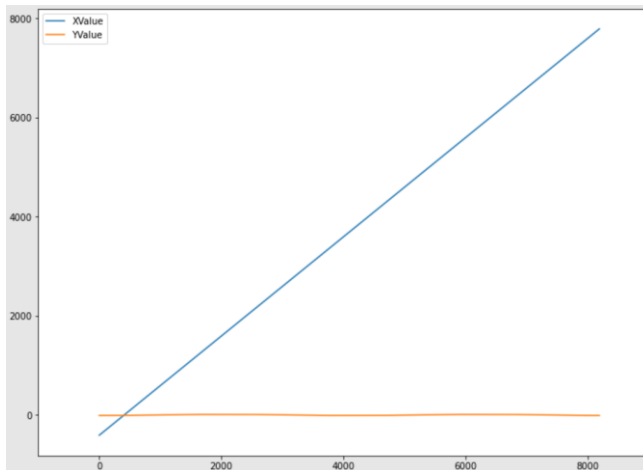


Xml file 5:

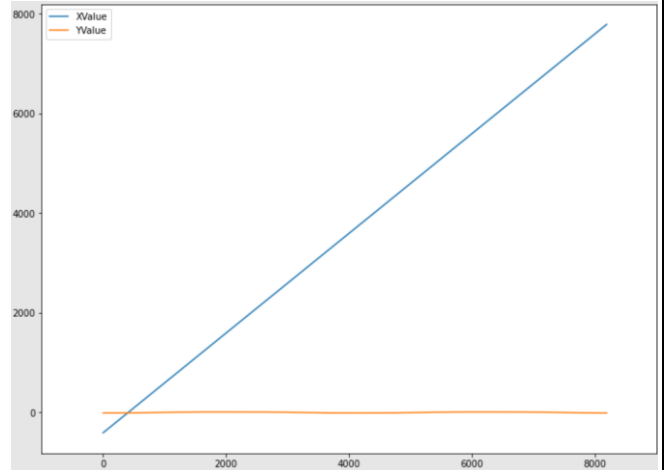


Position feedback value:

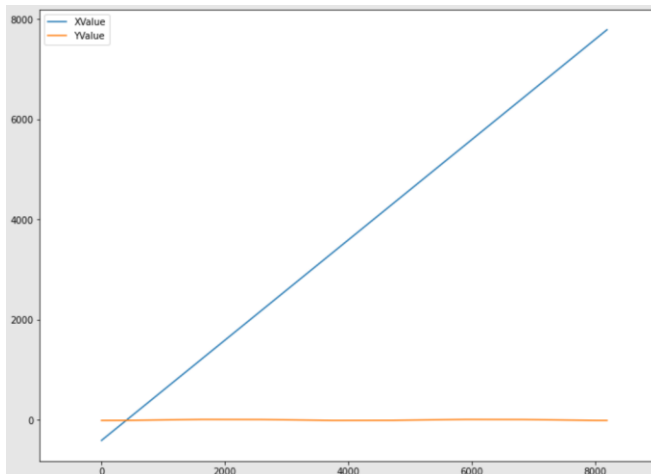
Xml file 1:



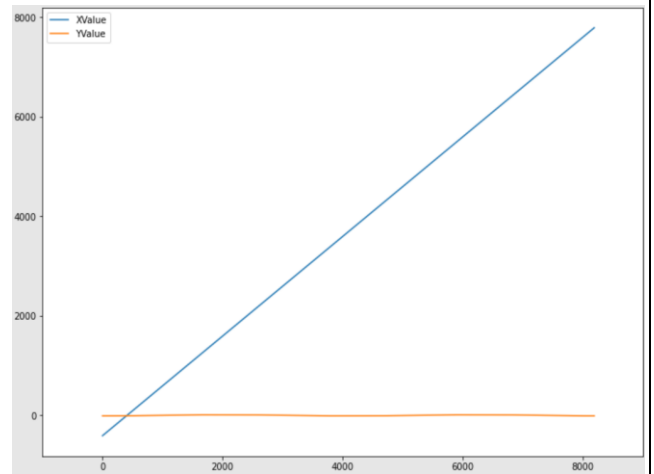
Xml file 2:



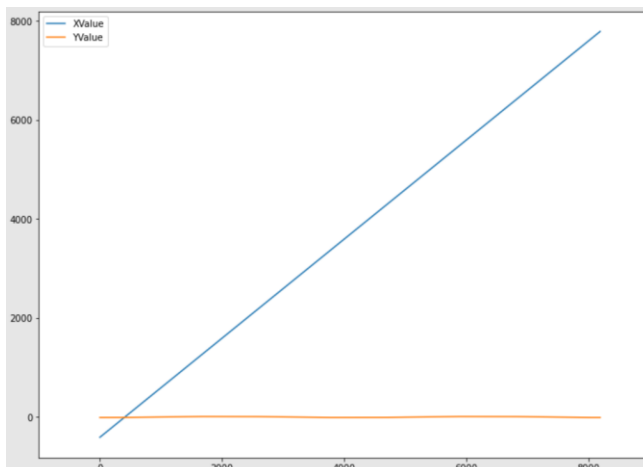
Xml file 3:



Xml file 4:

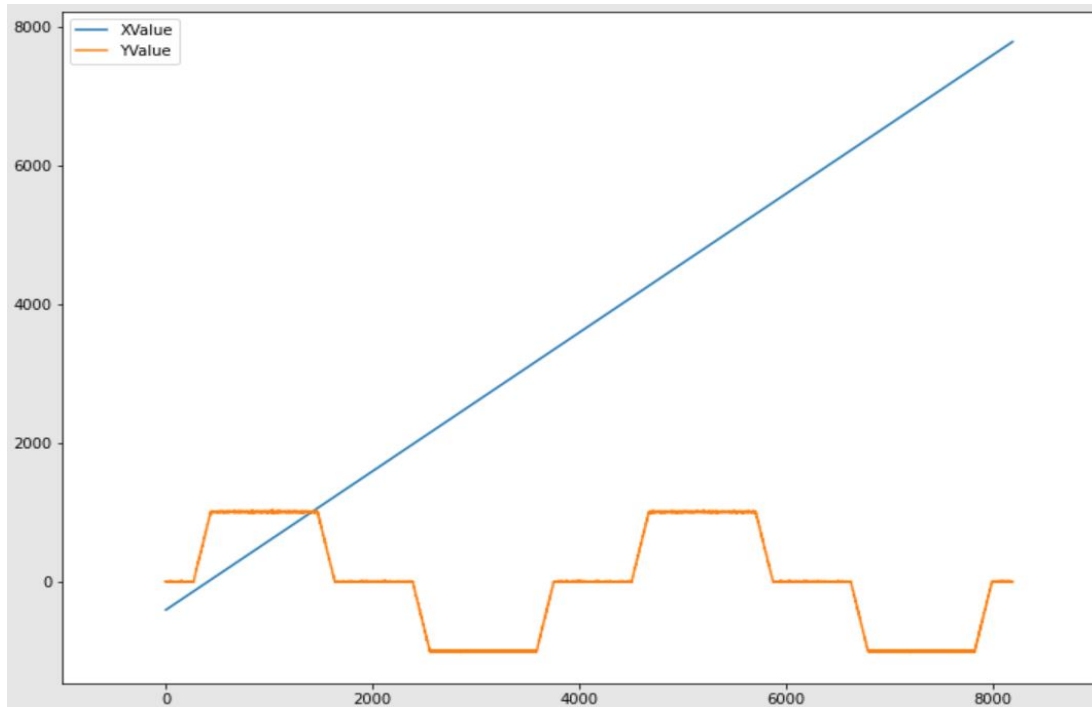


Xml file 5:

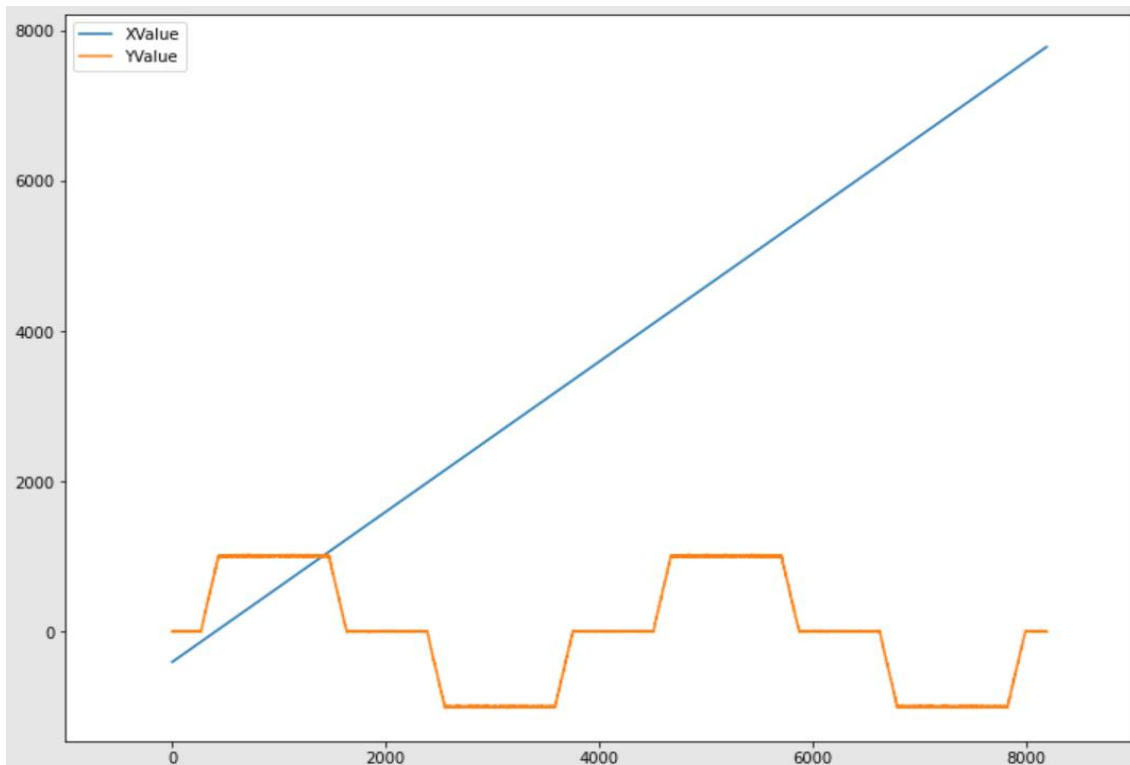


Velocity Feedback value:

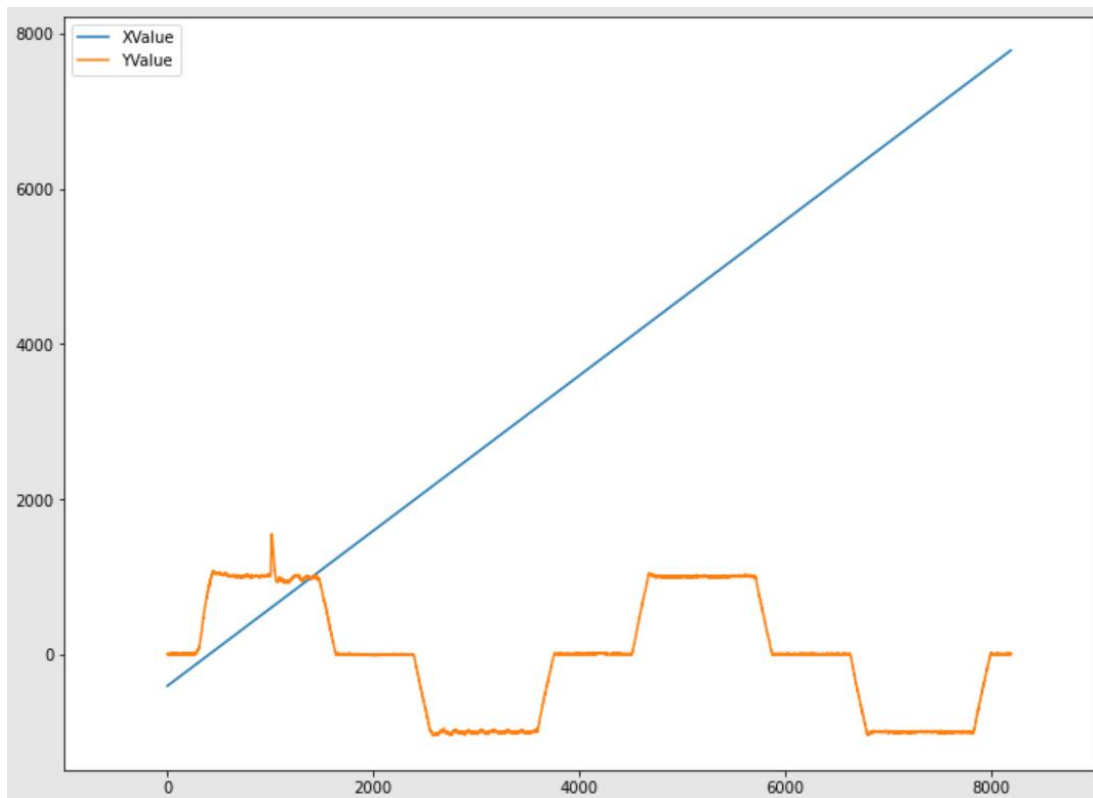
Xml file1:



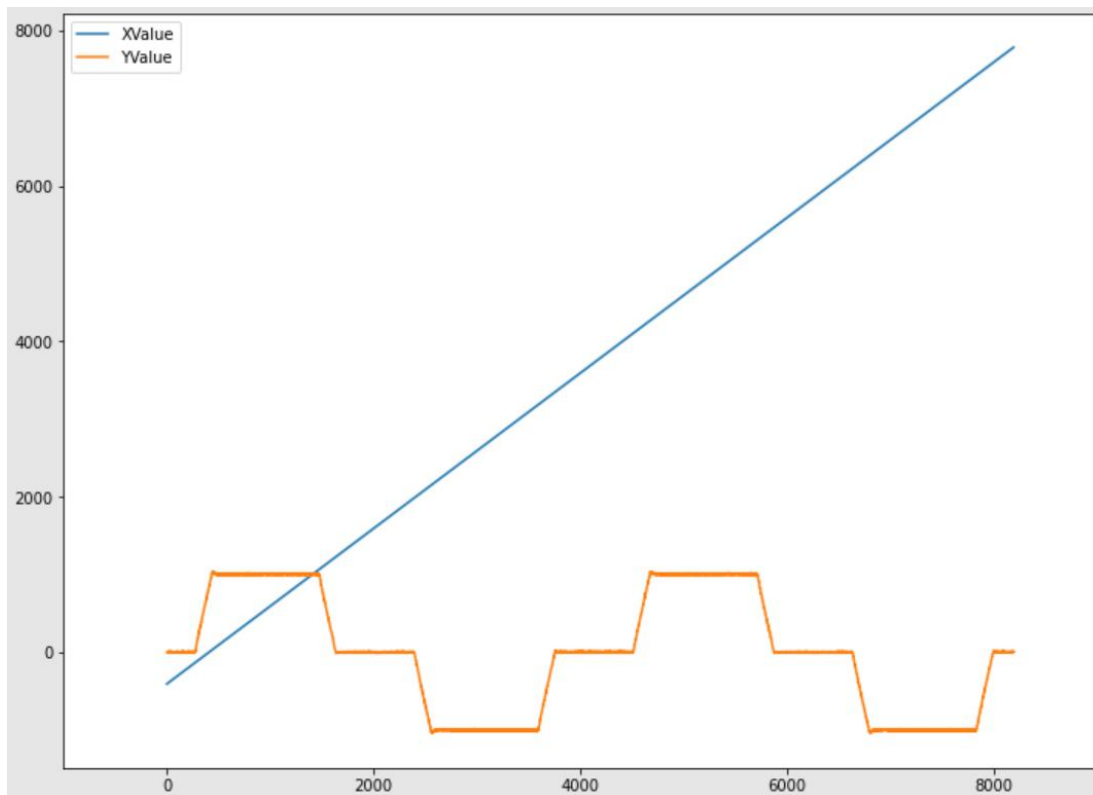
Xml file 2



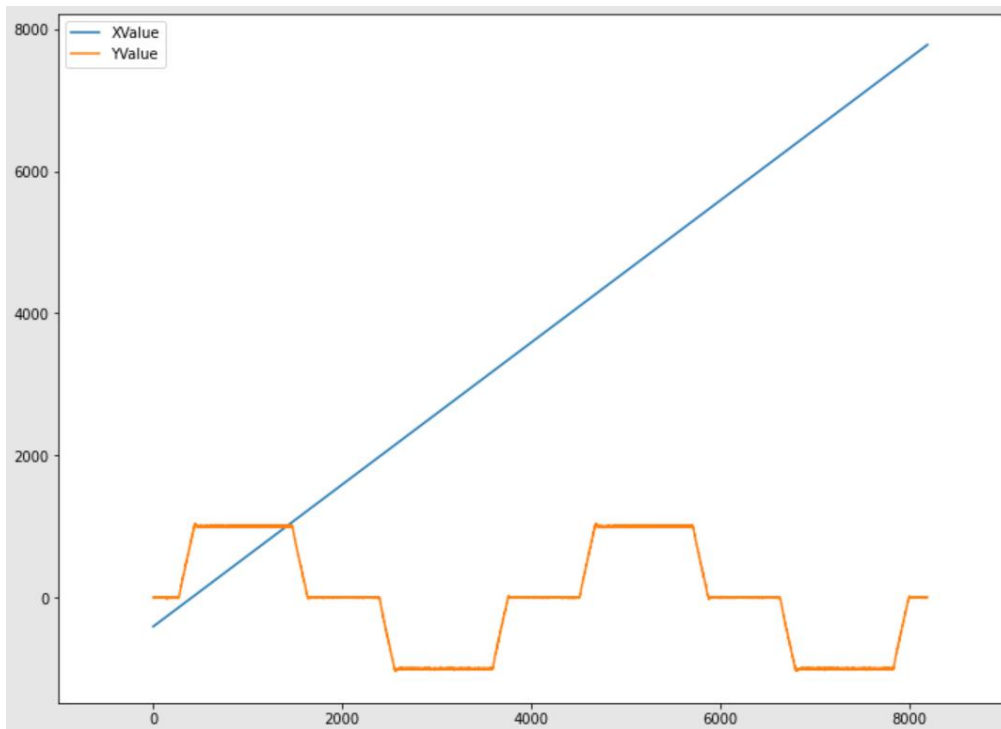
Xml file3



Xml file 4:

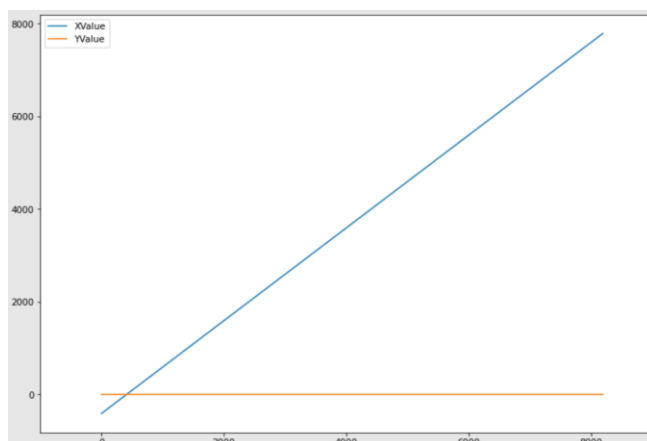


Xml file 5:

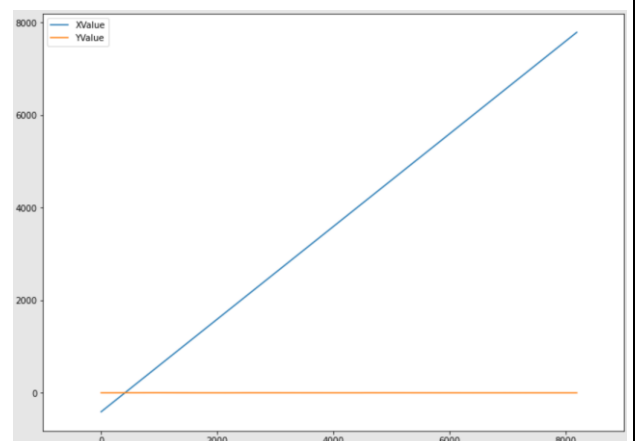


