AIML CA2

Tan Yu Hoe	P2026309
Diploma in Applied AI and Analytics	DAAA/FT/2A/04
School of Computing	Singapore Polytechnic
tanyh.20@ichat.sp.edu.sg	
AIML	ST1511
CA2 Assignment	

Part A: Air Pollution Forecasting

Prediction Task

The prediction task is to train a time series more for future air pollution forecasting.

Dataset

The dataset seems to be derived from an Air Quality dataset found in UCI Machine Learning Repository , with some of the values in this dataset to match with the Air Quality dataset. Based on the dataset information, here we can assume that the data is curated using 4 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensory Device. The device was located on the field in a significantly polluted area, at road level, within an Italian city.

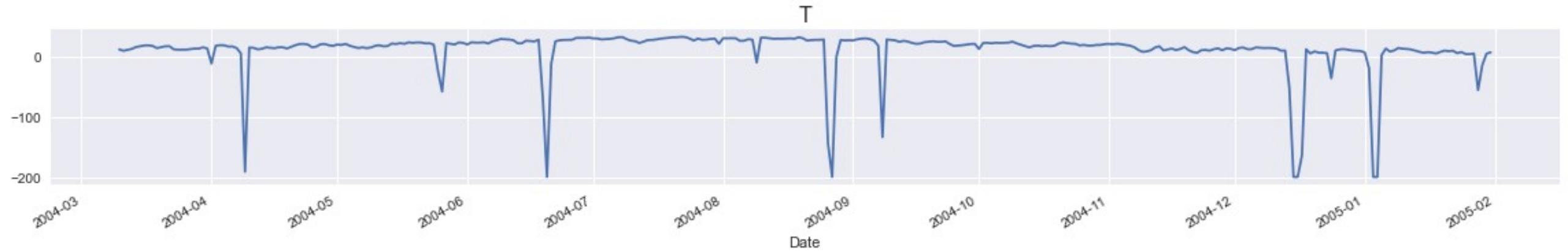
Metadata

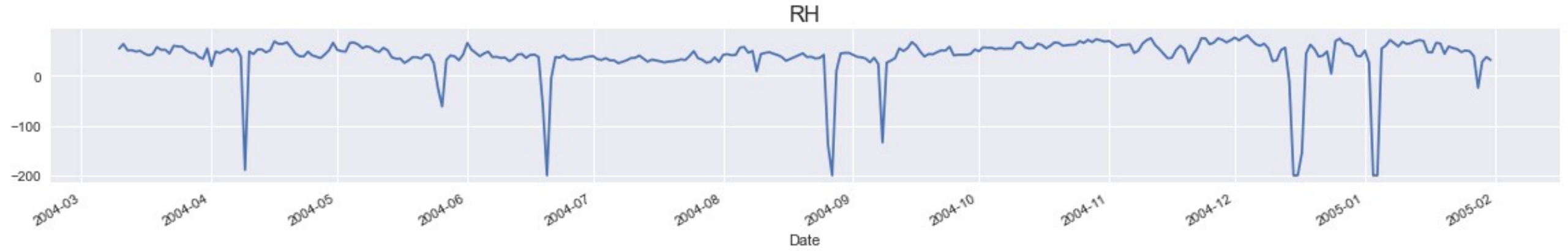
Attribute	Description	Unit of Measurement	Type
Date	DateTime	YYYY-MM-DD	Date
T	Climate Temperature	Degree Celsius	Numerical – Continuous
RH	Relative Humidity	Percentage	Numerical – Continuous
Gas	Type of Air Pollutant	-	Categorical –

	Date	T	RH	Gas	Value
0	2004-03-10	12.020833	54.883334	CO	1316.500000
1	2004-03-11	9.833333	64.069791	CO	1244.062500
2	2004-03-12	11.292708	51.107292	CO	1281.562500
3	2004-03-13	12.866319	51.530903	CO	1330.555556
4	2004-03-14	16.016667	48.843750	CO	1360.927083

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1312 entries, 2004-03-10 to 2005-01-31
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   T       1312 non-null   float64
 1   RH      1312 non-null   float64
 2   Gas     1312 non-null   object 
 3   Value   1312 non-null   float64
dtypes: float64(3), object(1)
memory usage: 51.2+ KB
```

Exploratory data

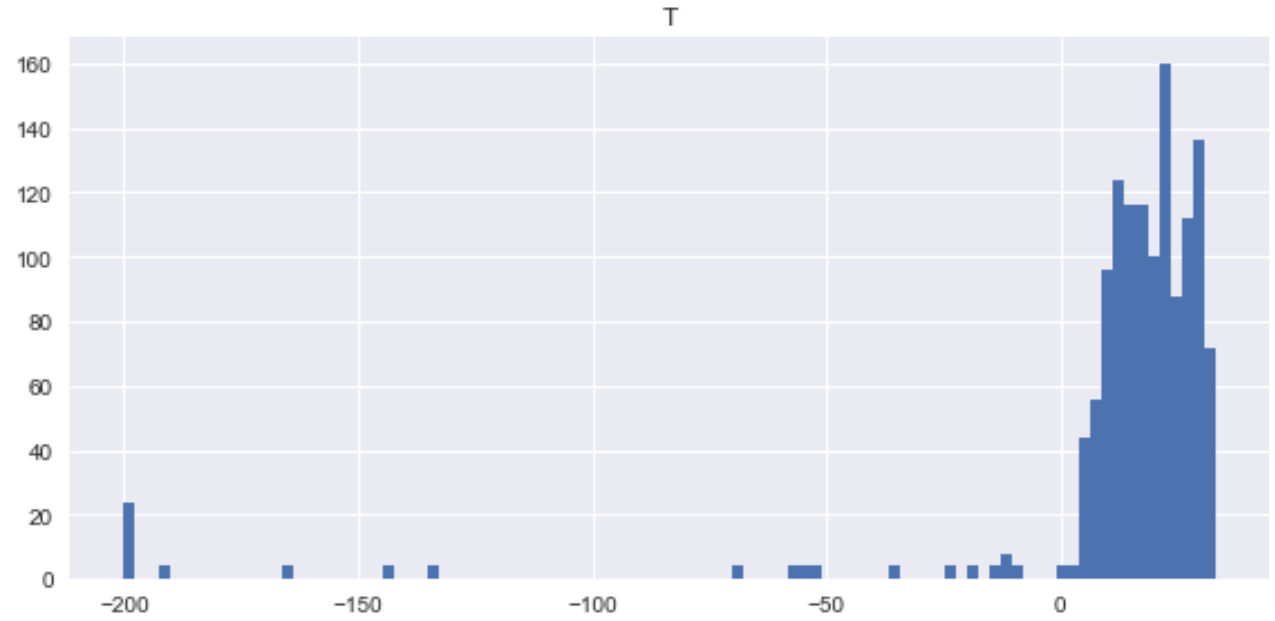


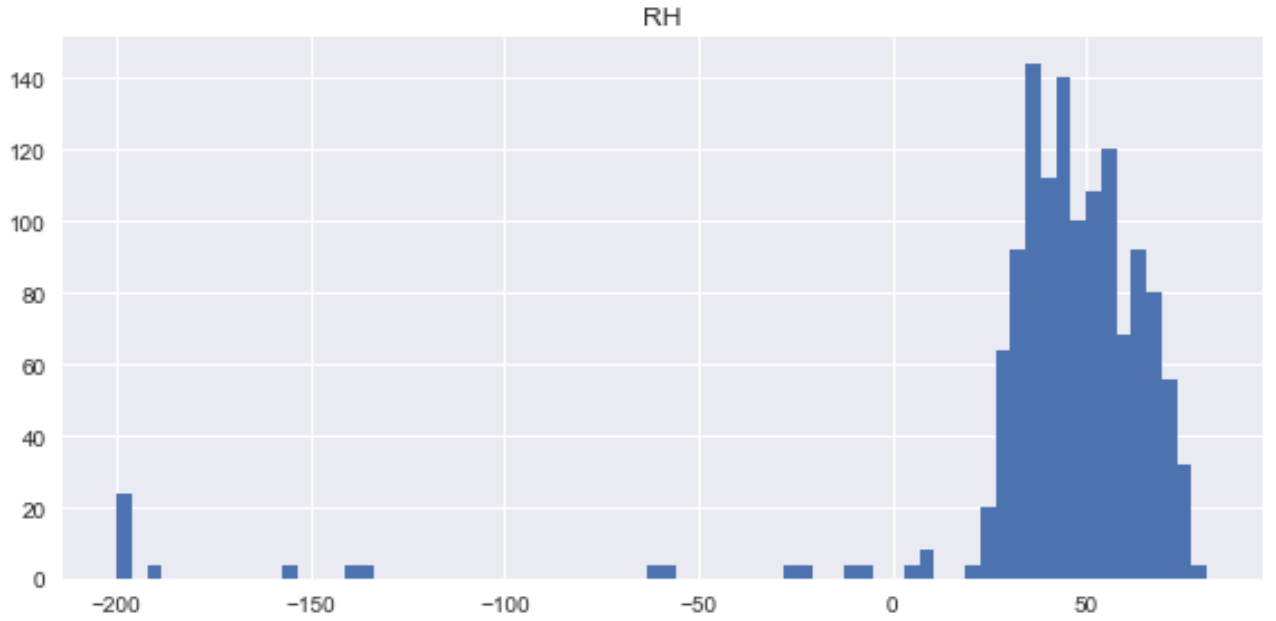


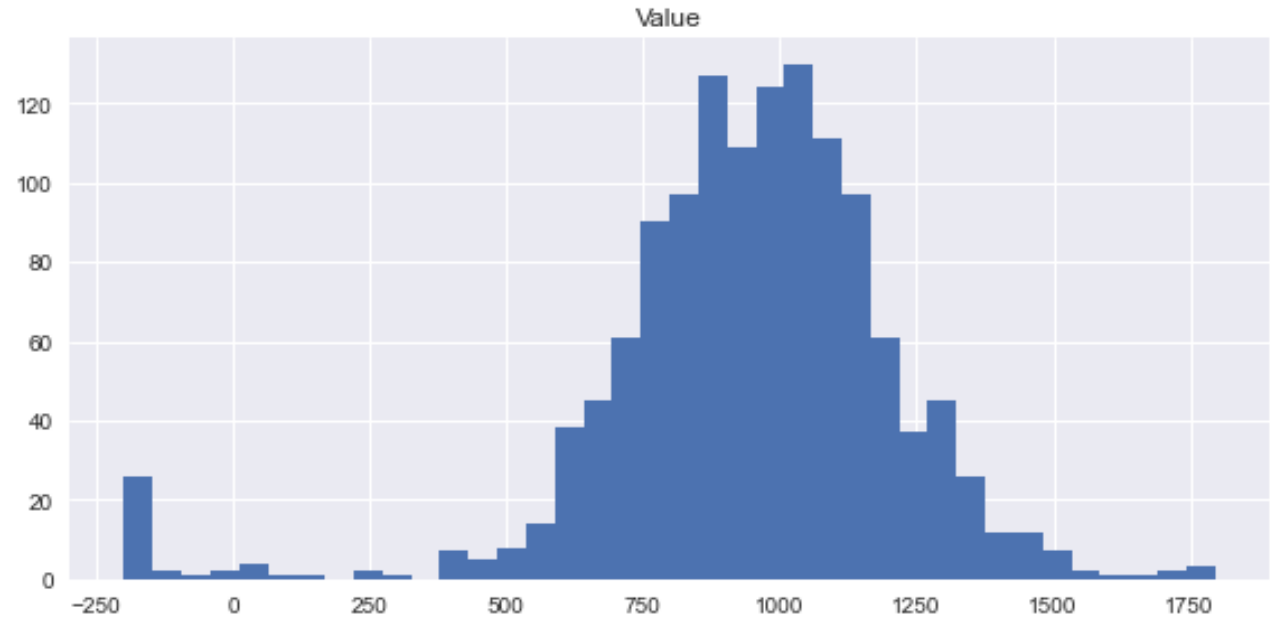
Descriptive Statistics

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
T	1312	NaN	NaN	NaN	11.6349	37.0418	-200	12.3878	18.8323	25.5571	32.9979
RH	1312	NaN	NaN	NaN	39.8737	43.2178	-200	36.3482	46.079	57.9323	81.1042
Gas	1312	4	O3	328	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Value	1312	NaN	NaN	NaN	937.774	281.9	-200	810.464	961.714	1100.71	1795.29

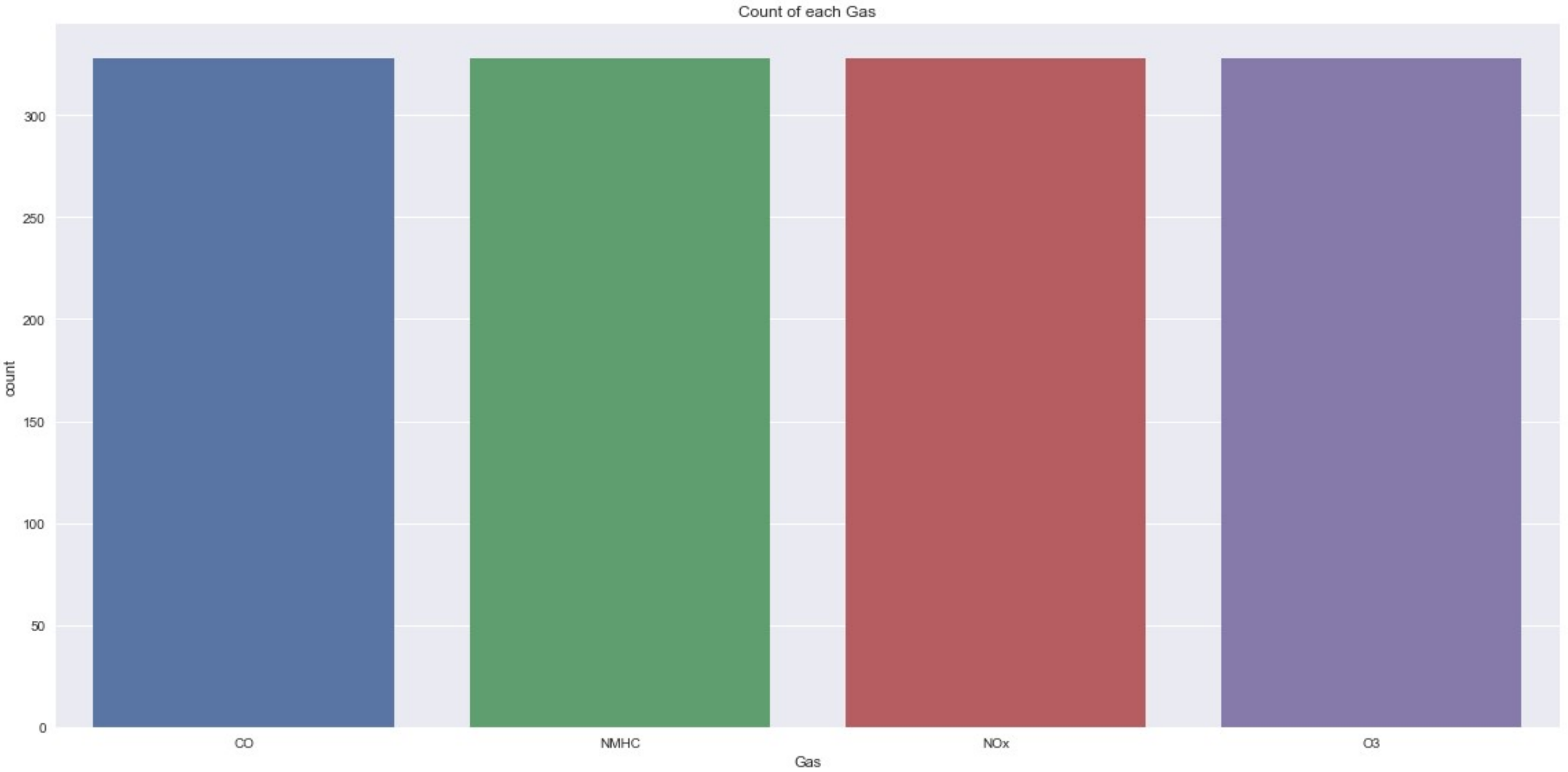
Exploratory data



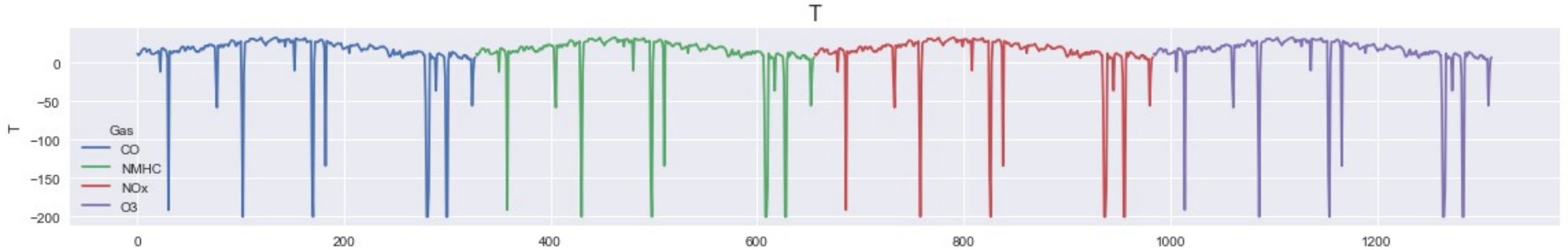


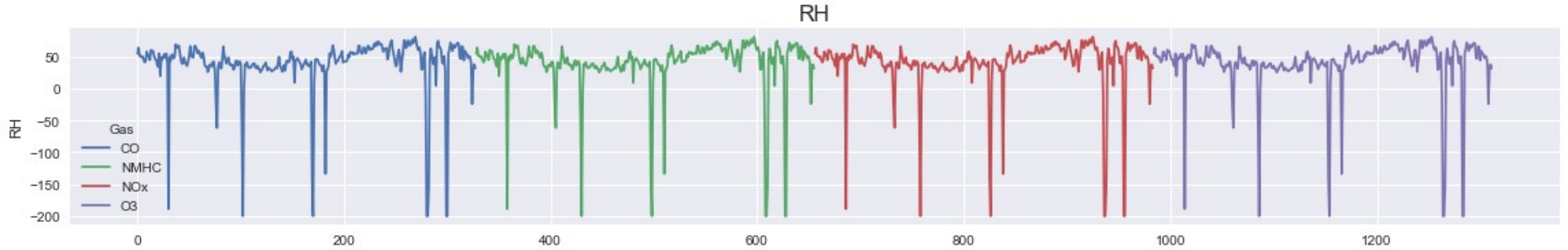


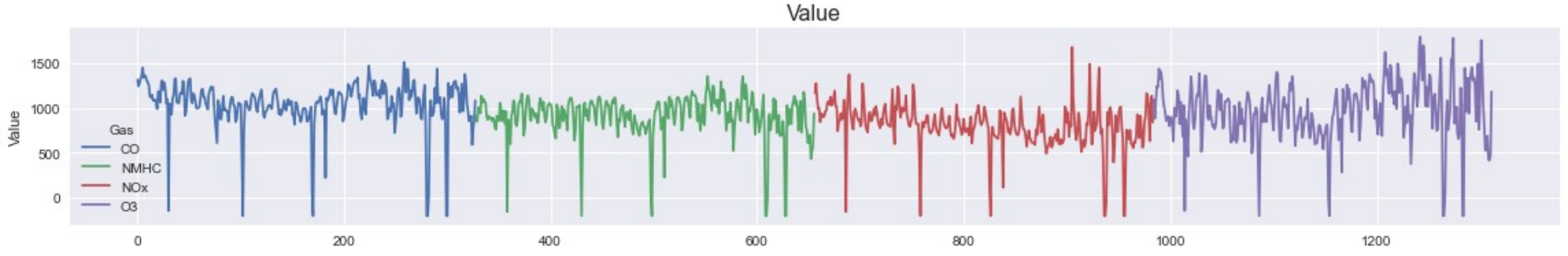
Exploratory data



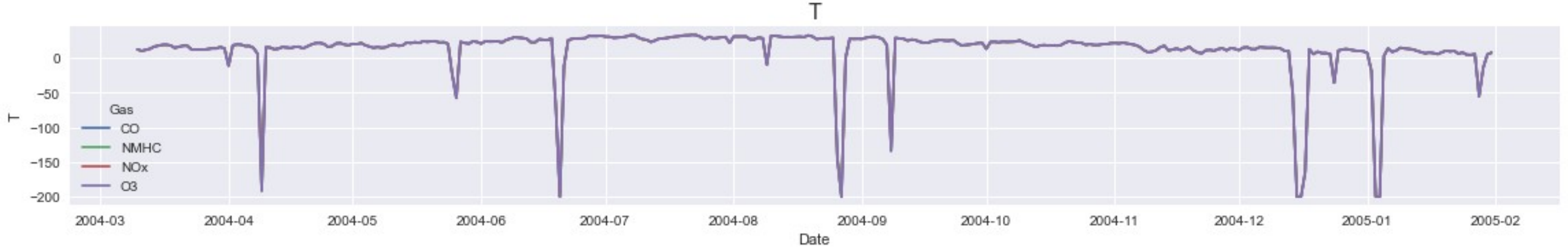
Exploratory data

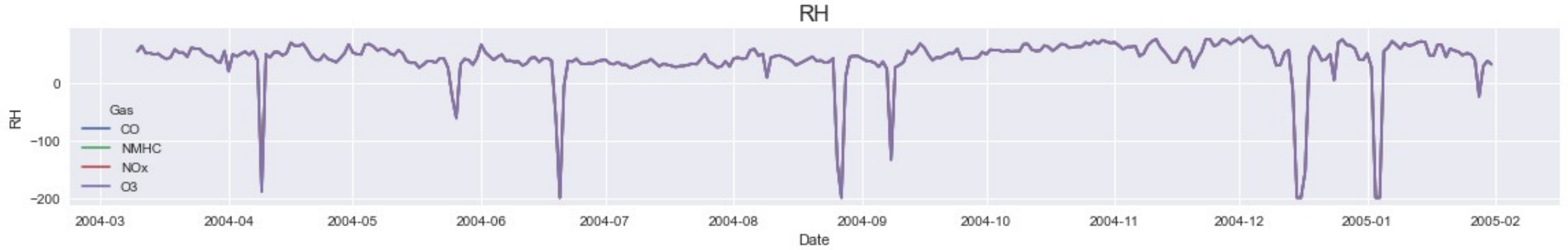


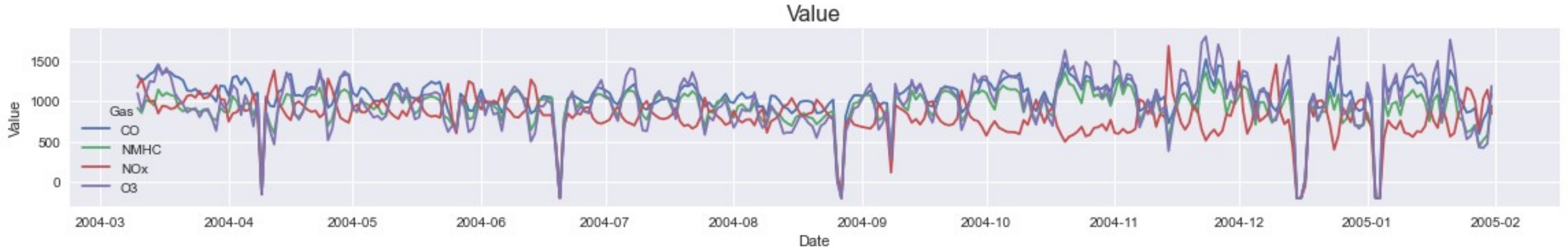




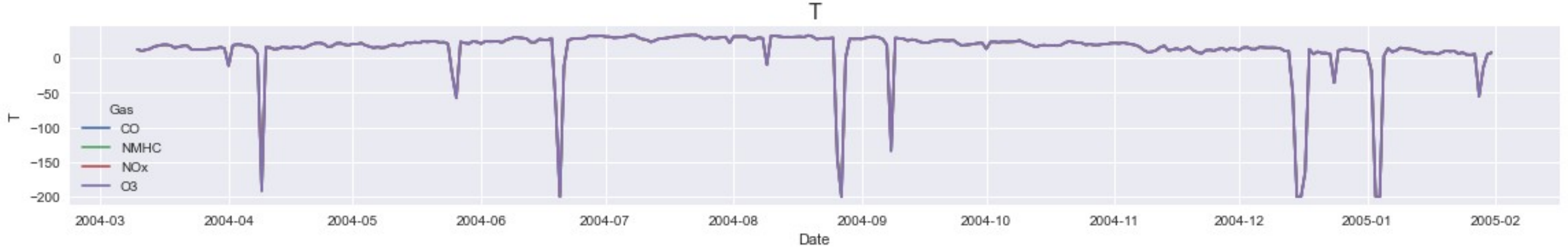
Exploratory data

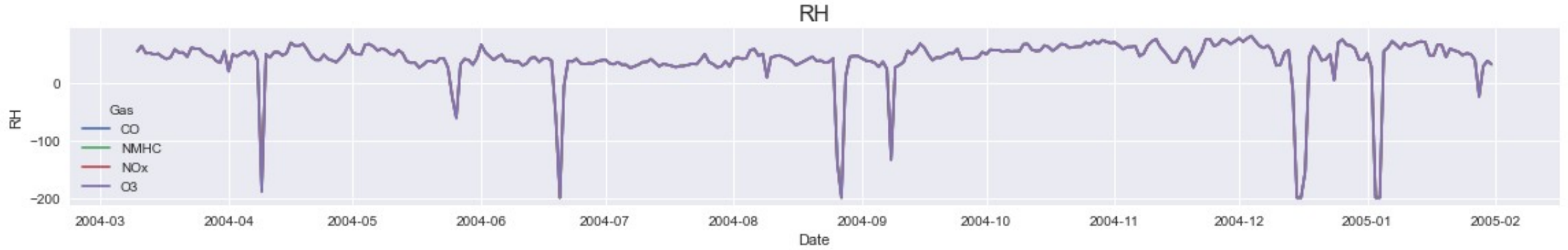


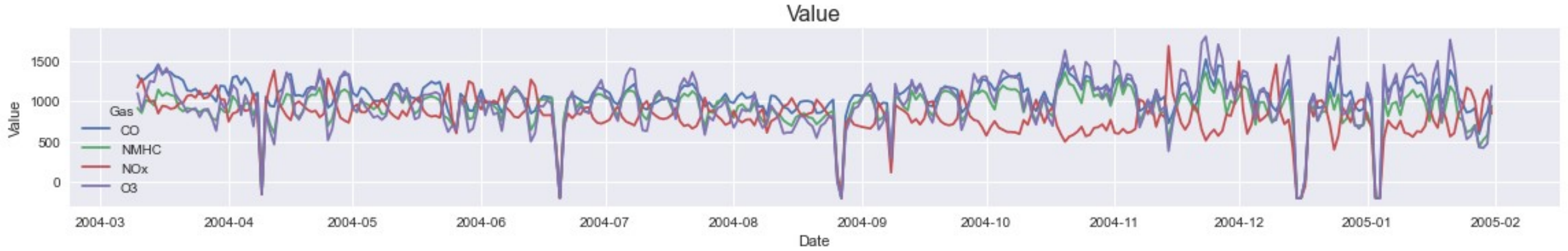




Exploratory data





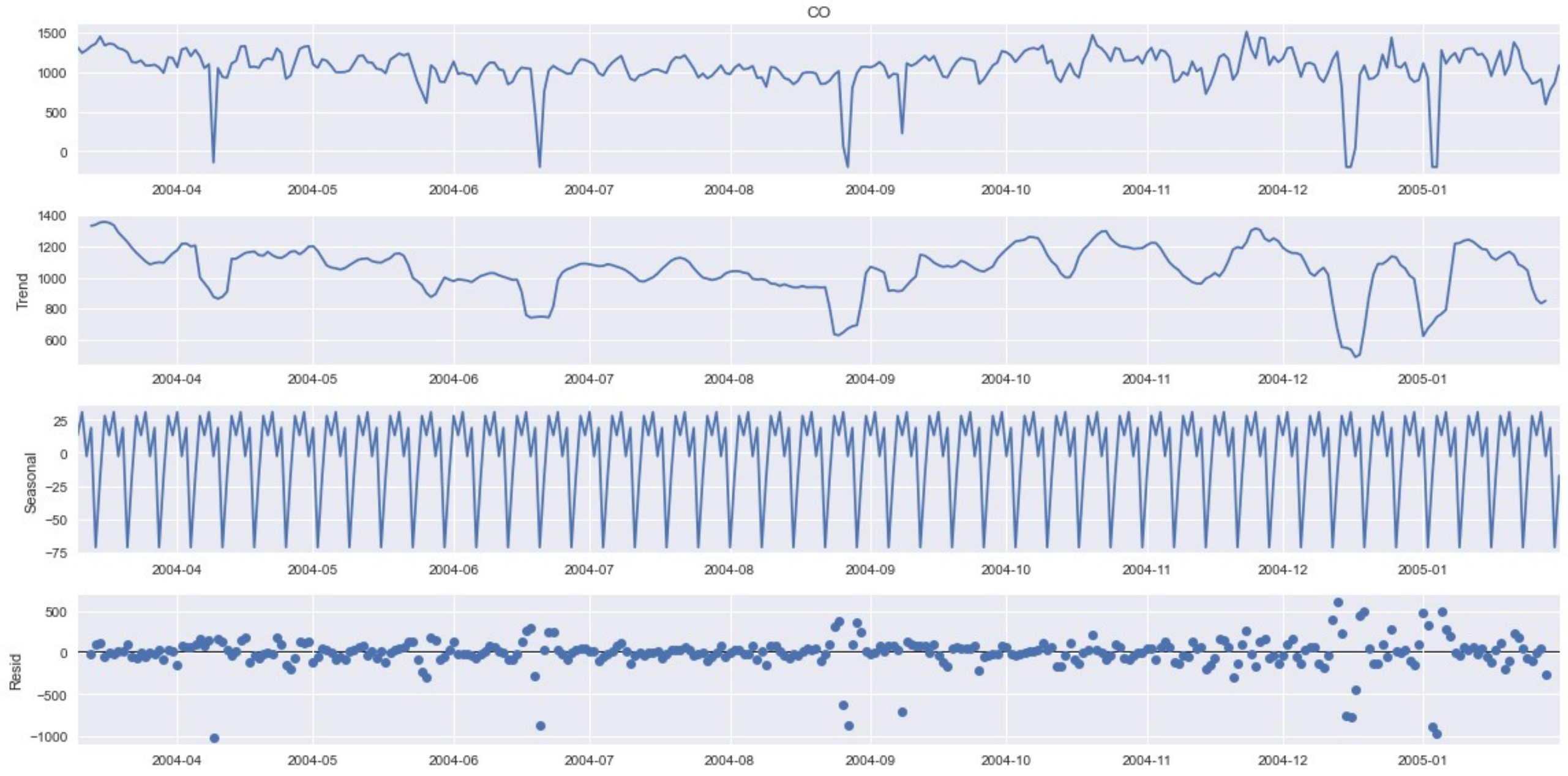


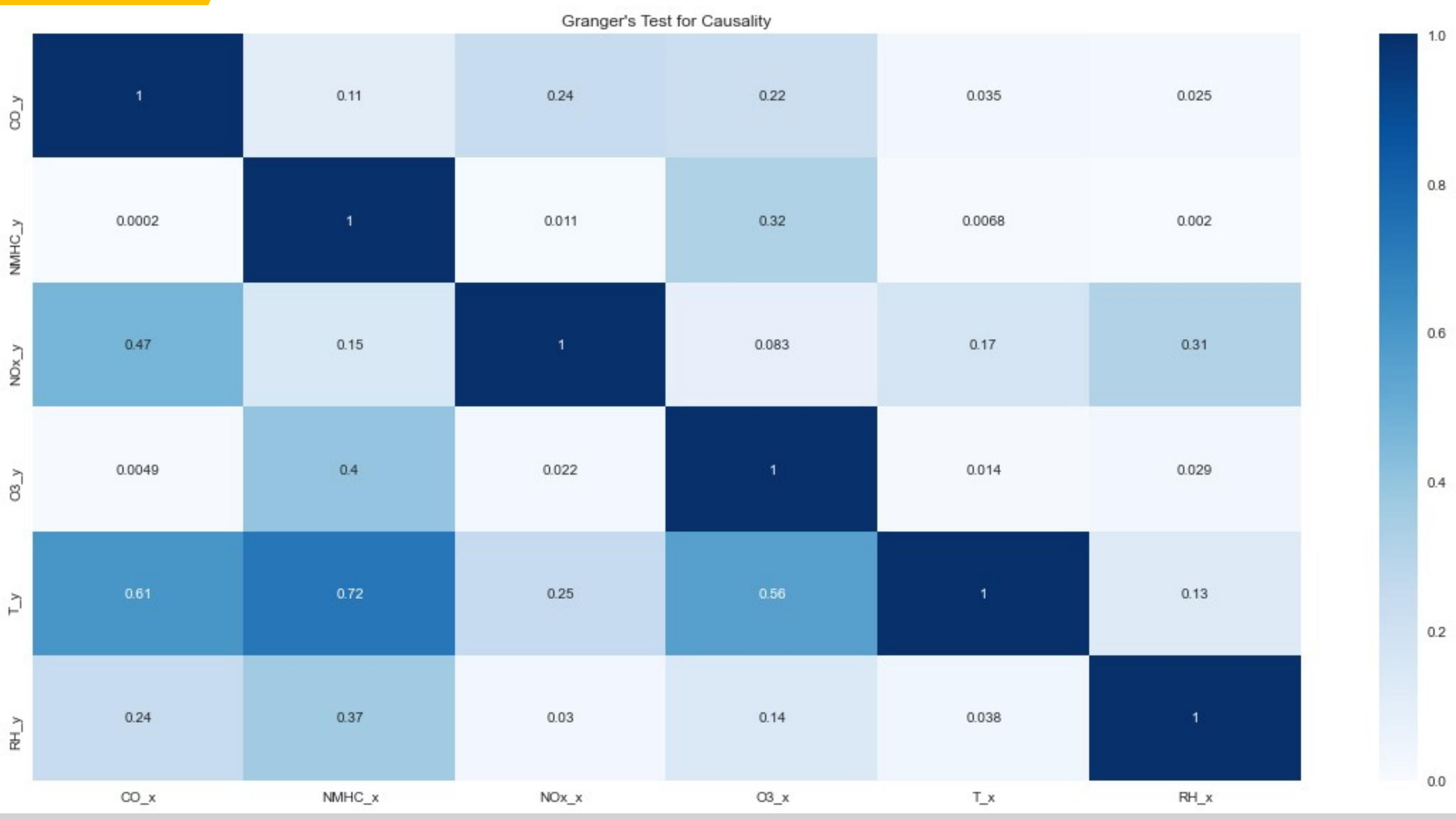
Pivot DataFrame

Since the data seems to have duplicate datetime index per gas, pivoting the DataFrame is necessary. Furthermore, this would ensure that we have a multivariate target variables in the DataFrame.

Date	CO	NMHC	NOx	O3	T	RH
2004-03-10	1316.500000	912.250000	1167.250000	1096.041667	12.020833	54.883334
2004-03-11	1244.062500	851.802083	1277.187500	885.031250	9.833333	64.069791
2004-03-12	1281.562500	1008.229167	1101.718750	1084.218750	11.292708	51.107292
2004-03-13	1330.555556	992.822917	993.159722	1245.781250	12.866319	51.530903
2004-03-14	1360.927083	943.854167	1001.104167	1234.177083	16.016667	48.843750
...	...	...	...	...	...	...
2005-01-27	911.777778	703.312500	997.059028	691.475694	5.267708	39.614930
2005-01-28	592.864583	434.350694	635.225694	429.375000	-55.515972	-24.010417
2005-01-29	769.625000	518.093750	1024.666667	418.072917	-14.272917	28.563542
2005-01-30	864.642361	573.684028	1136.718750	474.392361	4.848611	37.832986
2005-01-31	1084.500000	939.791667	842.177083	1184.166667	7.273958	31.809375

328 rows × 6 columns


CO
1500
1000
500
0
2004-04
2004-05
2004-06
2004-07
2004-08
2004-09
2004-10
2004-11
2004-12
2005-01
Trend
1400
1200
1000
800
600
Seasonal
25
0
−25
−50
−75
Resid
500
0
−500
−1000


Granger's Test for Causality
CO_y
NMHC_y
NOx_y
O3_y
T_y
RH_y
1
0.11
0.24
0.22
0.035
0.025
0.0002
1
0.011
0.32
0.0068
0.002
0.47
0.15
1
0.083
0.17
0.31
0.0049
0.4
0.022
1
0.014
0.029
0.61
0.72
0.25
0.56
1
0.13
0.24
0.37
0.03
0.14
0.038
1
CO_x
NMHC_x
NOx_x
O3_x
T_x
RH_x
1.0
0.8
0.6
0.4
0.2
0.0

Johansen’s Test for Cointegration

Cointegration test helps to establish the presence of a statistically significant connection between two or more time series. This is the basic premise on which multivariate models is based on.

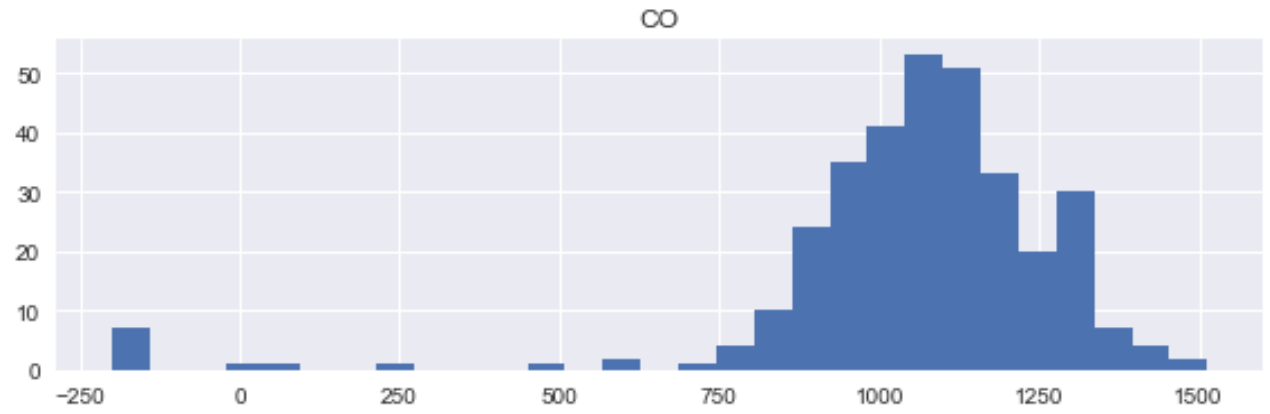
Name	::	Test Stat	> C(95%)	=>	Signif
CO	::	151.09	> 83.9383	=>	True
NMHC	::	90.83	> 60.0627	=>	True
NOx	::	45.03	> 40.1749	=>	True
O3	::	19.66	> 24.2761	=>	False
T	::	4.85	> 12.3212	=>	False
RH	::	0.64	> 4.1296	=>	False

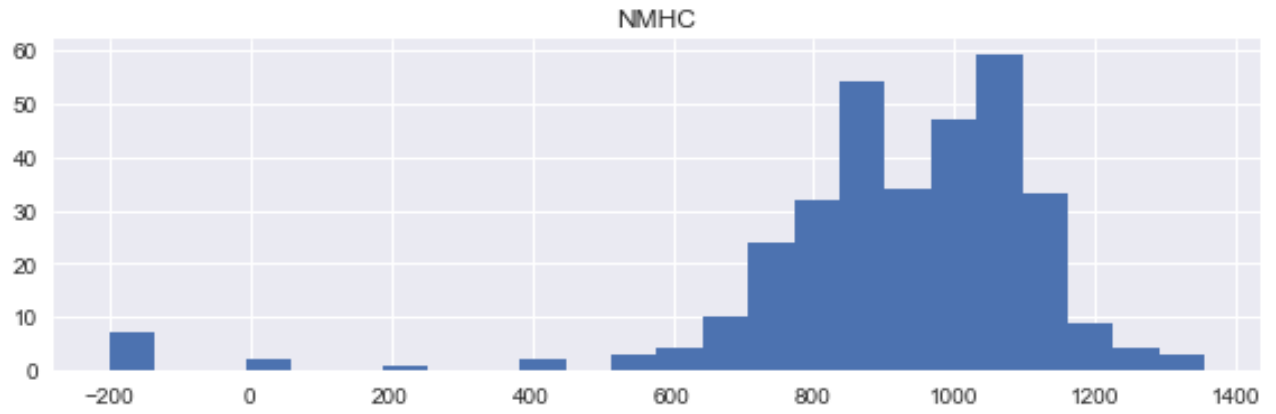
Testing for Stationarity

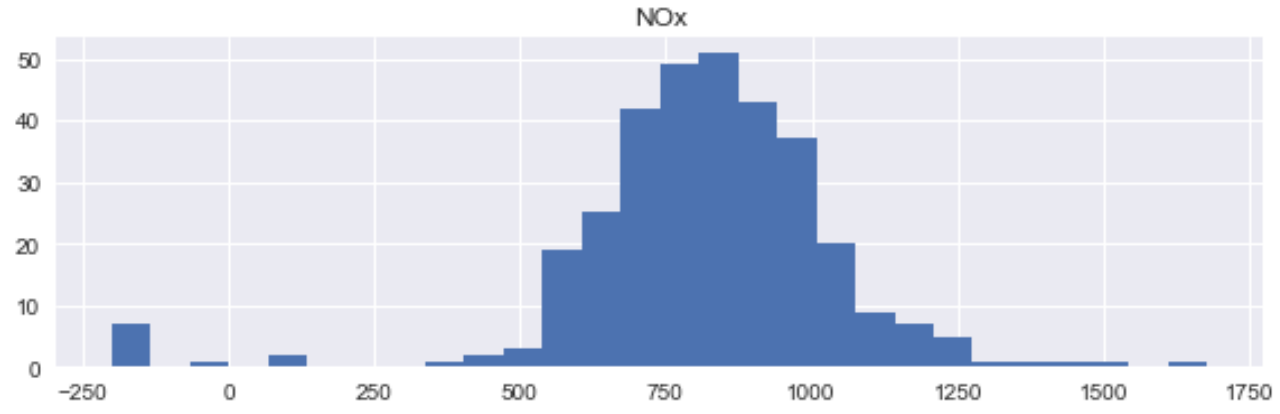
```
   Augmented Dickey-Fuller Test on "CO"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -9.5435
No. Lags Chosen       = 1
Critical value 1%     = -3.451
Critical value 5%     = -2.87
Critical value 10%    = -2.572
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "NMHC"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -9.6683
No. Lags Chosen       = 1
Critical value 1%     = -3.451
Critical value 5%     = -2.87
Critical value 10%    = -2.572
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.
```

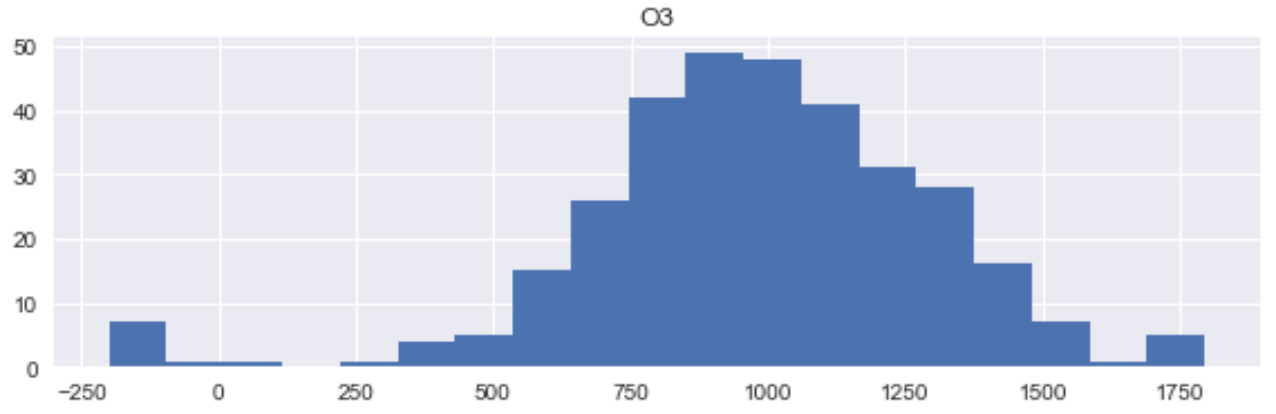
```
   Augmented Dickey-Fuller Test on "NOx"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -10.0552
No. Lags Chosen       = 1
Critical value 1%     = -3.451
Critical value 5%     = -2.87
Critical value 10%    = -2.572
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "O3"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -9.5691
No. Lags Chosen       = 1
Critical value 1%     = -3.451
Critical value 5%     = -2.87
Critical value 10%    = -2.572
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.
```

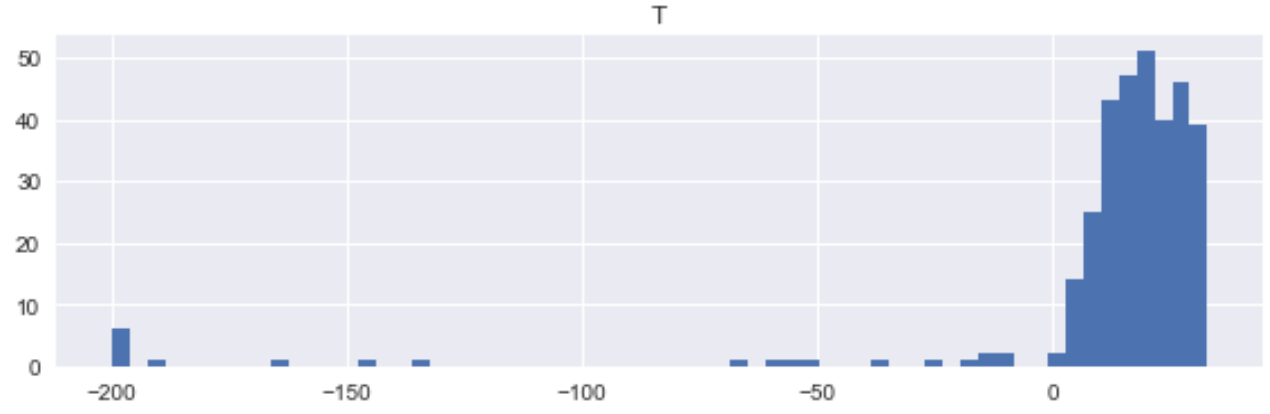
```
   Augmented Dickey-Fuller Test on "T"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -10.251
No. Lags Chosen       = 1
Critical value 1%     = -3.451
Critical value 5%     = -2.87
Critical value 10%    = -2.572
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "RH"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -10.2458
No. Lags Chosen       = 1
Critical value 1%     = -3.451
Critical value 5%     = -2.87
Critical value 10%    = -2.572
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.
```

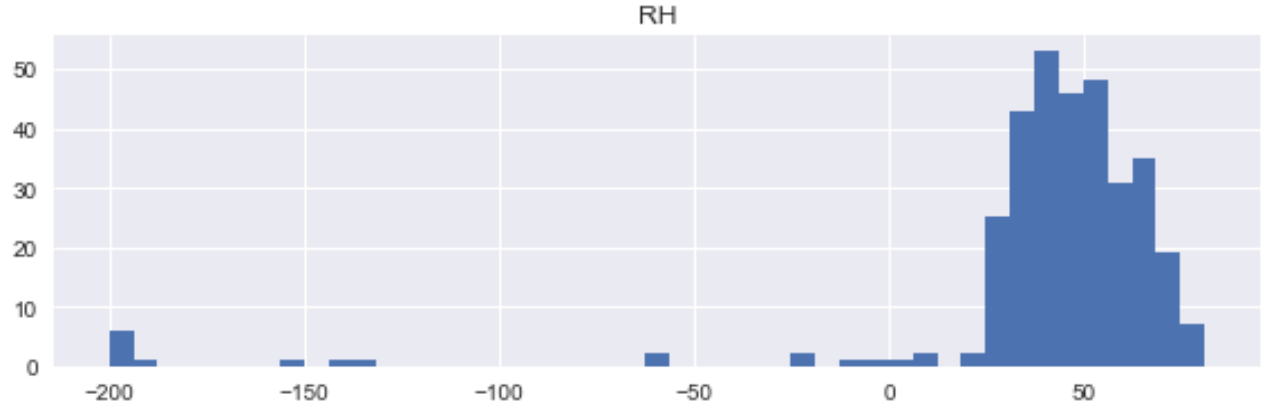

CO
50
40
30
20
10
0
−250
0
250
500
750
1000
1250
1500

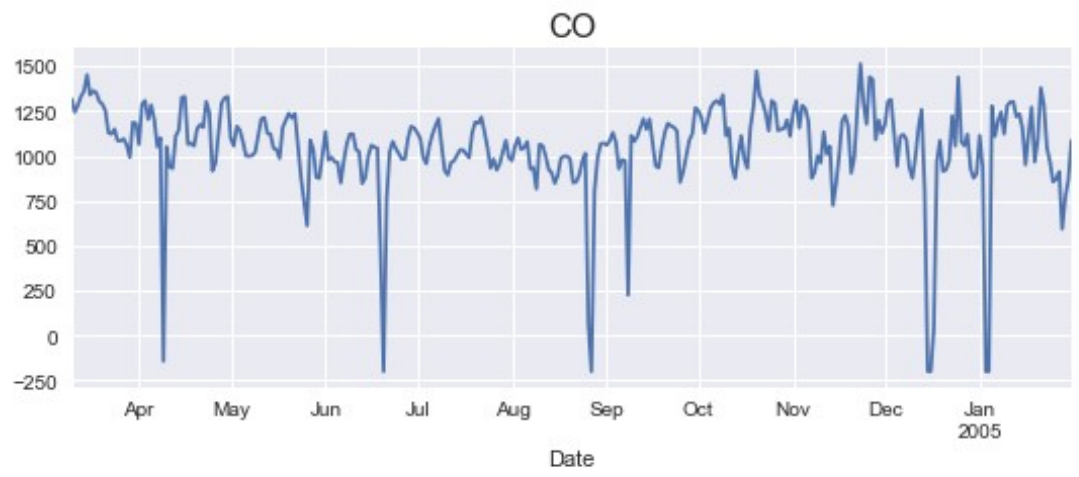

NMHC
60
50
40
30
20
10
0
−200
0
200
400
600
800
1000
1200
1400

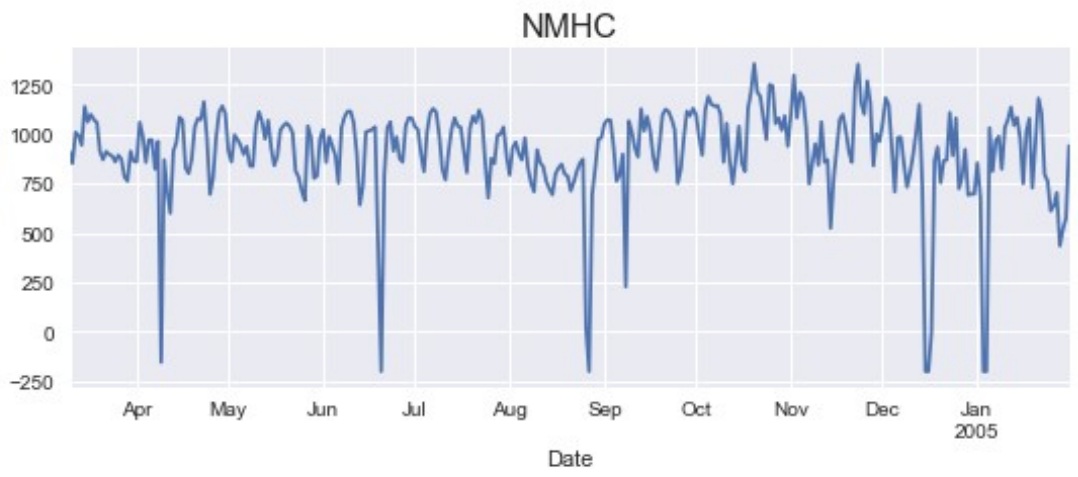

NOx
50
40
30
20
10
0
−250
0
250
500
750
1000
1250
1500
1750

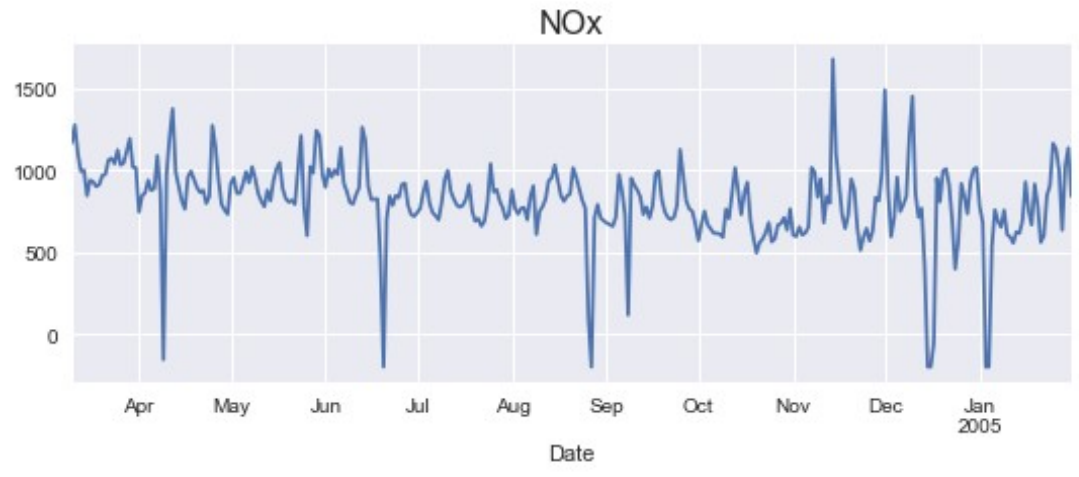

O3
50
40
30
20
10
0
−250
0
250
500
750
1000
1250
1500
1750

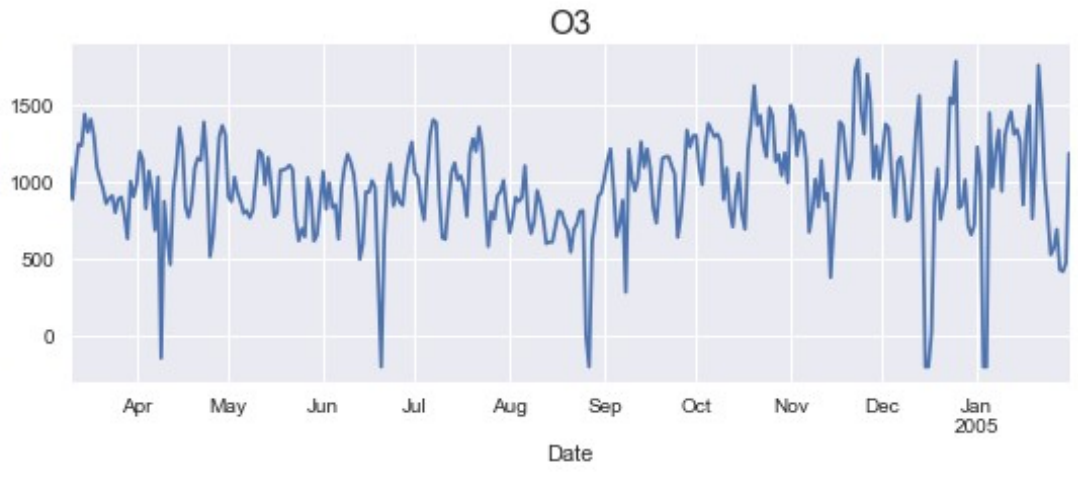

T
50
40
30
20
10
0
−200
−150
−100
−50
0

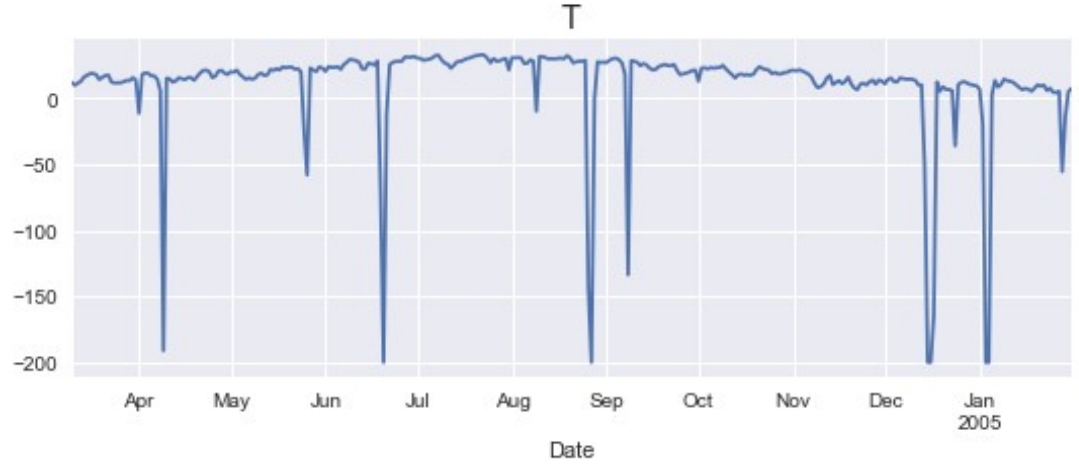

RH
50
40
30
20
10
0
−200
−150
−100
−50
0
50

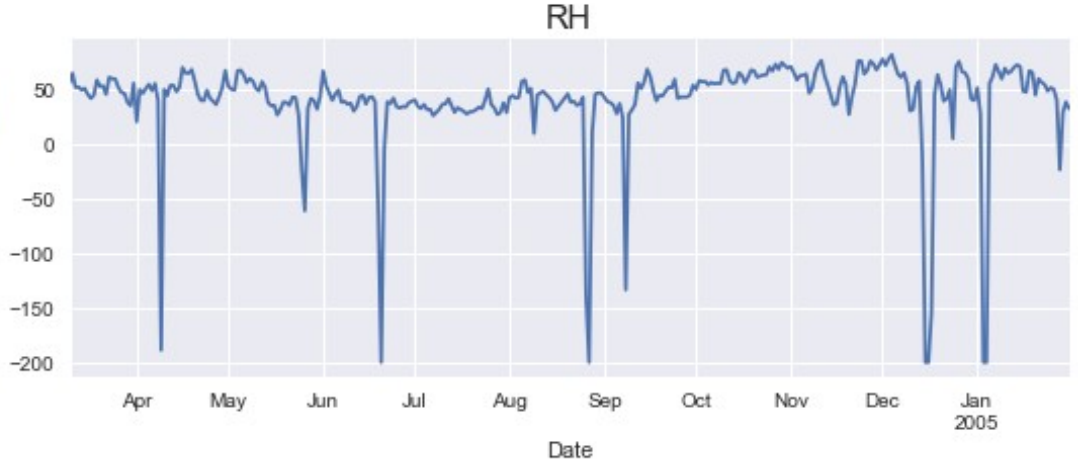

CO
1500
1250
1000
750
500
250
0
−250
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

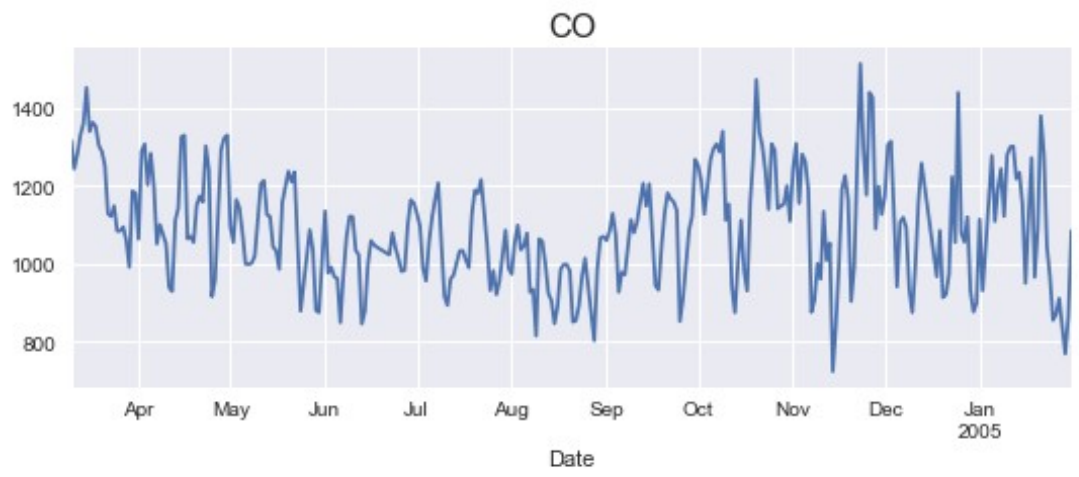

NMHC
1250
1000
750
500
250
0
−250
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

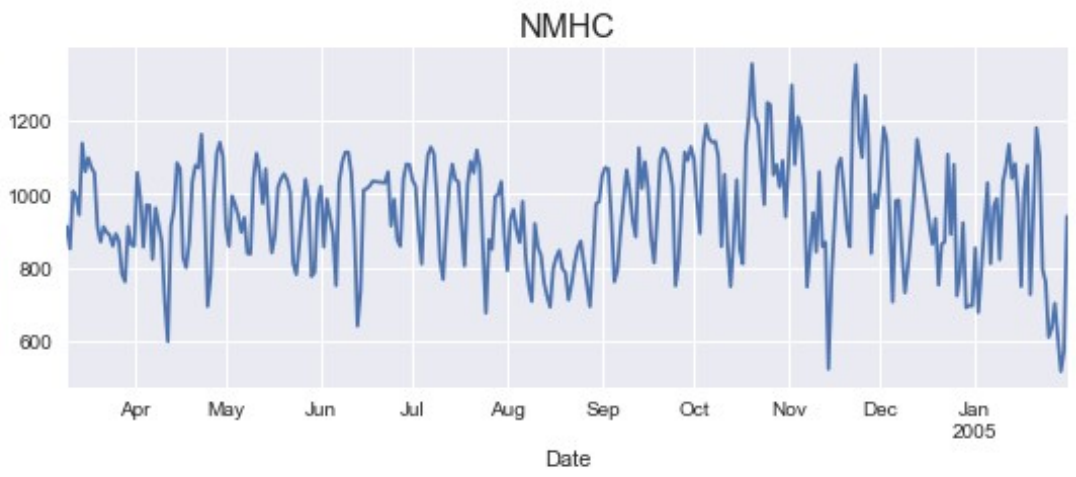

NOx
1500
1000
500
0
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

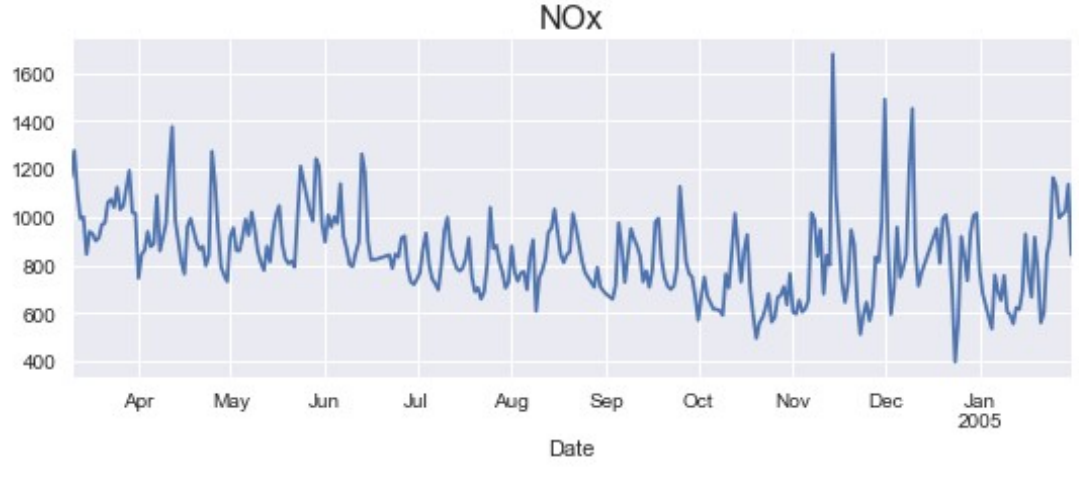

O3
1500
1000
500
0
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

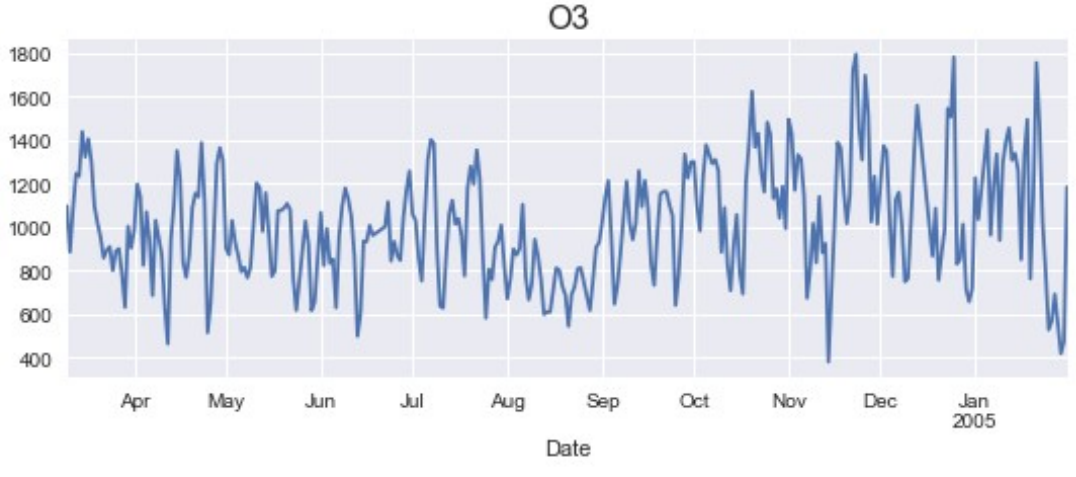

T
0
−50
−100
−150
−200
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

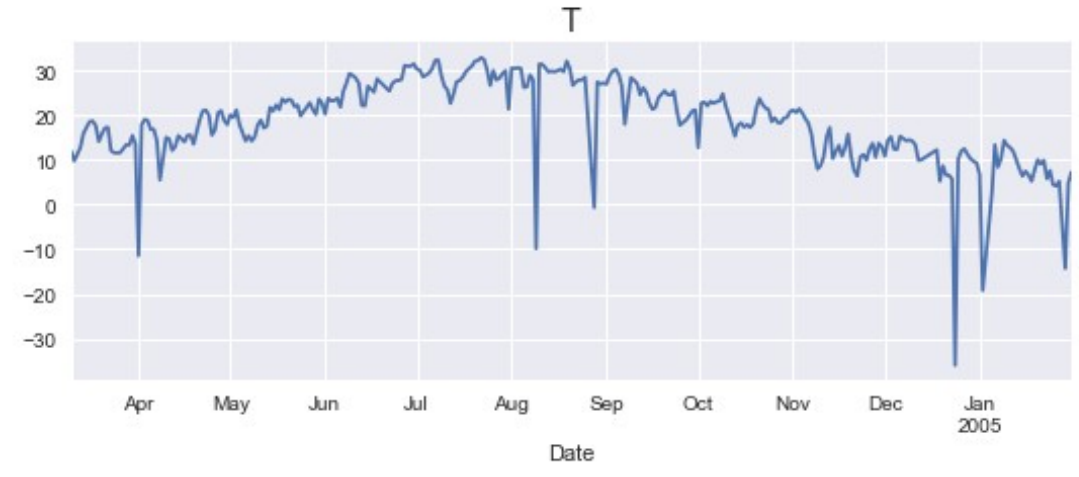

RH
50
0
−50
−100
−150
−200
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

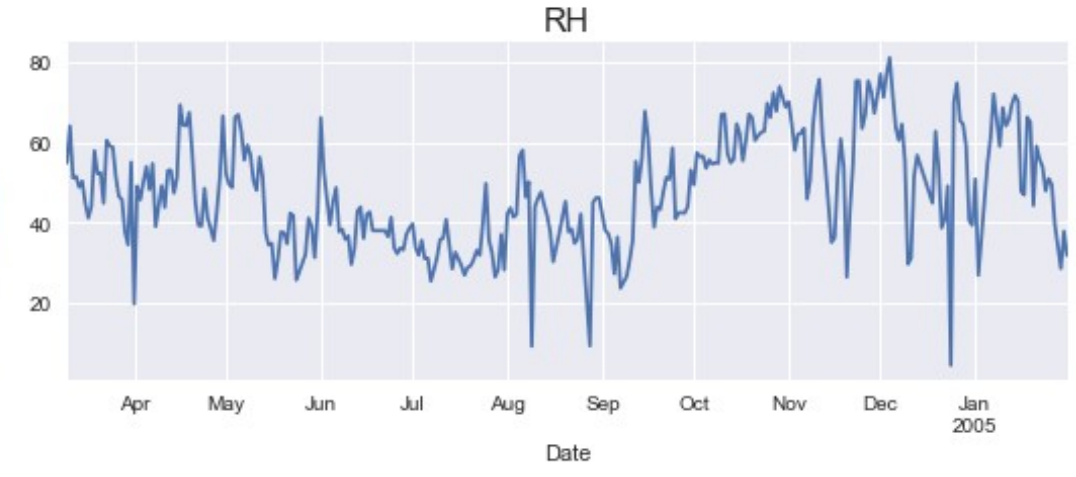

CO
1400
1200
1000
800
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date


NMHC
1200
1000
800
600
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

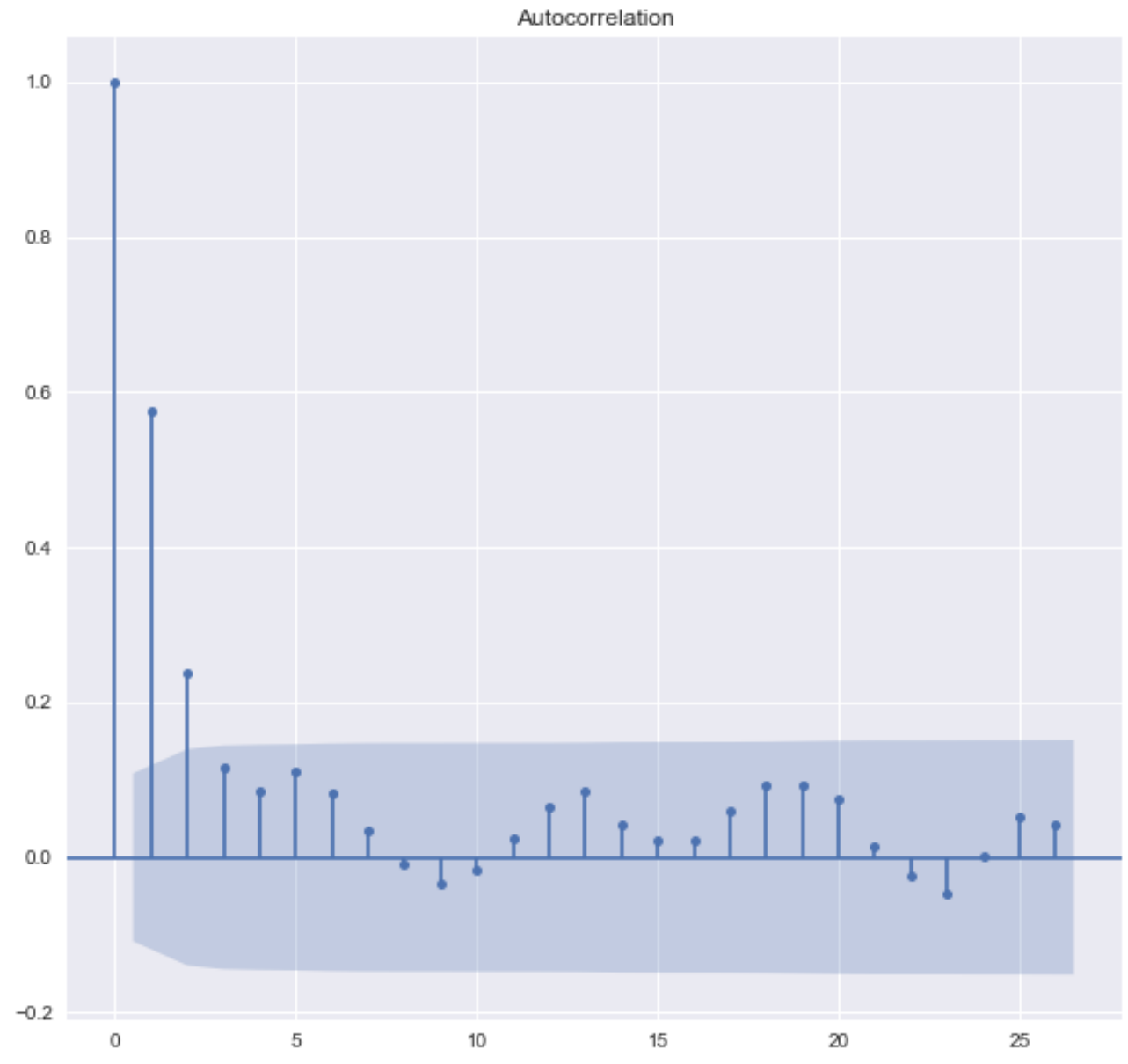

NOx
1600
1400
1200
1000
800
600
400
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

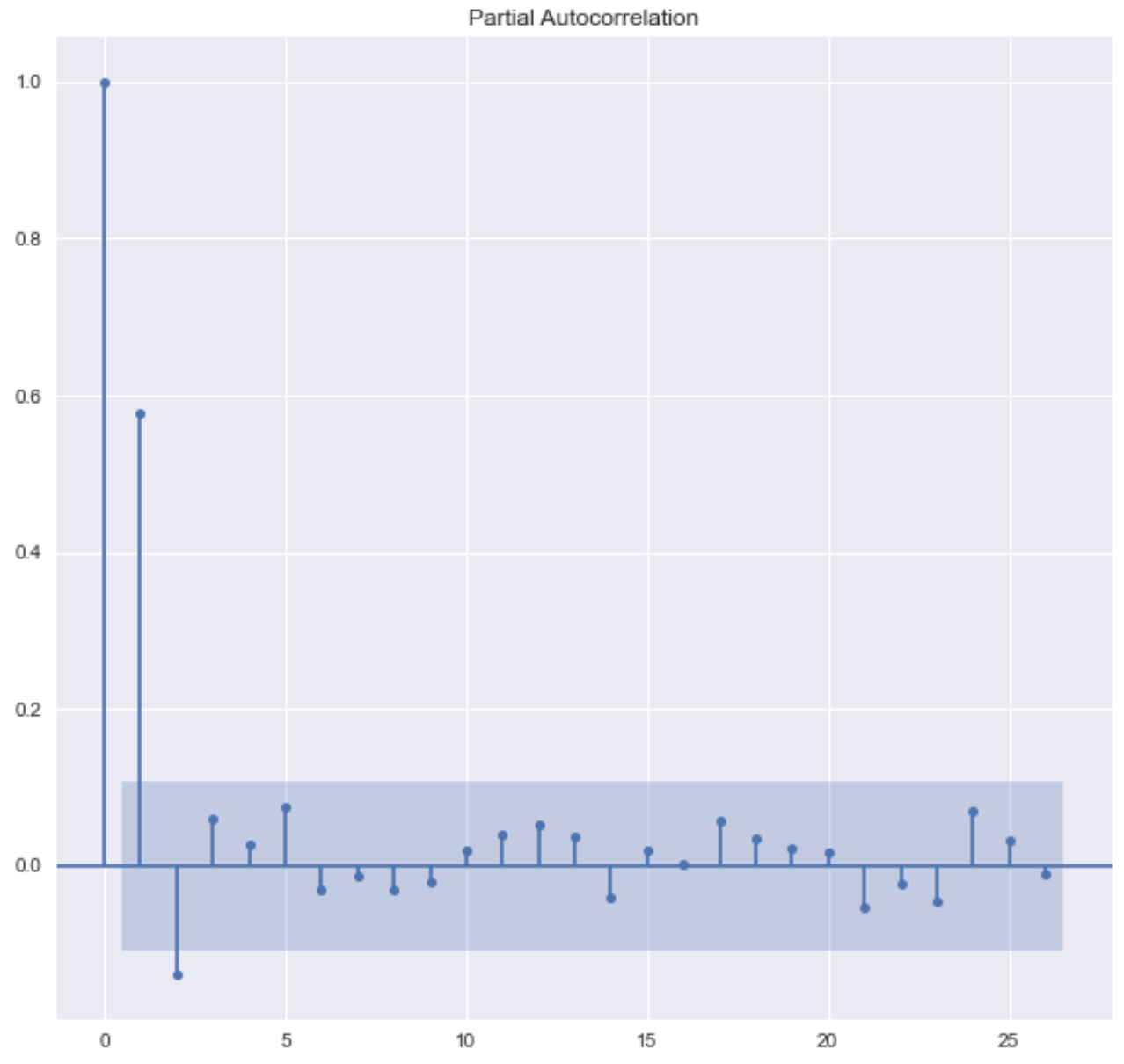

O3
1800
1600
1400
1200
1000
800
600
400
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date


T
30
20
10
0
−10
−20
−30
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date


RH
80
60
40
20
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2005
Date

CO


Autocorrelation
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
5
10
15
20
25

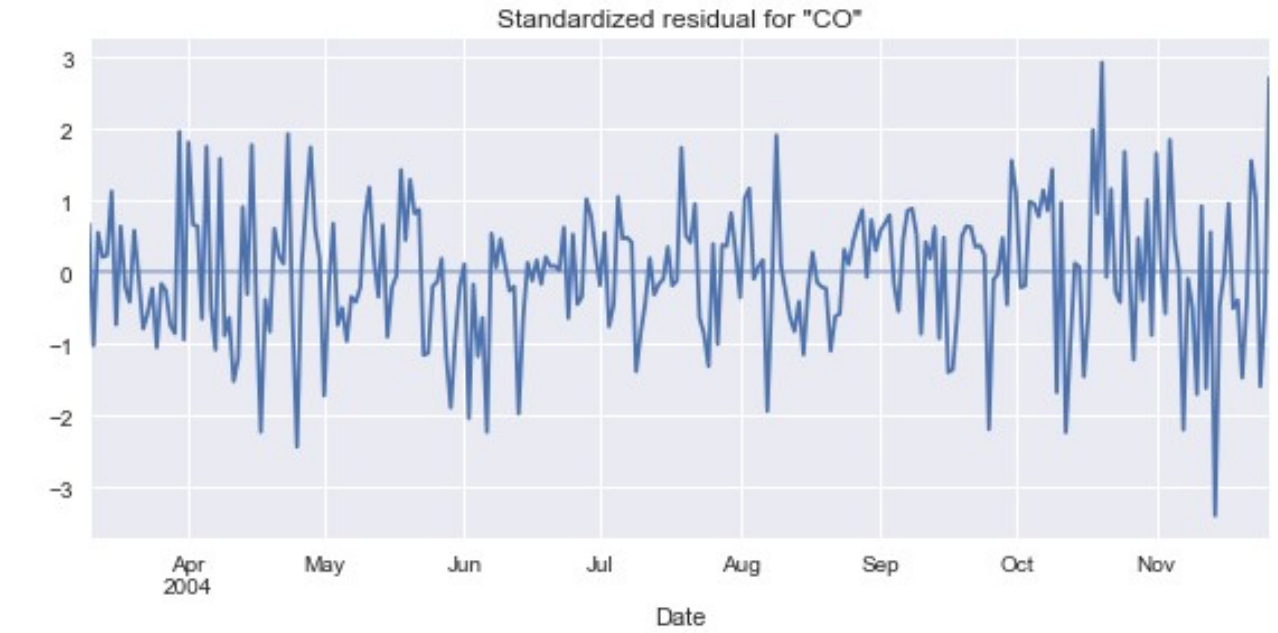

Partial Autocorrelation
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25

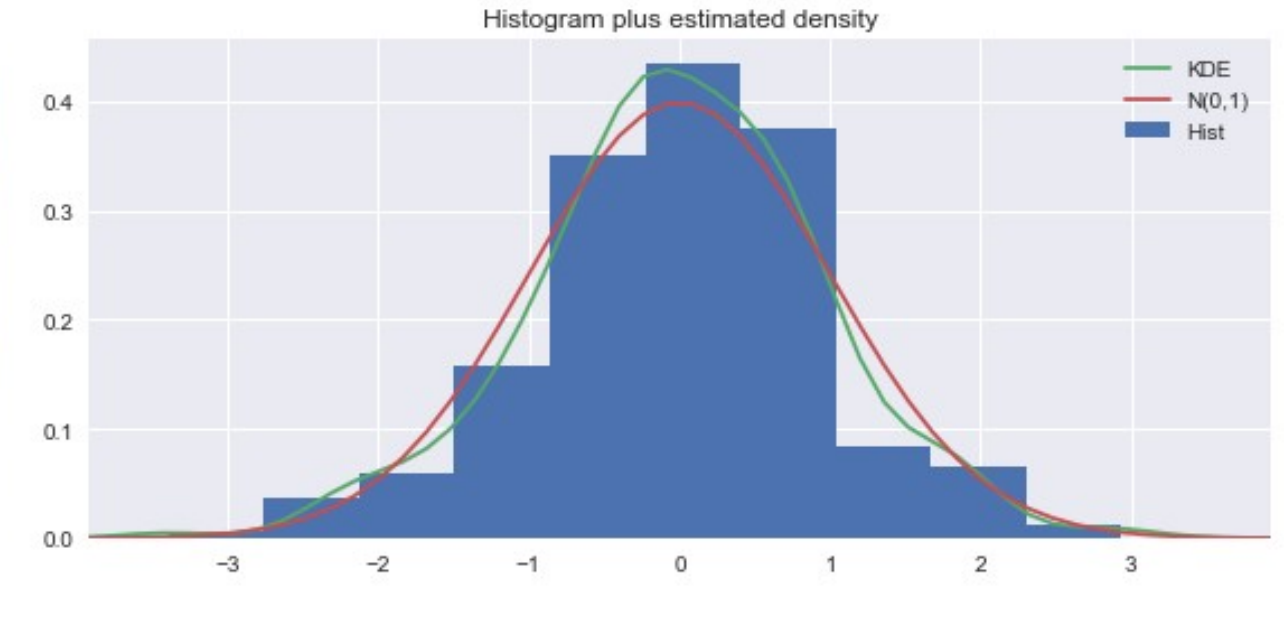
VARMAX

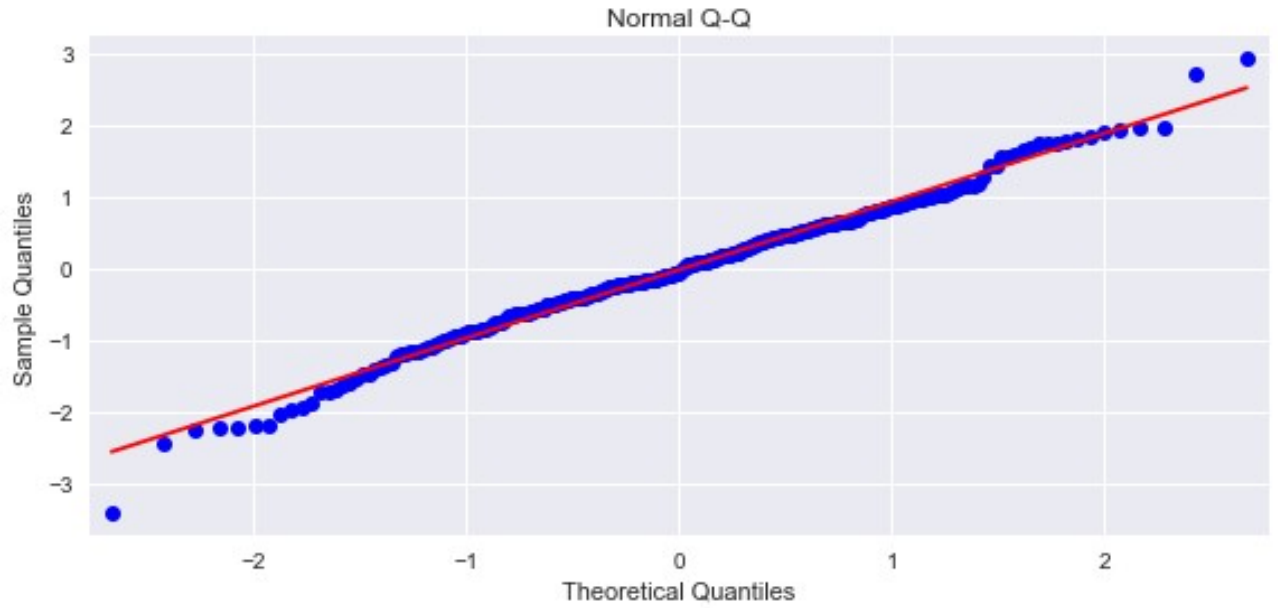
VARMAX: Vector Autoregressive Moving Average with eXogenous regressors model

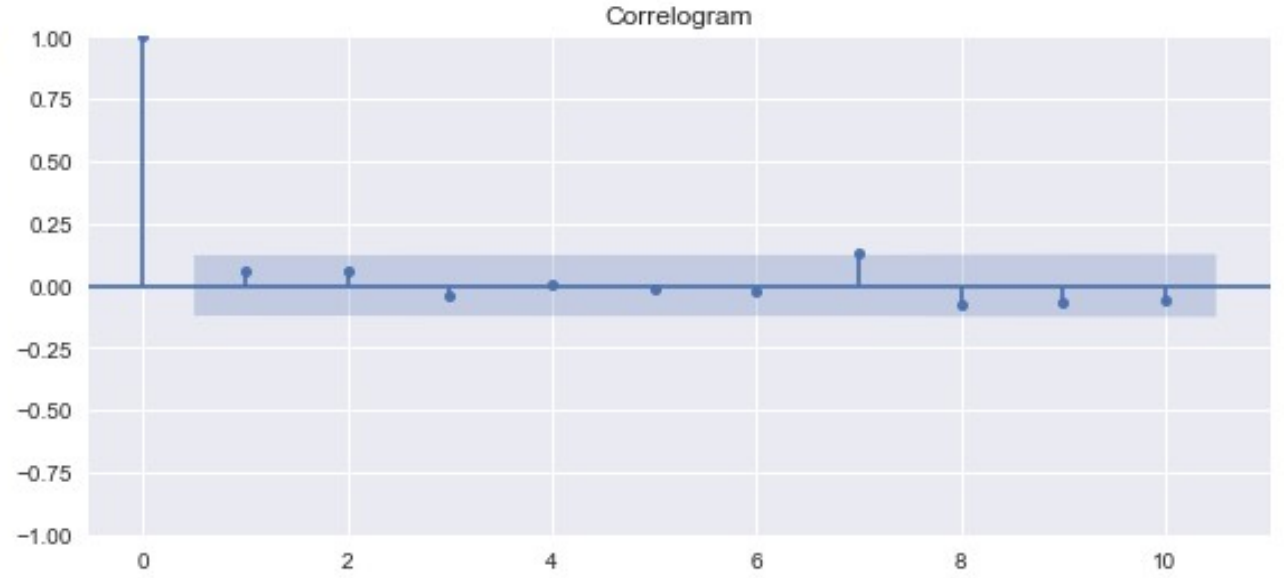
I chose to use this model as it is more suitable for this scenario, multivariate dependent/endogenous variables, with multiple independent/exogenous variables. By using VARMAX, we come to an assumption that at least two time series influence each other, as shown in Causality and Cointegration Test. NOx, Carbon Monoxide, and volatile organic compounds, react in the atmosphere in the presence of sunlight to produce tropospheric ozone

```
# training model
model = VARMAX(endog=y_clean_train, exog=X_clean_train, order=(1, 1)).fit(maxiter=1000, disp=False)
```


Standardized residual for "CO"
3
2
1
0
−1
−2
−3
Apr
2004
May
Jun
Jul
Aug
Sep
Oct
Nov
Date


Histogram plus estimated density
KDE
N(0,1)
Hist
0.4
0.3
0.2
0.1
0.0
−3
−2
−1
0
1
2
3


Normal Q-Q
Sample Quantiles
3
2
1
0
−1
−2
−3
−2
−1
0
1
2
Theoretical Quantiles

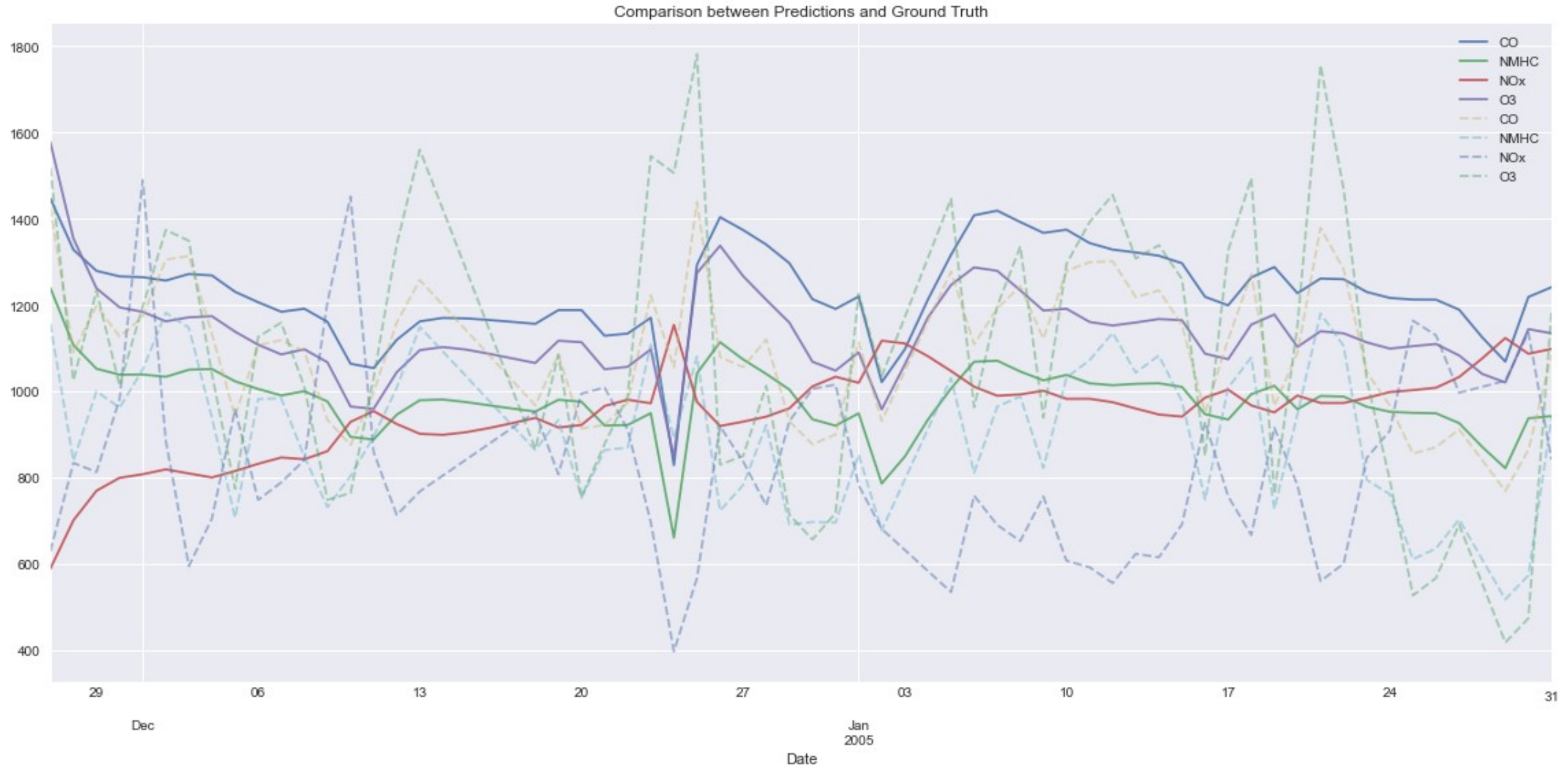

Correlogram
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

Hold-Out Validation

A simple split of the training data into 80% training set and 20% validation set to do an initial evaluation of the model performance.

Model seems to be overfitting slightly, but overall seems to be a good fit. RMSE 235.5 seems to be a good starting score comparative to the Standard Deviation 281.9.

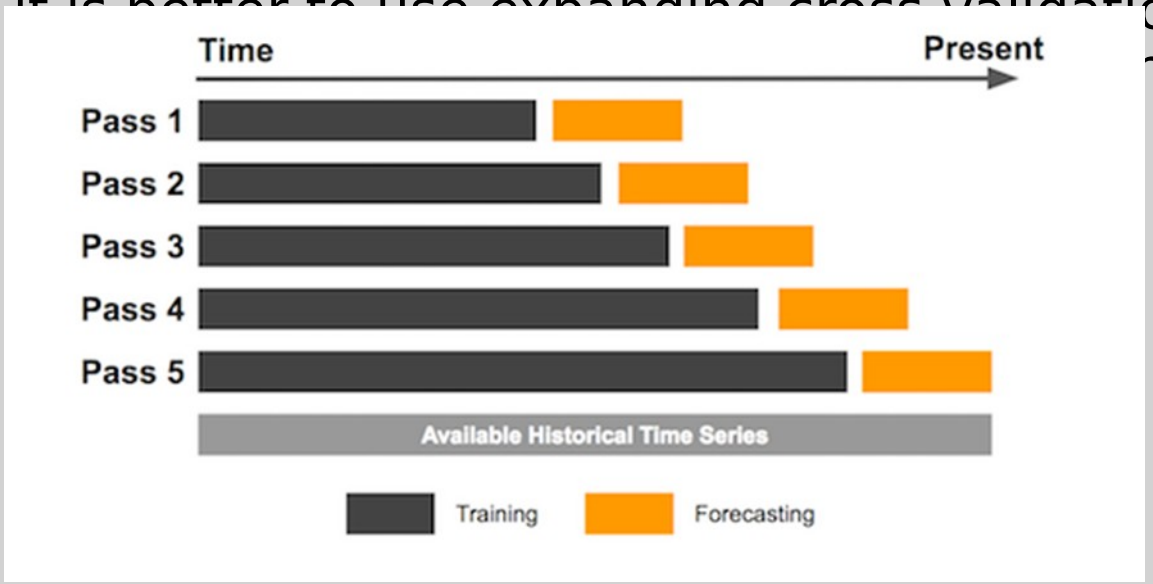
```
Train MAPE:      9.890%
Train RMSE:      117.538
Hold Out MAPE:   22.939%
Hold Out RMSE:   235.562
```


Comparison between Predictions and Ground Truth
1800
1600
1400
1200
1000
800
600
400
CO
NMHC
NOx
O3
CO
NMHC
NOx
O3
29
06
13
20
27
03
10
17
24
31
Dec
Jan
2005
Date

Expanding Cross Validation

In my opinion, we cannot use solely use the hold-out validation method, since are evaluating the model's performance based on a single hold-out set. As such, we need to use a more robust method to evaluate the model's performance.

Since order matters in time series analysis, we are unable to use k-fold cross validation as stratified sampling is no longer possible. For time-series data, it is better to use expanding cross validation method, since order matte re dataset.



Expanding Cross Validation

- training scores and validation scores are quite deviated apart
- clear indications of overfitting due to the large deviation between the training and validation scores

```
Wall time: 1min 29s

train_rmse          107.884641
train_rmse_std        6.695809
valid_rmse          187.189535
valid_rmse_std       46.275822
train_mape            0.091331
valid_mape            0.174605
AIC               12385.506963
BIC               12580.615880
Name: (1, 1), dtype: float64
```

Hyperparameter Tuning

The objective of hyperparameter tuning is to find the best set of parameters that minimize the mean squared error and reduce the deviation between the training score and validation score. Since VARMAX contains only two parameters, AR(p) and MA(q), we can use Grid Search Cross Validation to find the best set of parameters.

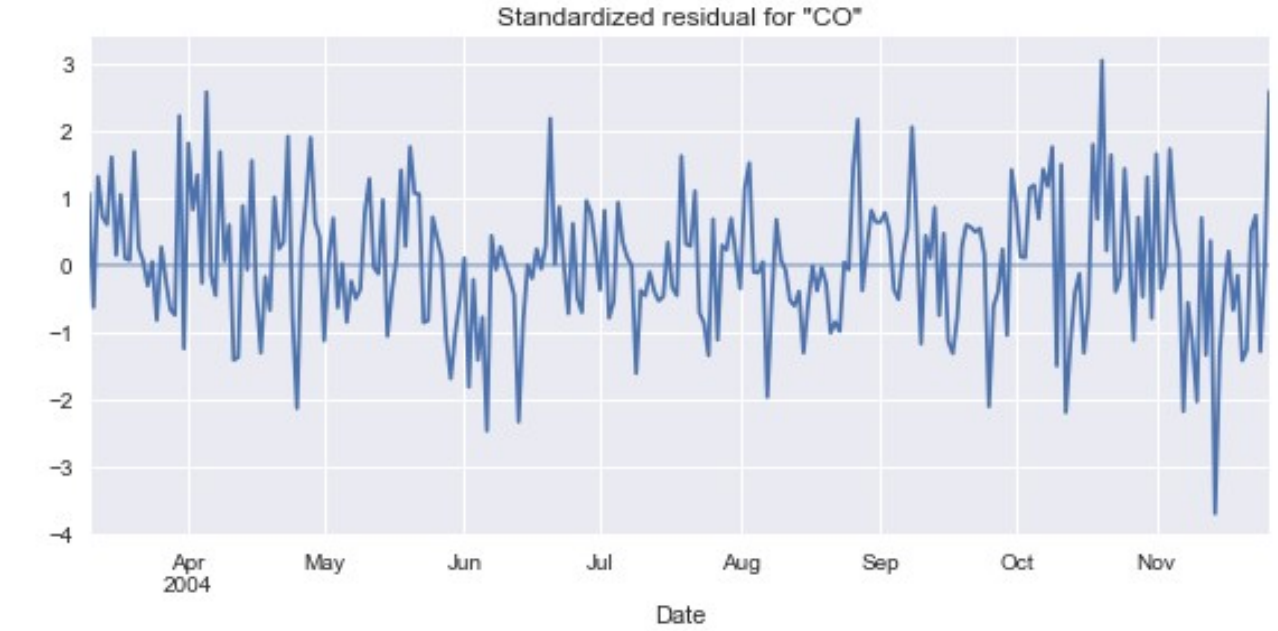
	train_rmse	train_rmse_std	valid_rmse	valid_rmse_std	train_mape	valid_mape	AIC	BIC
(0, 1)	115.640229	6.956601	190.136304	25.669501	0.099217	0.179808	12625.291864	12762.590732
(0, 2)	111.504013	7.166570	191.111898	22.965573	0.095222	0.179431	12540.742249	12735.851166
(1, 0)	114.340057	5.425806	196.396224	29.726744	0.096097	0.185558	12462.324565	12599.623433
(1, 1)	107.884641	6.695809	187.189535	46.275822	0.091331	0.174605	12385.506963	12580.615880
(1, 2)	107.123232	7.221657	186.028987	45.335757	0.091202	0.173185	12376.173957	12629.092925
(2, 0)	107.152312	6.883711	180.210028	42.138377	0.090554	0.166784	12458.044029	12653.152947
(2, 1)	106.035101	7.540197	190.586384	43.233530	0.090073	0.176741	12384.819640	12637.738607
(2, 2)	104.266112	7.748196	197.923456	47.830616	0.088403	0.181769	12400.161064	12710.890081

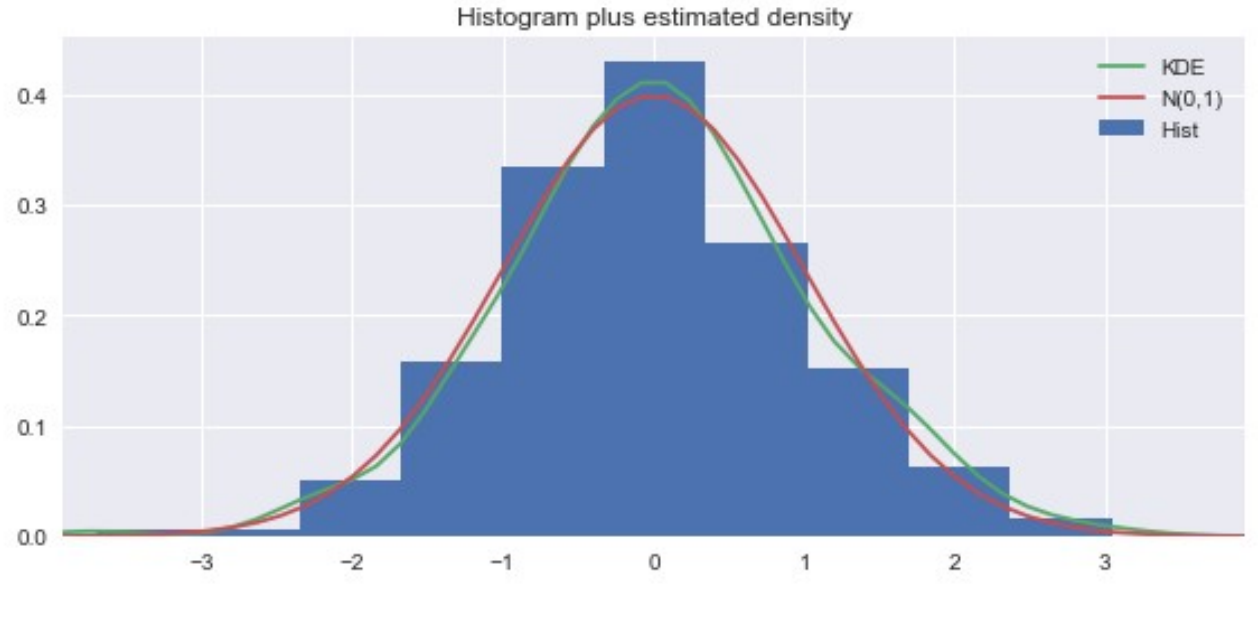
Question: Does removing extreme outliers achieves my desired results?

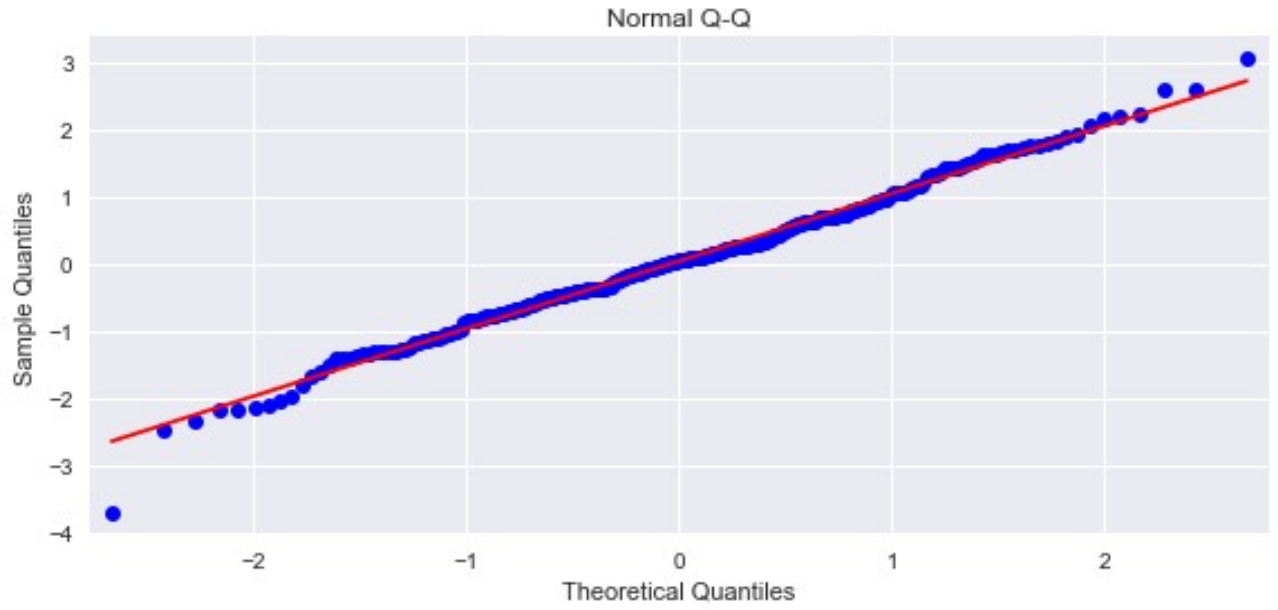
This is a question that came into my mind, after repetitive testing my baseline model in Kaggle. In this segment of the code, I implement back the raw data after pivoting and train my VARMAX model. In this case for Kaggle, I would want my model to be more robust in predicting extreme outliers, as such I would feed my model with the original raw data. However, in a production model I would remove outliers are I would want my forecasting model to be insensitive to outliers.

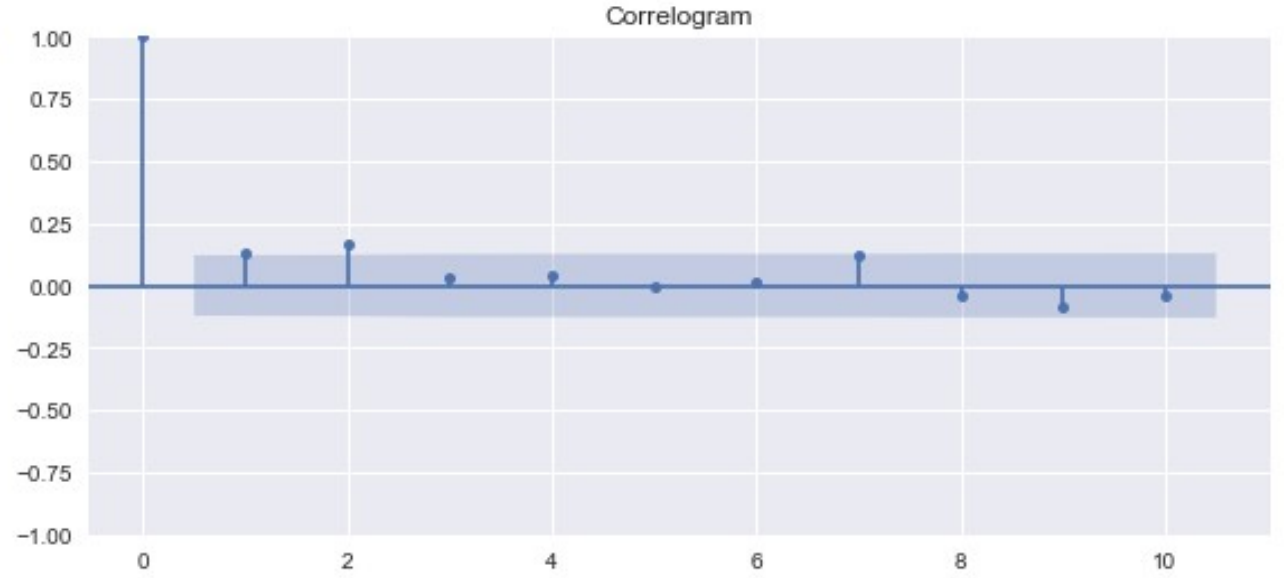
```
# training model
model = VARMAX(endog=y_train, exog=X_train, order=(1, 1)).fit(maxiter=1000, disp=False)
model.summary()
```

Question: Does removing

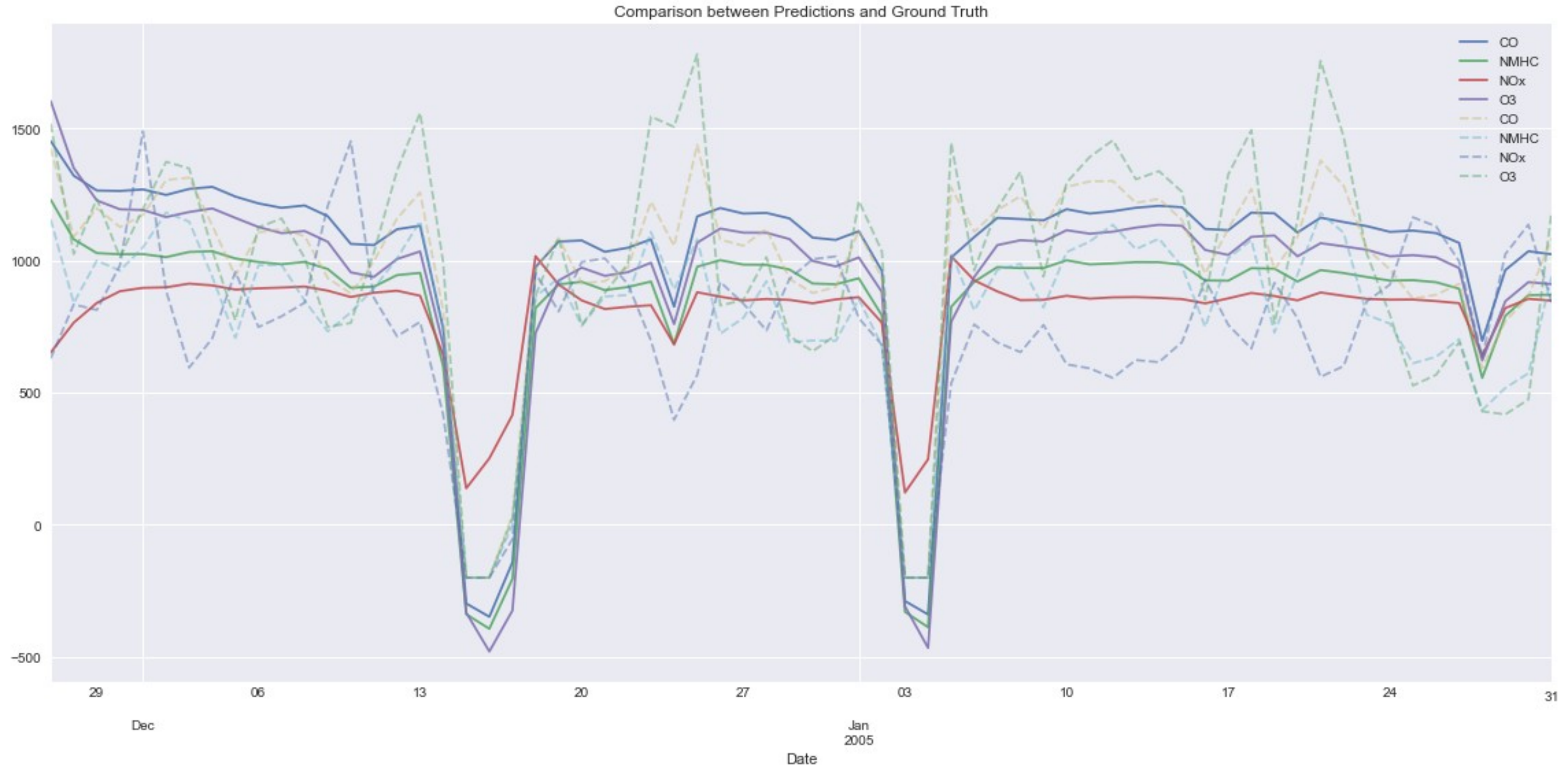








Question: Does removing



Hyperparameter Tuning

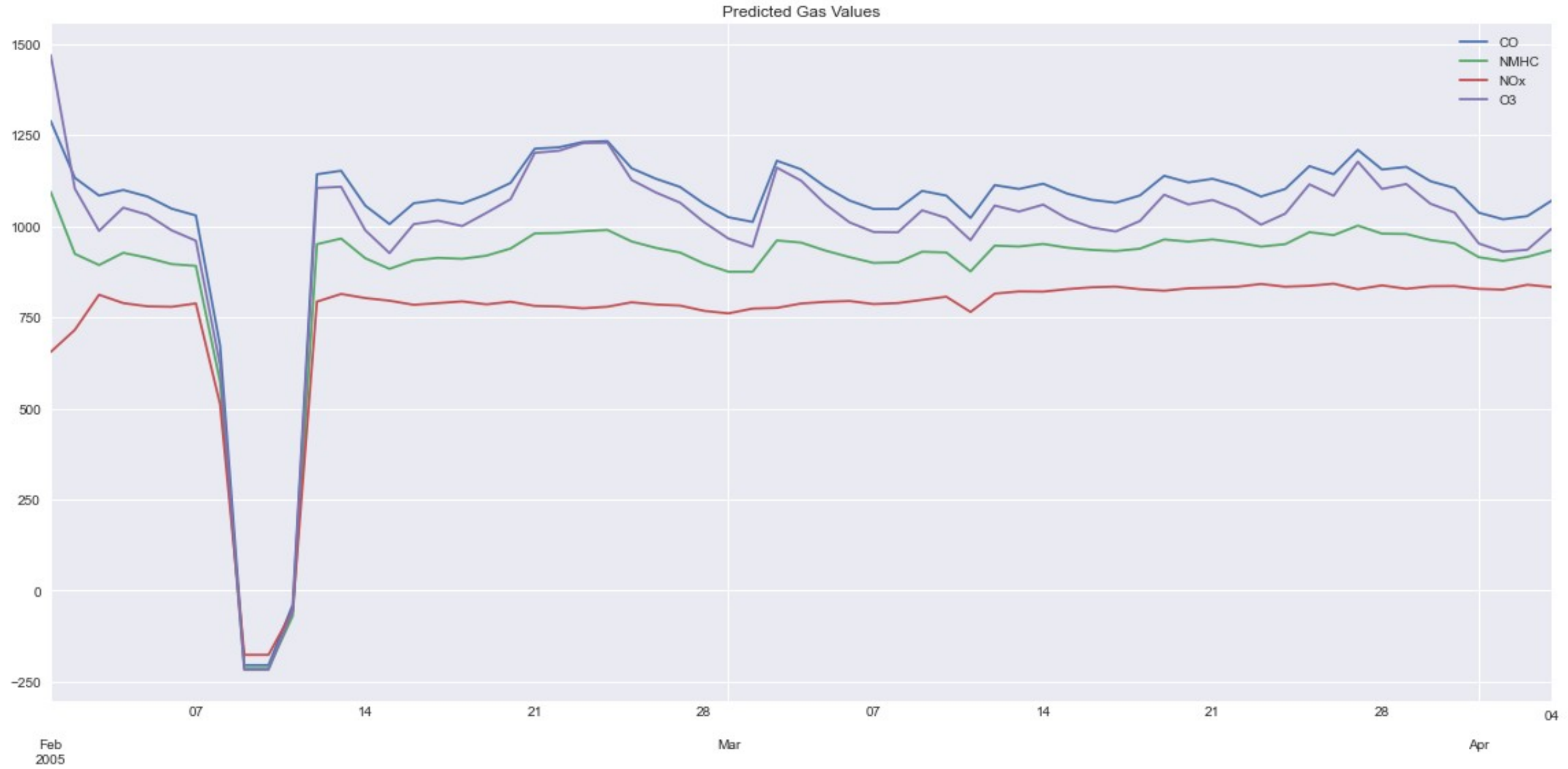
- (0, 2) is the best set of parameters for VARMAX
- p = 0 seems to be the best parameter for AR coefficients, as it produces a validation RMSE below 200 compared to the other AR parameters.

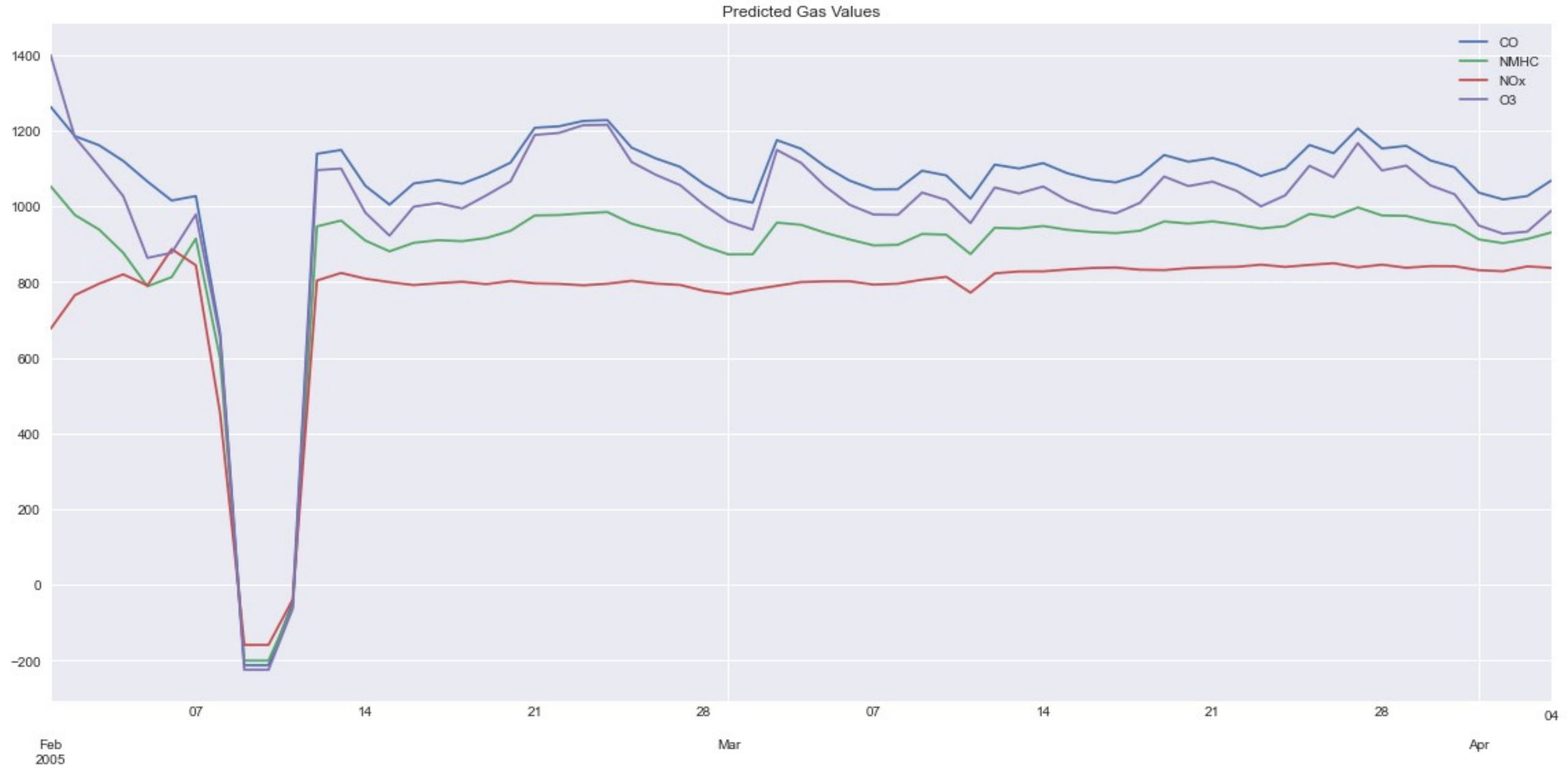
	train_rmse	train_rmse_std	valid_rmse	valid_rmse_std	train_mape	valid_mape	AIC	BIC
(0, 1)	115.922793	6.891766	186.434711	26.239974	0.118130	0.313302	12642.072749	12779.371617
(0, 2)	112.833710	6.119358	185.357896	26.905121	0.112244	0.322518	12540.783076	12735.891994
(1, 0)	171.883921	13.276003	528.276537	325.865974	0.205843	2.813995	13480.969501	13618.268369
(1, 1)	141.835128	25.292466	212.776692	36.149826	0.224952	1.075762	12597.730114	12792.839032
(1, 2)	136.879346	27.162293	231.658268	40.434437	0.236034	1.131922	12538.779192	12791.698159
(2, 0)	147.844146	6.748860	481.428832	420.311342	0.184547	2.967160	13433.087456	13628.196374
(2, 1)	128.375960	19.691943	298.663558	109.661485	0.253389	2.768230	12533.930564	12786.849531
(2, 2)	120.342353	5.632663	249.221535	23.794896	0.225318	2.164926	12538.337935	12849.066952

Hyperparameter Tuning

- (0, 3) has the best cross – validated root mean square error results
- (0, 8) also seem to have good results, having a balance between low AIC, validation RMSE, validation RMSE and MAPE

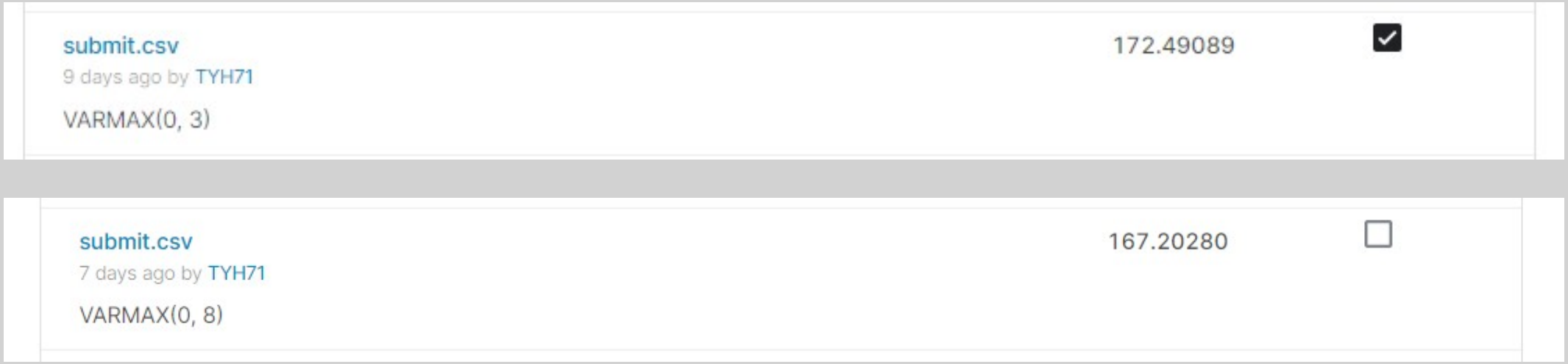
	train_rmse	train_rmse_std	valid_rmse	valid_rmse_std	train_mape	valid_mape	AIC	BIC
(0, 1)	115.922793	6.891766	186.434711	26.239974	0.118130	0.313302	12642.072749	12779.371617
(0, 2)	112.833710	6.119358	185.357896	26.905121	0.112244	0.322518	12540.783076	12735.891994
(0, 3)	108.767626	6.658620	182.970910	27.645803	0.106363	0.327329	12516.270384	12769.189351
(0, 4)	106.531823	8.251328	183.534671	29.028084	0.101082	0.335134	12526.759685	12837.488703
(0, 5)	106.715673	7.483262	186.042743	29.127180	0.123351	0.530512	12513.694072	12882.233138
(0, 6)	104.613802	7.020979	185.422080	27.341014	0.105626	0.374152	12523.552868	12949.901985
(0, 7)	101.469269	7.793975	184.073765	27.970350	0.107532	0.384039	12479.916673	12964.075840
(0, 8)	100.668406	7.485469	186.340161	28.726883	0.095809	0.339951	12503.964134	13045.933350
(0, 9)	99.928545	7.627810	186.294336	27.920344	0.115662	0.412191	12522.588043	13122.367308
(0, 10)	100.740646	8.305777	186.624318	27.415289	0.106272	0.409003	12578.976481	13236.565796


Predicted Gas Values
CO
NMHC
NOx
O3
1500
1250
1000
750
500
250
0
−250
07
14
21
28
07
14
21
28
04
Feb
2005
Mar
Apr


Predicted Gas Values
CO
NMHC
NOx
O3
1400
1200
1000
800
600
400
200
0
−200
07
14
21
28
07
14
21
28
04
Feb
2005
Mar
Apr

Kaggle Prediction

Among the attempts, it seems like VARMAX(0, 8) is the most robust model on Kaggle Public Leader, 40% of the testing data.



Conclusion

Even though I may not have enough experience to perform well in time series forecasting, I think this is a good first attempt on trying out this sub domain of machine learning. Here, I manage to get a multivariate model to forecast air pollution in Italy based on raw data. I think from this experience I learnt a lot about time series analysis, and I would like to apply this knowledge to other problems in the future.

Part B: Customer Segmentation

Segmentation Task

Given a mall customer dataset containing customers' attributes, the task is to perform clustering on the dataset to provide relevant insights to the Mall Owners to start their marketing strategy.

- How to achieve customer segmentation using unsupervised learning machine learning algorithm in Python?
- Who are your target customer with whom you can start marketing strategy?

Metadata

Attribute	Description	Type
CustomerID	Customer identifier key	Discrete
Gender	Customer’s gender	Categorical - Nominal
Age	Customer’s Age	Numerical – Continuous
Annual Income (k$)	Customer’s annual income denominated in thousands	Numerical – Continuous
Spending Score (1-100)	Spending score between 1 to 100	**Numerical – Continuous**

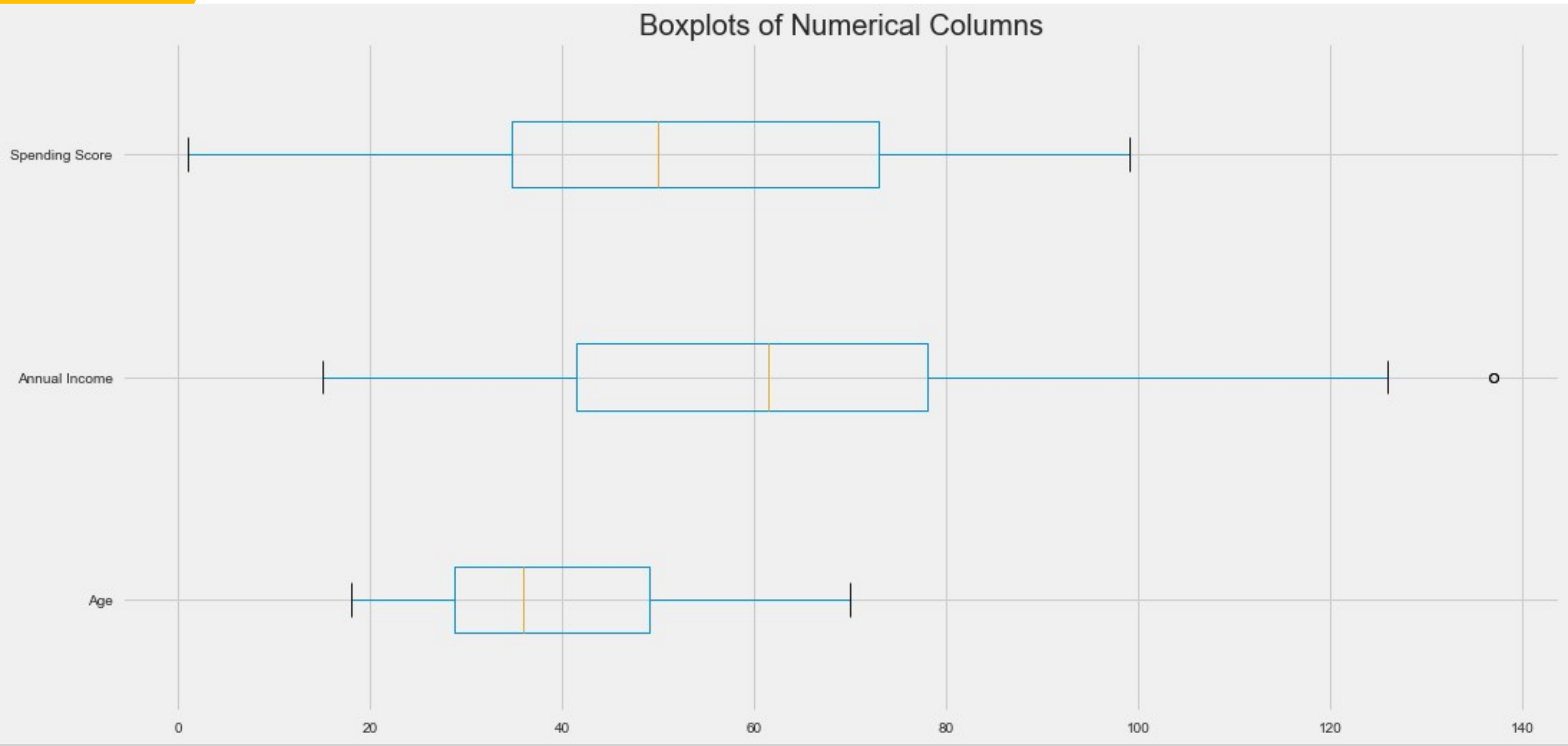
CustomerID	Gender	Age	Annual Income	Spending Score
1	Male	19	15	39
2	Male	21	15	81
3	Female	20	16	6
4	Female	23	16	77
5	Female	31	17	40

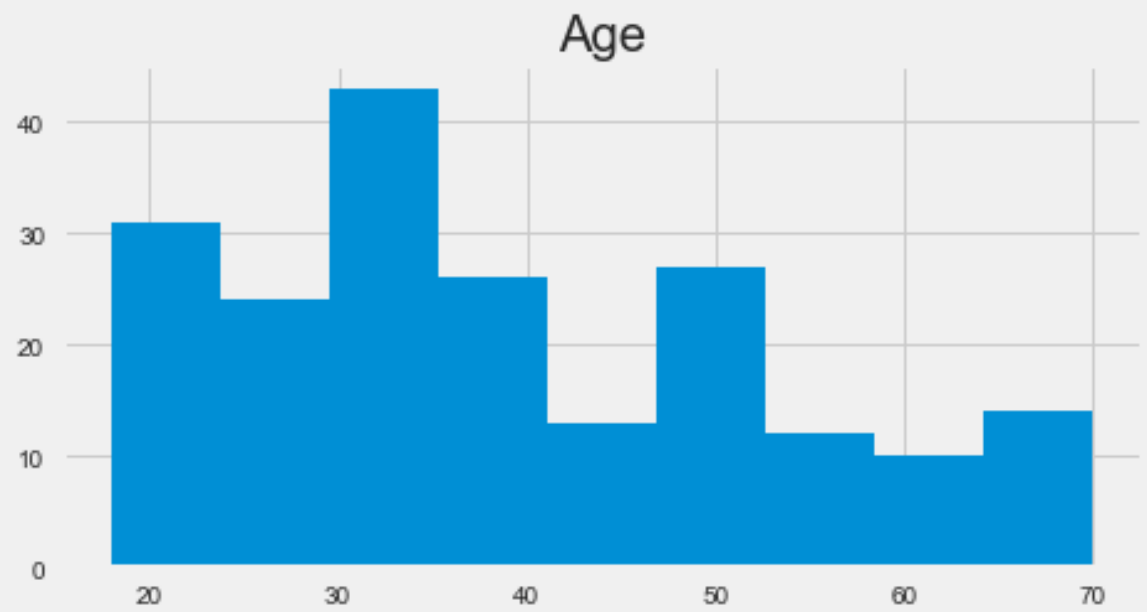
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 200 entries, 1 to 200
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Gender          200 non-null    object
 1   Age             200 non-null    int64
 2   Annual Income   200 non-null    int64
 3   Spending Score  200 non-null    int64
dtypes: int64(3), object(1)
memory usage: 7.8+ KB
```

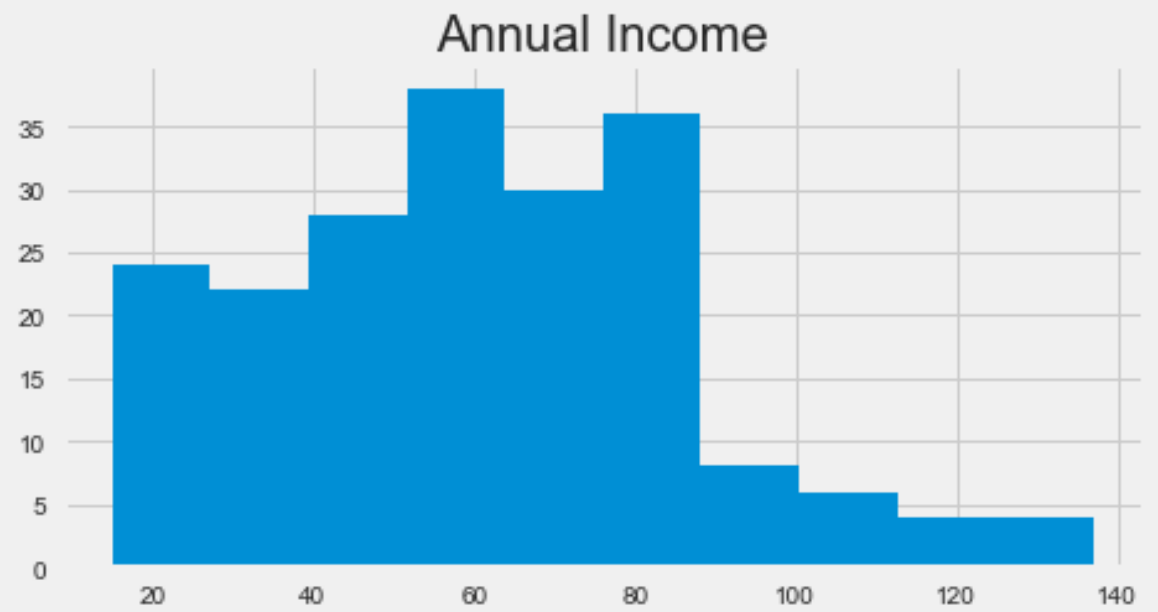
Exploratory Data Analysis

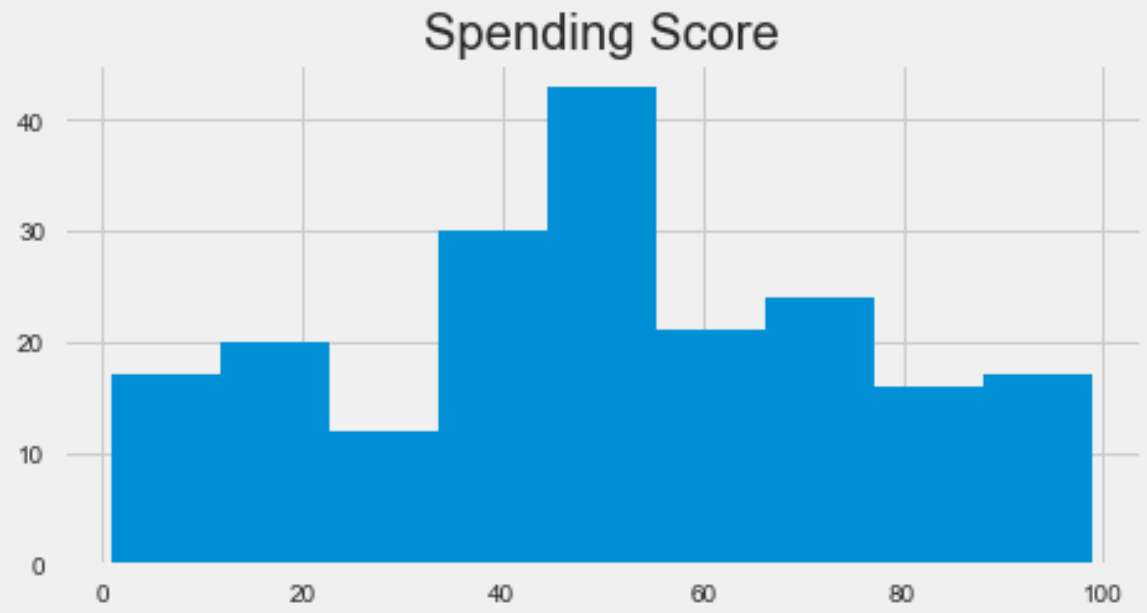
Topics I want to cover:

- Univariate Analysis
- Bivariate Analysis
- Multivariate Analysis
- Inferential Statistics

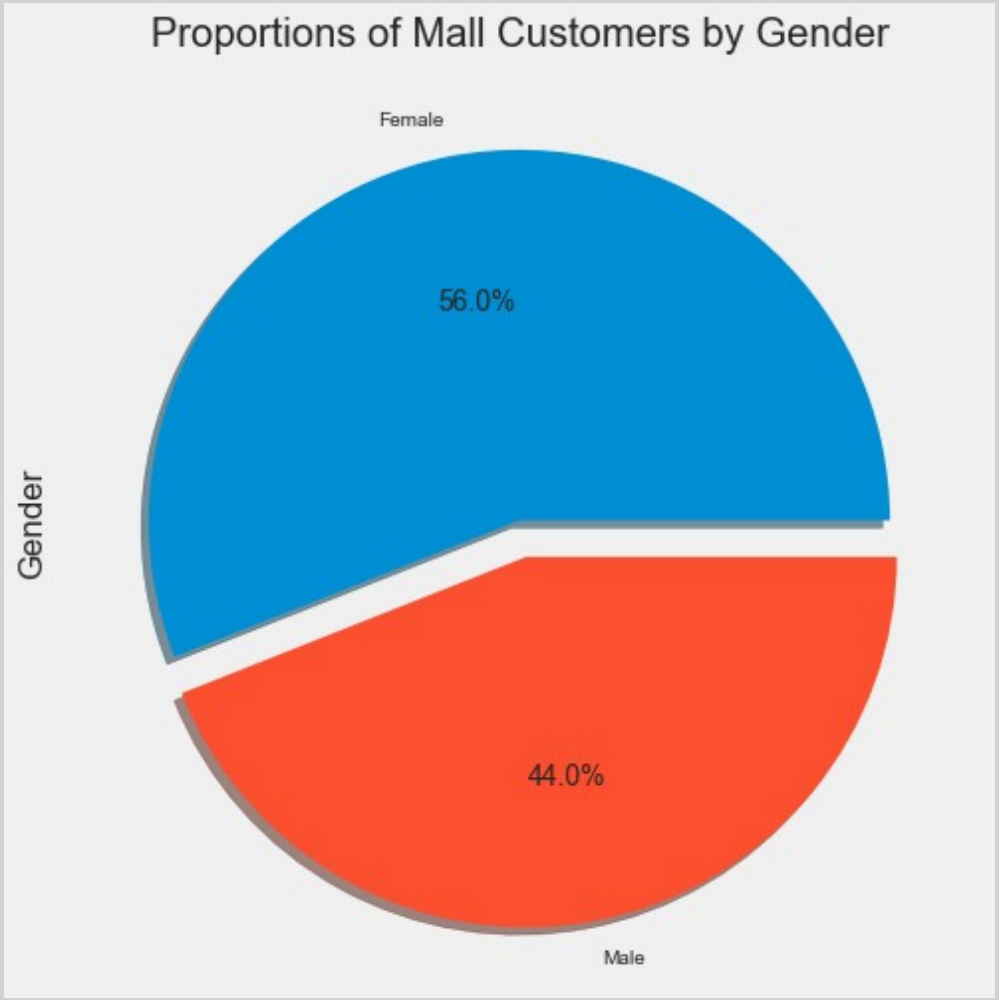

Boxplots of Numerical Columns
Spending Score
Annual Income
Age
0
20
40
60
80
100
120
140

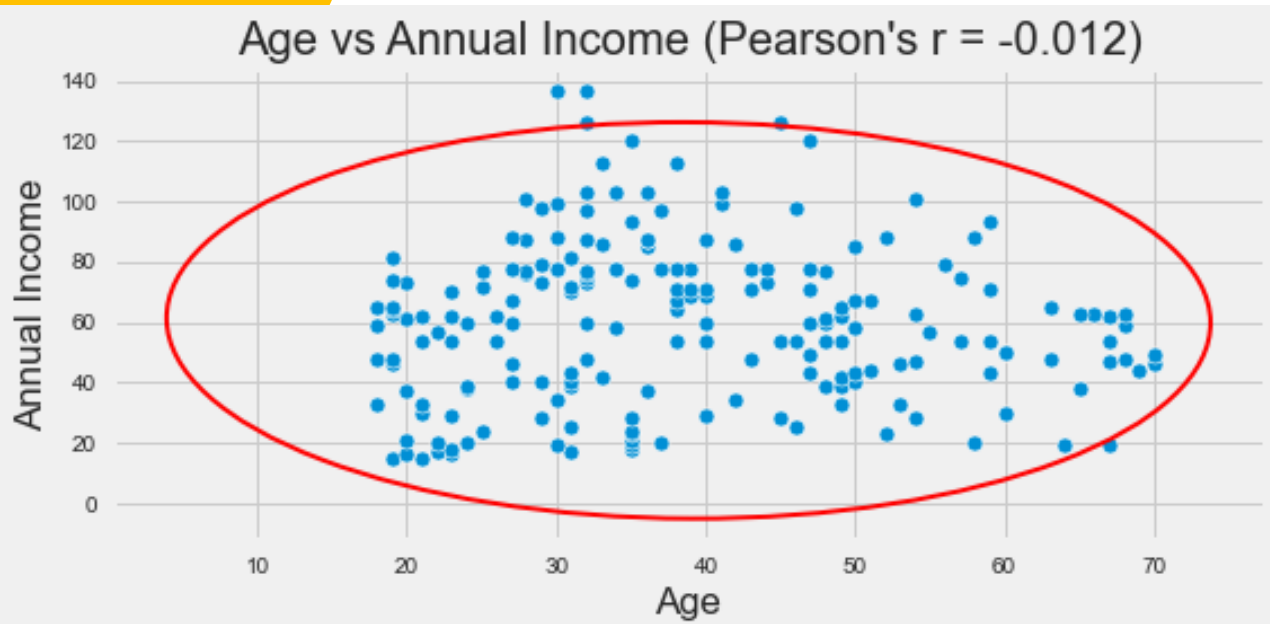

Age
40
30
20
10
0
20
30
40
50
60
70

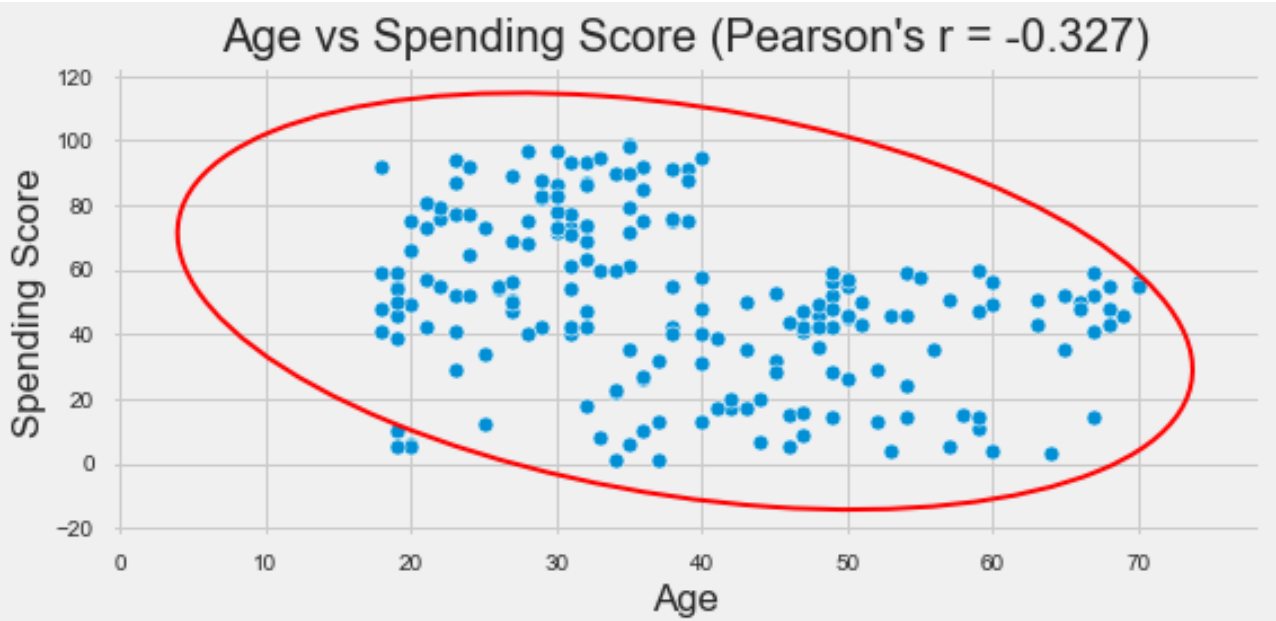

Annual Income
35
30
25
20
15
10
5
0
20
40
60
80
100
120
140

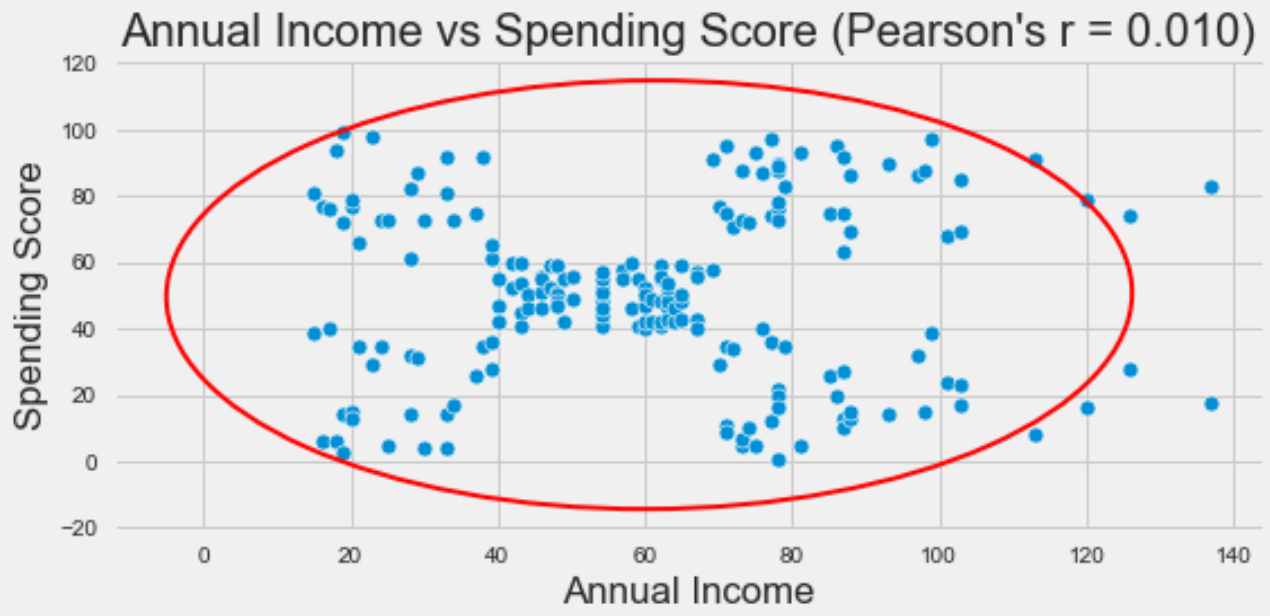

Spending Score
40
30
20
10
0
0
20
40
60
80
100

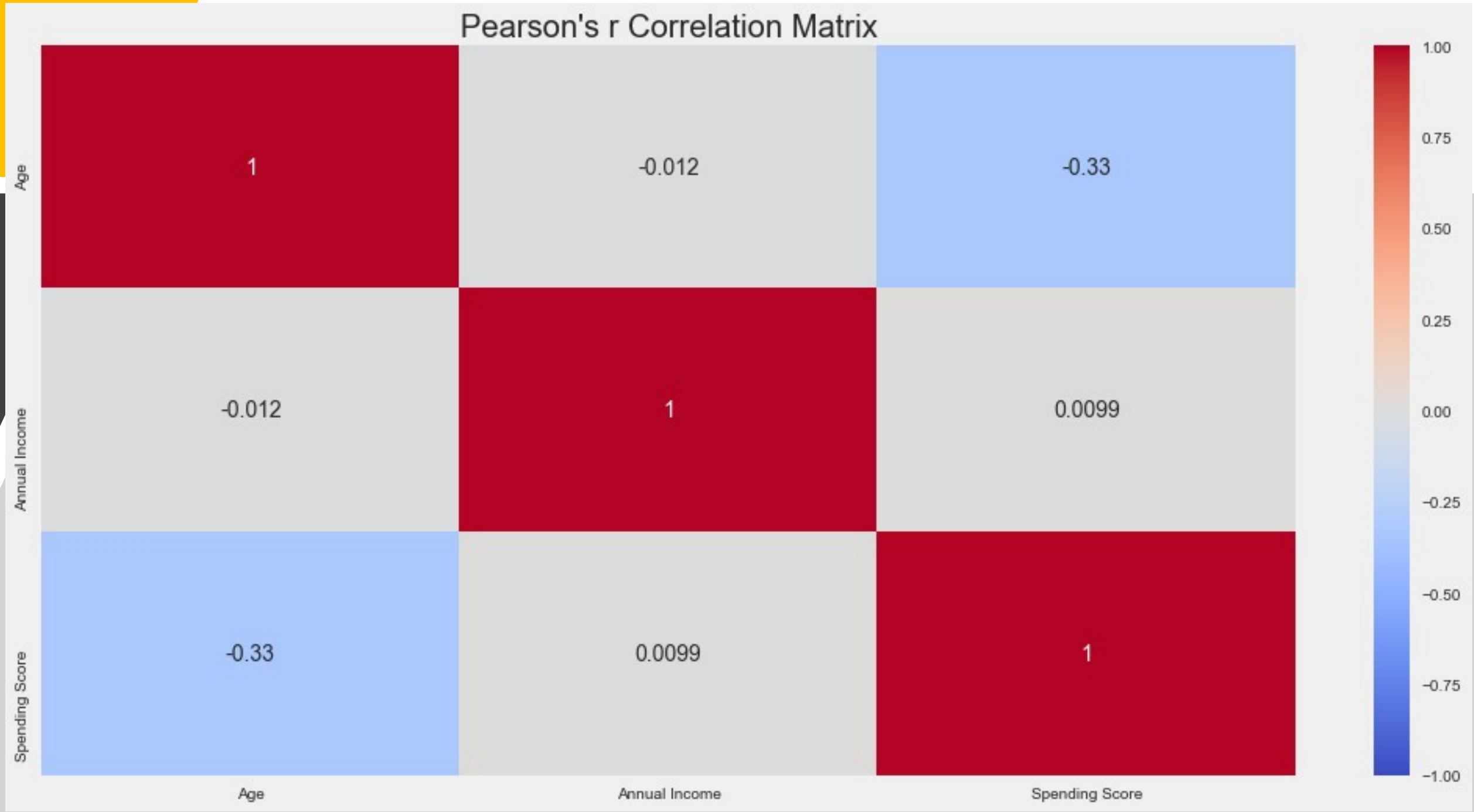
Univariate Analysis

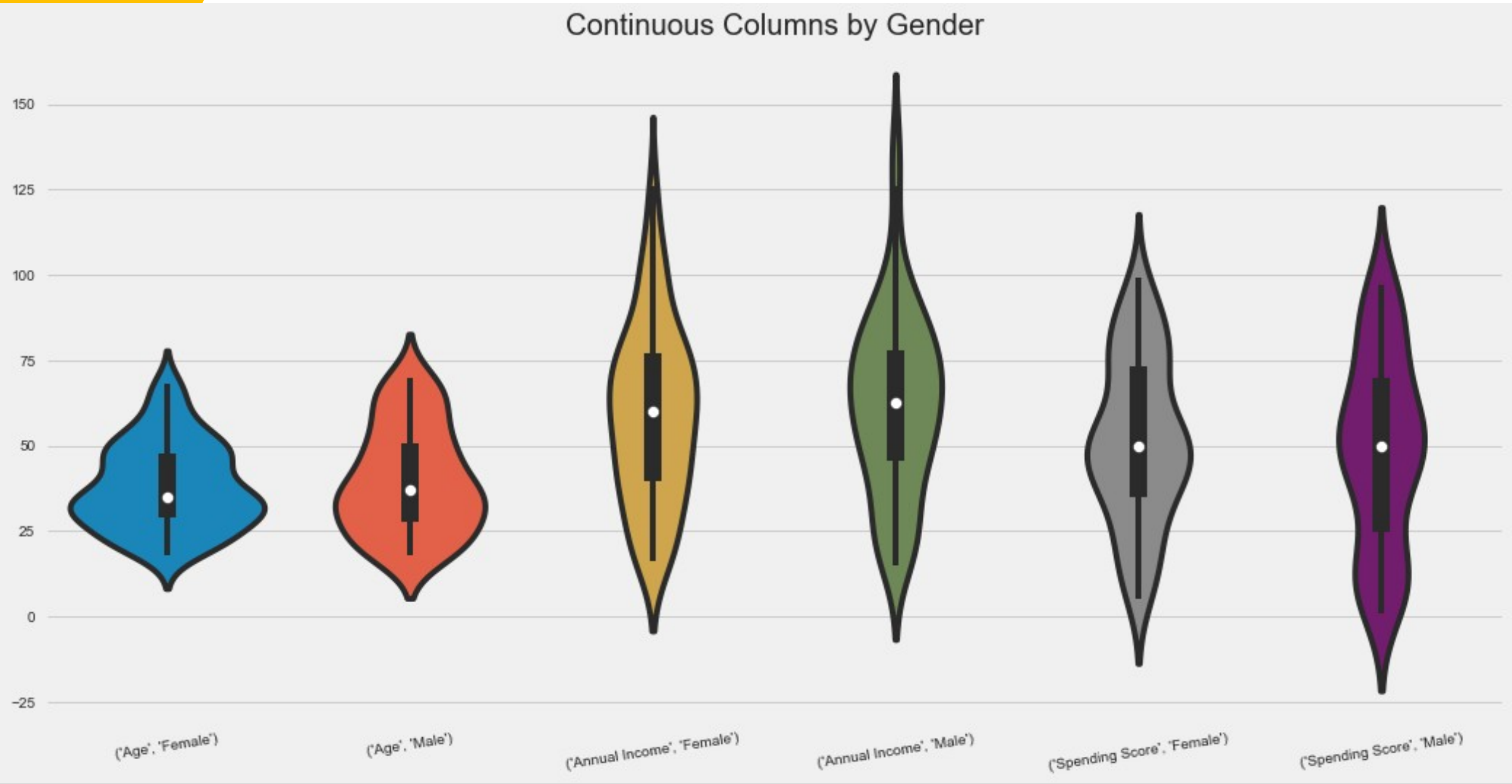


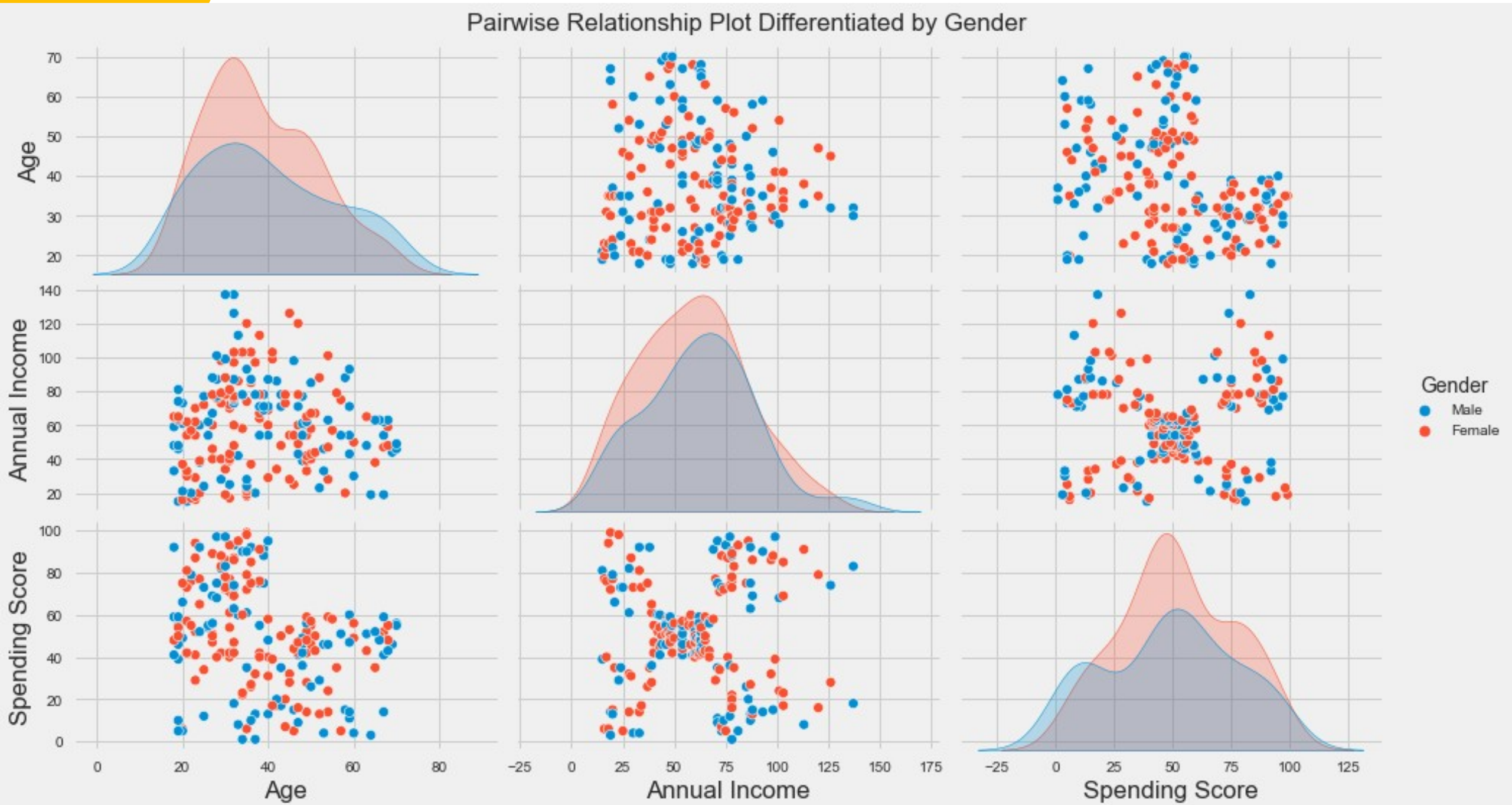

Age vs Annual Income (Pearson's r = -0.012)
Annual Income
Age
0
20
40
60
80
100
120
140
10
20
30
40
50
60
70


Age vs Spending Score (Pearson's r = -0.327)
Spending Score
Age
−20
0
20
40
60
80
100
120
0
10
20
30
40
50
60
70


Annual Income vs Spending Score (Pearson's r = 0.010)
Spending Score
Annual Income
−20
0
20
40
60
80
100
120
0
20
40
60
80
100
120
140


Pearson's r Correlation Matrix
Age
Annual Income
Spending Score
1
-0.012
-0.33
-0.012
1
0.0099
-0.33
0.0099
1
Age
Annual Income
Spending Score
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Continuous Columns by Gender
150
125
100
75
50
25
0
−25
('Age', 'Female')
('Age', 'Male')
('Annual Income', 'Female')
('Annual Income', 'Male')
('Spending Score', 'Female')
('Spending Score', 'Male')


Pairwise Relationship Plot Differentiated by Gender
Age
Annual Income
Spending Score
Gender
Male
Female

Is Gender necessary as a feature?

Based on previous analysis, there are doubts casted on whether Gender being useful as a feature to provide information to the algorithm for customer segmentation.

As such, I will be using an independent sample t-test to test whether there is any statistical difference between the Male group and the Female group.

Testing for Variance

The independent t-test assumes that the two variables, Male and Female, have similar variance, or else it will perform another type of t-test. I will be using the Levene's test to test for equal variance, at $\alpha = 0.05$.

	Age	Annual Income	Spending Score
Statistics	4.396601	0.009383	2.342619
P-Value	0.037281	0.922933	0.127474

Reject null hypothesis for *Age*, due to P-Value falling in the critical region. Not enough evidence to reject null hypothesis for *Annual Income* and *Spending Score*, as p-value falls outside the critical region.

Independent t-test

To determine whether there is a statistical significant difference between the means in two unrelated groups

	Age	Annual Income	Spending Score
Statistics	0.837433	0.795022	-0.819046
P-Value	0.403553	0.427552	0.413745

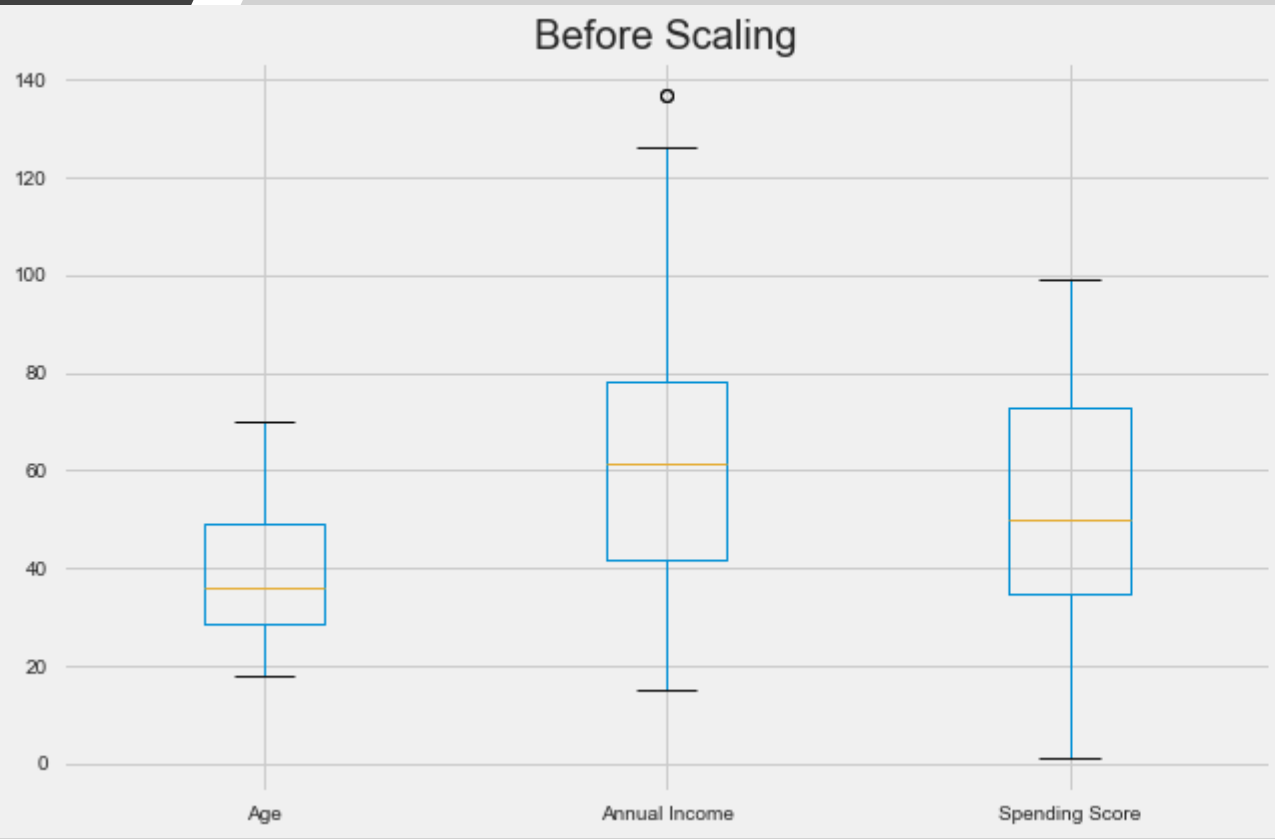
The P-Value falls above the significance level, α = 0.05, we cannot reject the null hypothesis. As such, we can deem male and female to be statistically indifferent. Since gender does not bring in any meaningful information, we drop Gender.

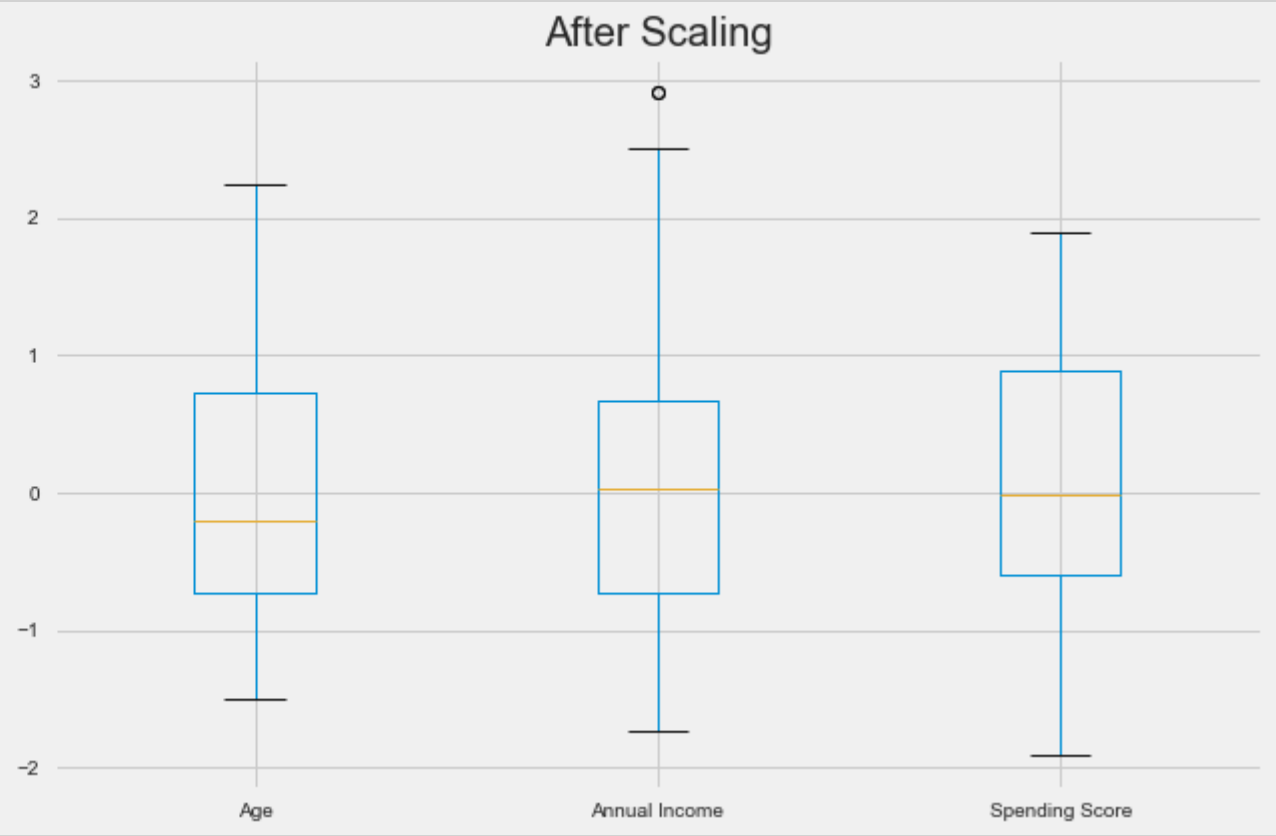
Z-Score Standardization

Typical Clustering Models are distance-based algorithm, in order to measure similarities and form clusters using distance metrics. By standardization, there would be no statistical bias between features within the learning algorithm. Therefore, standardization is required before building a clustering model.

In this case, I would be using Z-Score Standardization to perform feature scaling. This is to ensure that all of the features would have the mean and variance, following N(0, 1).

Z-Score Standardization

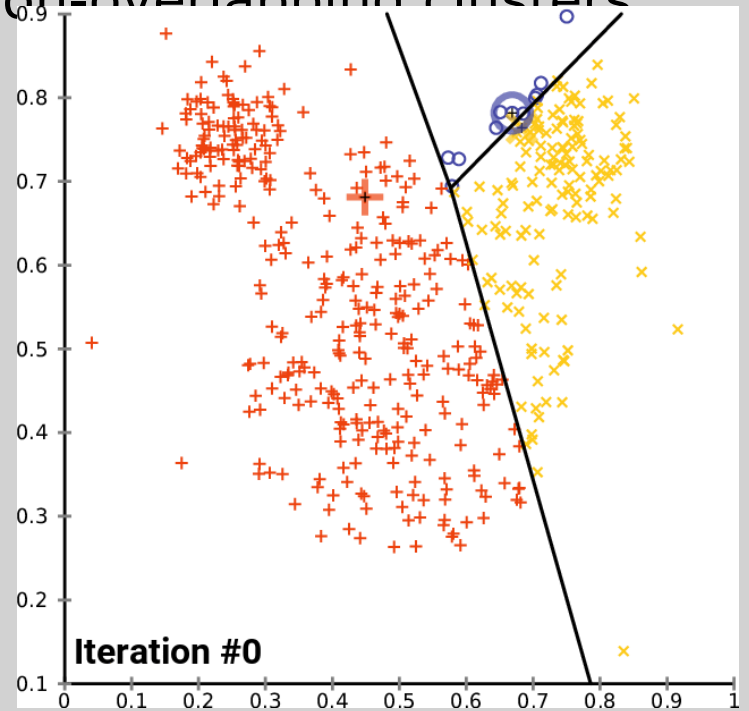


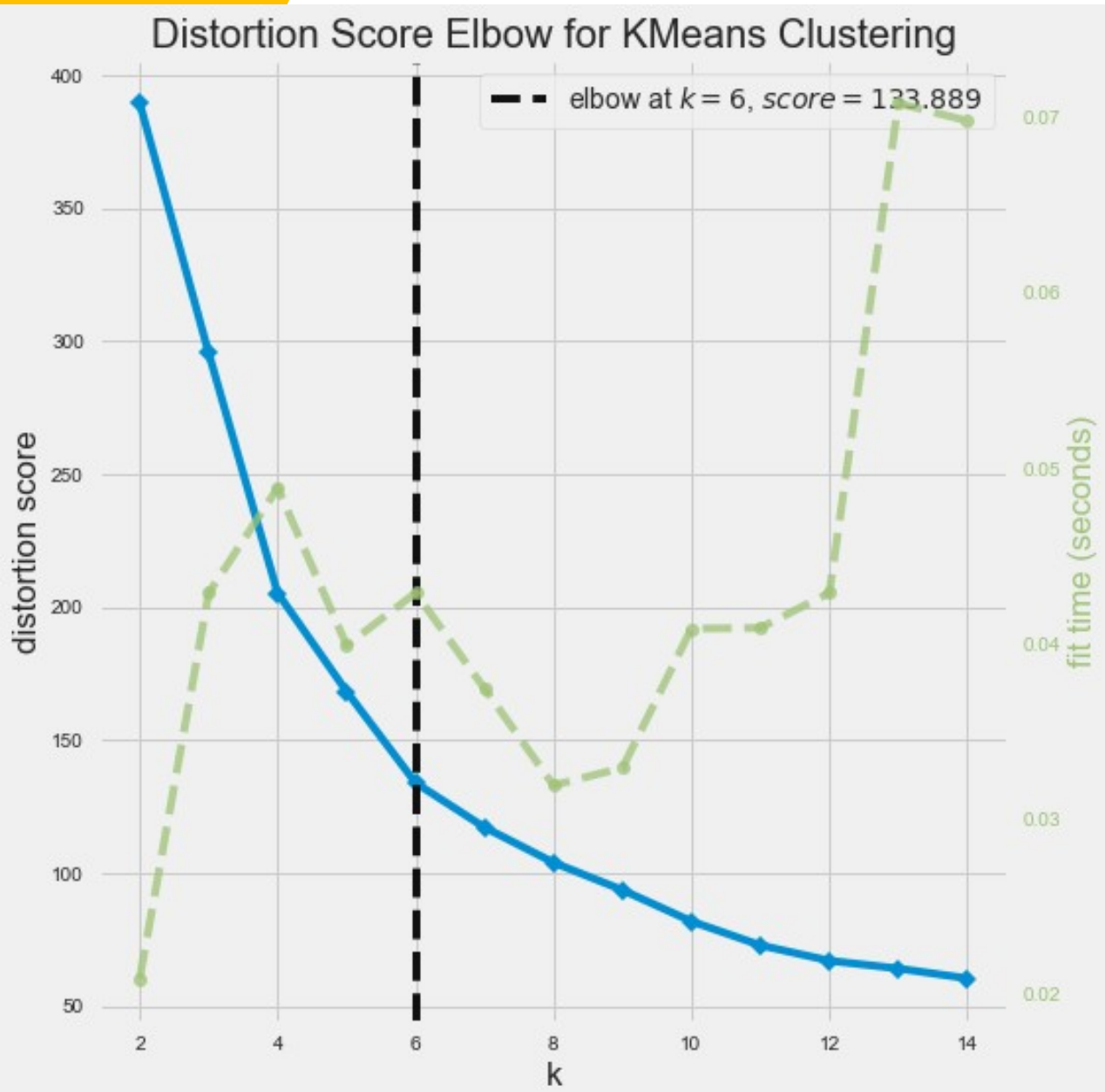


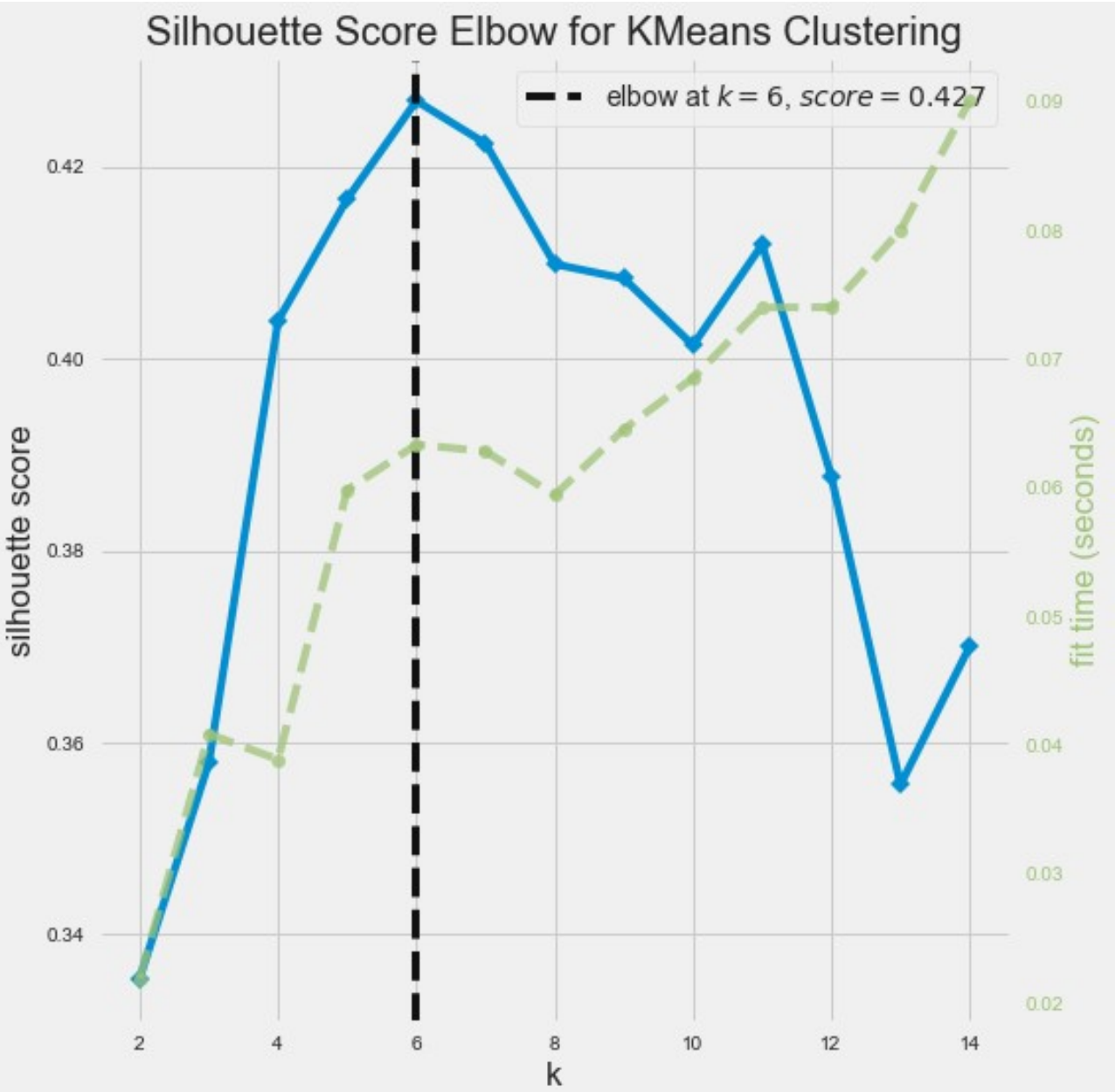
K-Means Clustering

K-means clustering is a simple and popular algorithm for clustering data. It is a simple algorithm that is easy to implement and understand. It is also a popular algorithm for unsupervised learning.

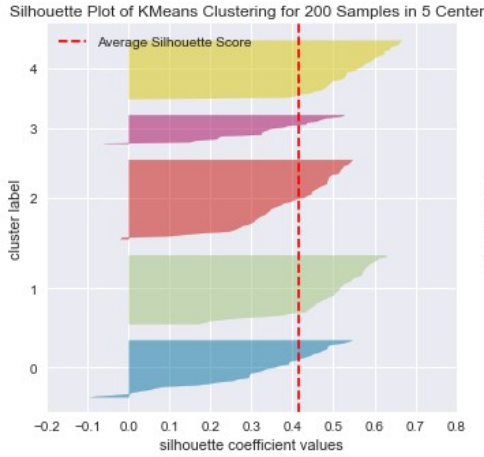
It follows an iterative approach which attempts to partition the dataset into different 'k' number of predefined and non-overlapping clusters

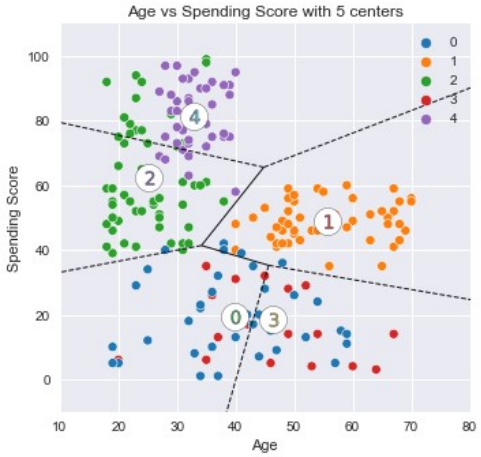


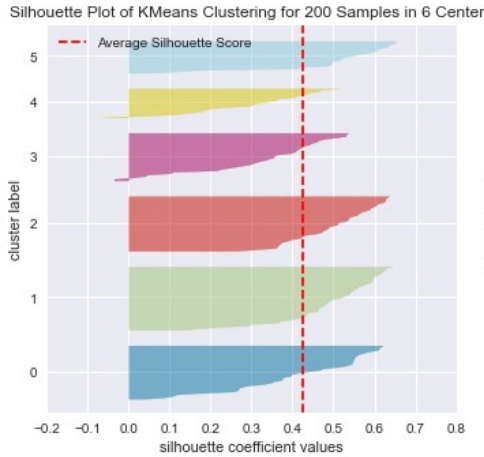

Distortion Score Elbow for KMeans Clustering
elbow at k = 6, score = 133.889
distortion score
fit time (seconds)
k
400
350
300
250
200
150
100
50
0.07
0.06
0.05
0.04
0.03
0.02
2
4
6
8
10
12
14

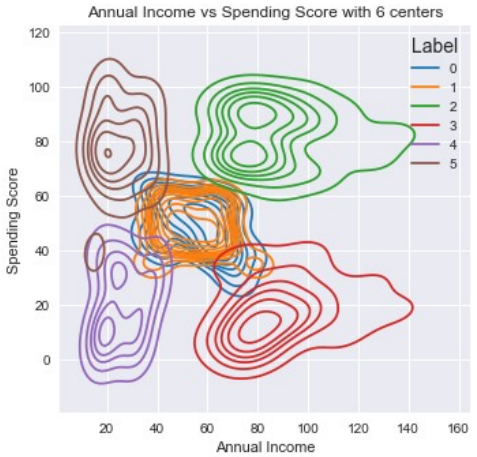

Silhouette Score Elbow for KMeans Clustering
elbow at k = 6, score = 0.427
silhouette score
fit time (seconds)
k
0.42
0.40
0.38
0.36
0.34
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
2
4
6
8
10
12
14

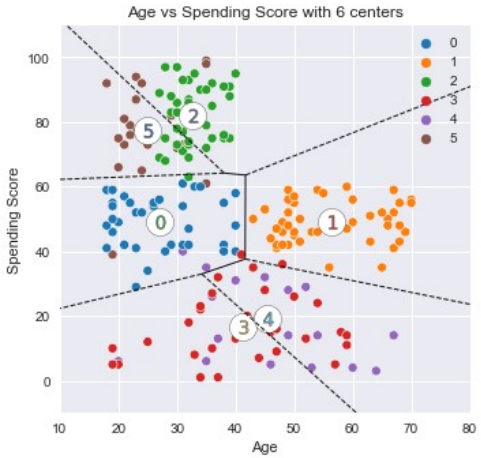
K-Means Clustering

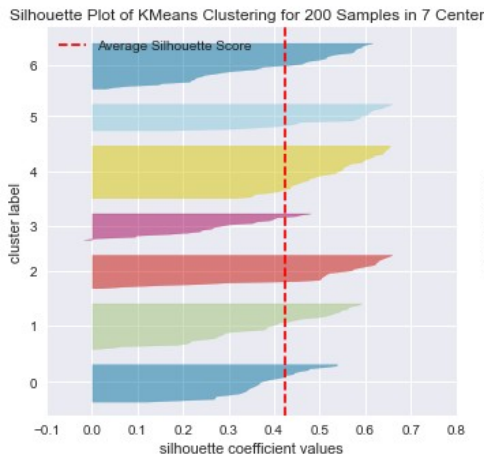


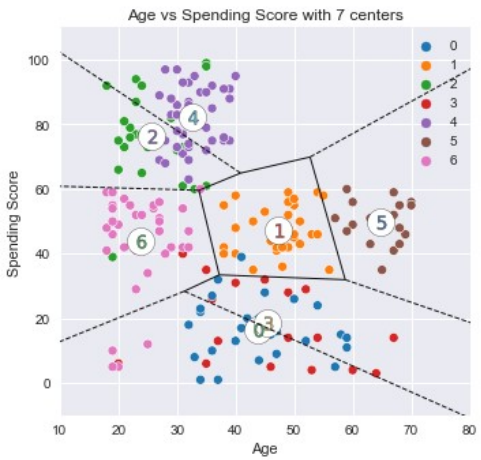


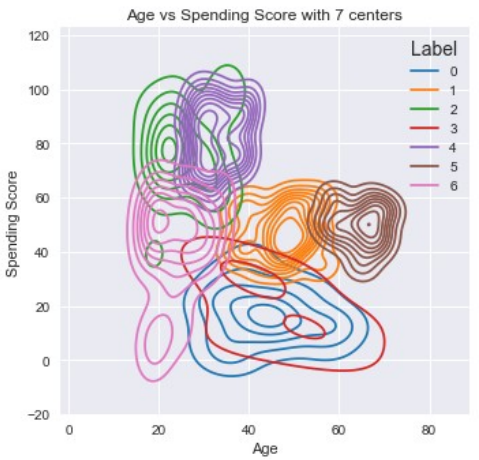








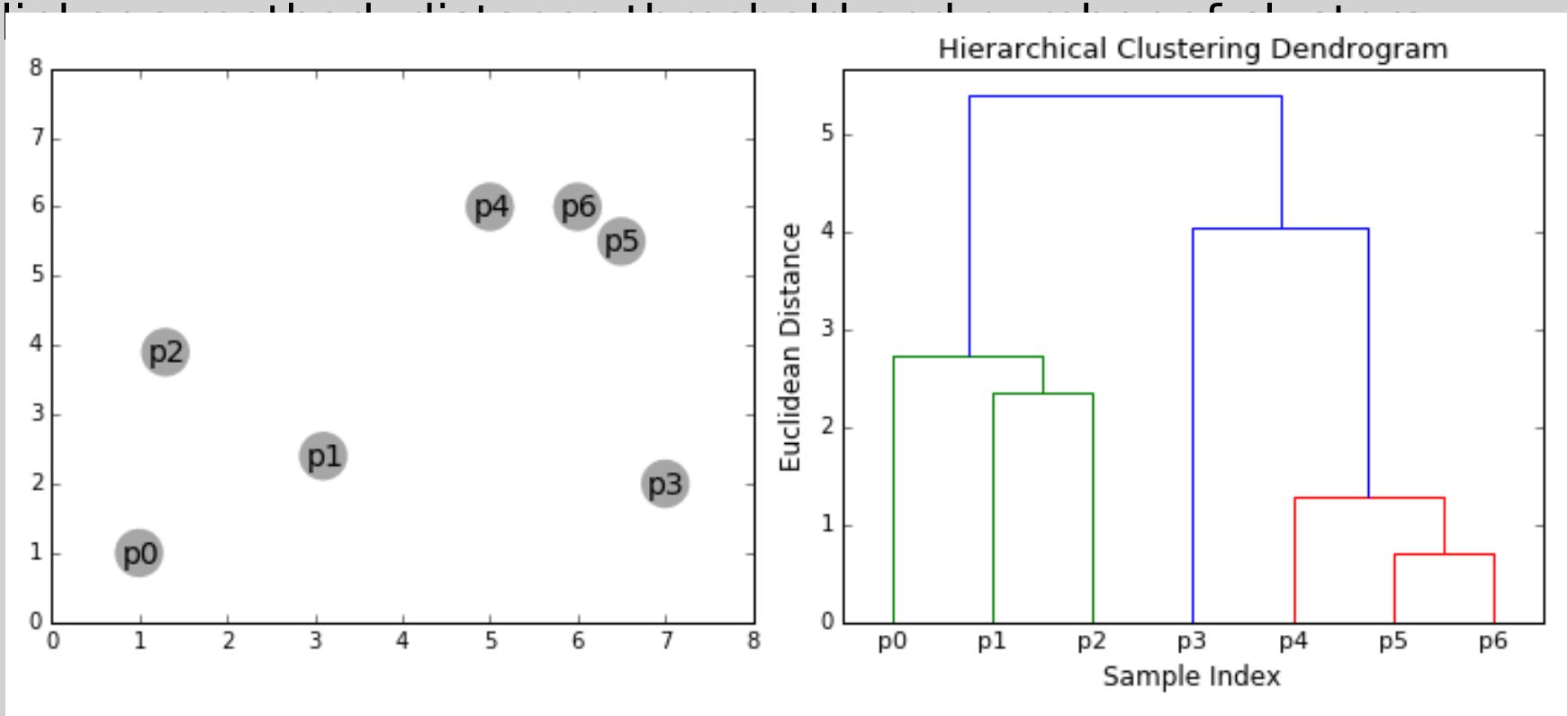


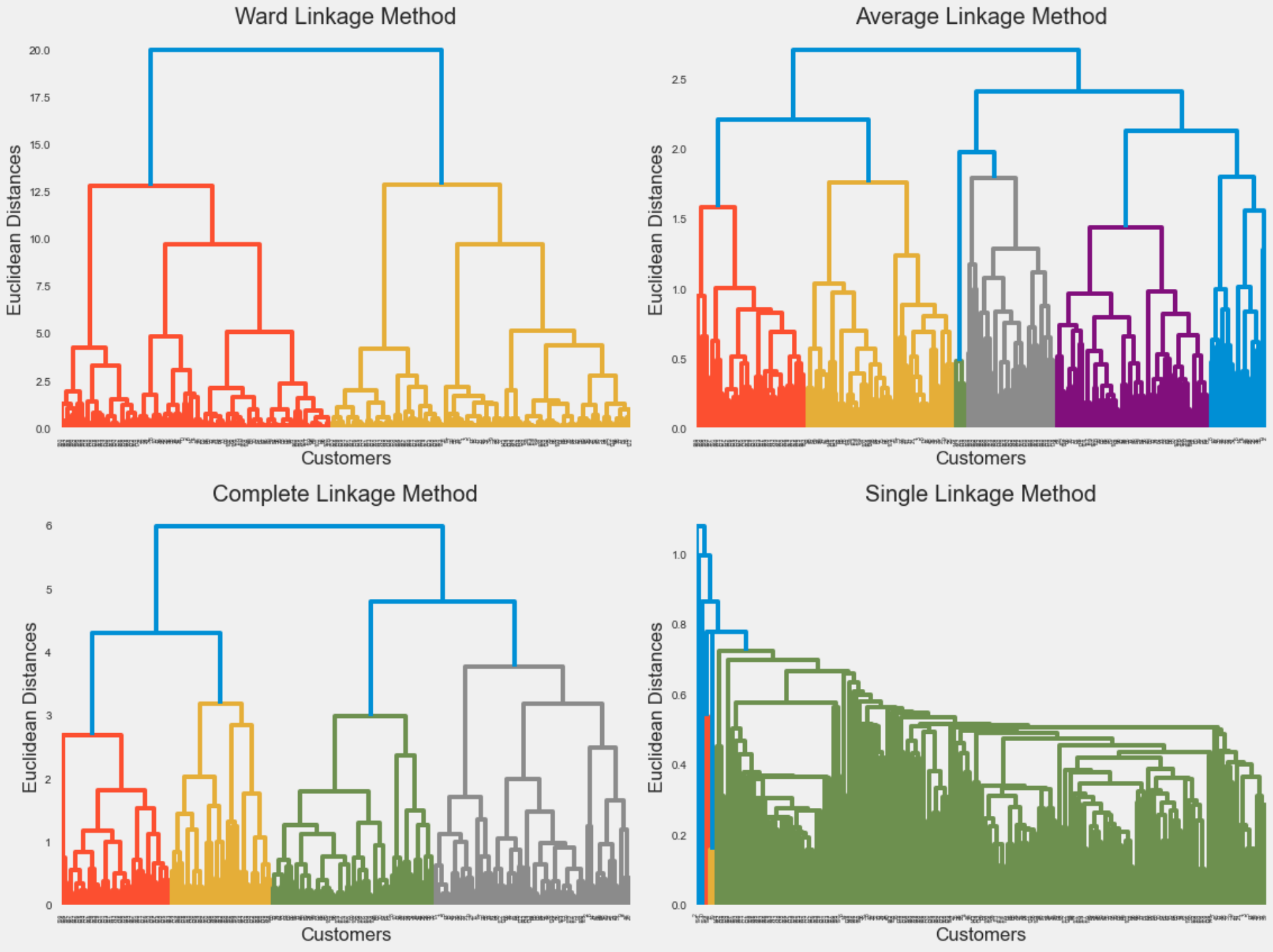