Actual output current value:

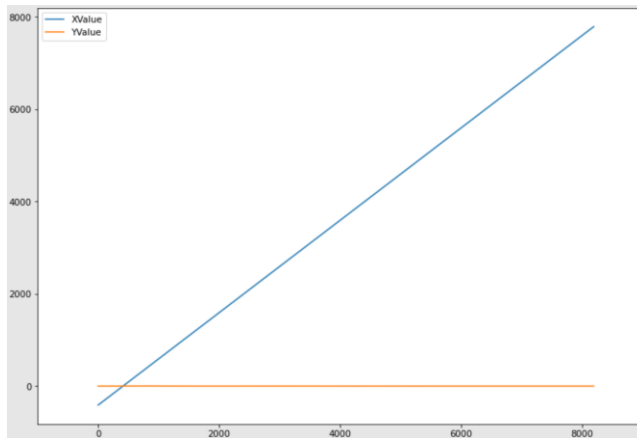
Xml file 1:



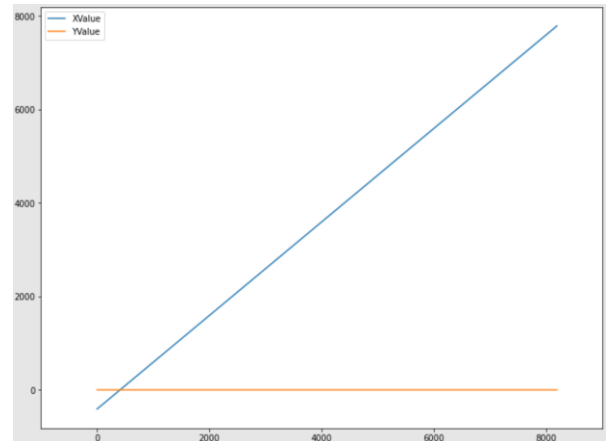
Xml file 2:



Xml file 3:



Xml file 4:



Xml file 5:

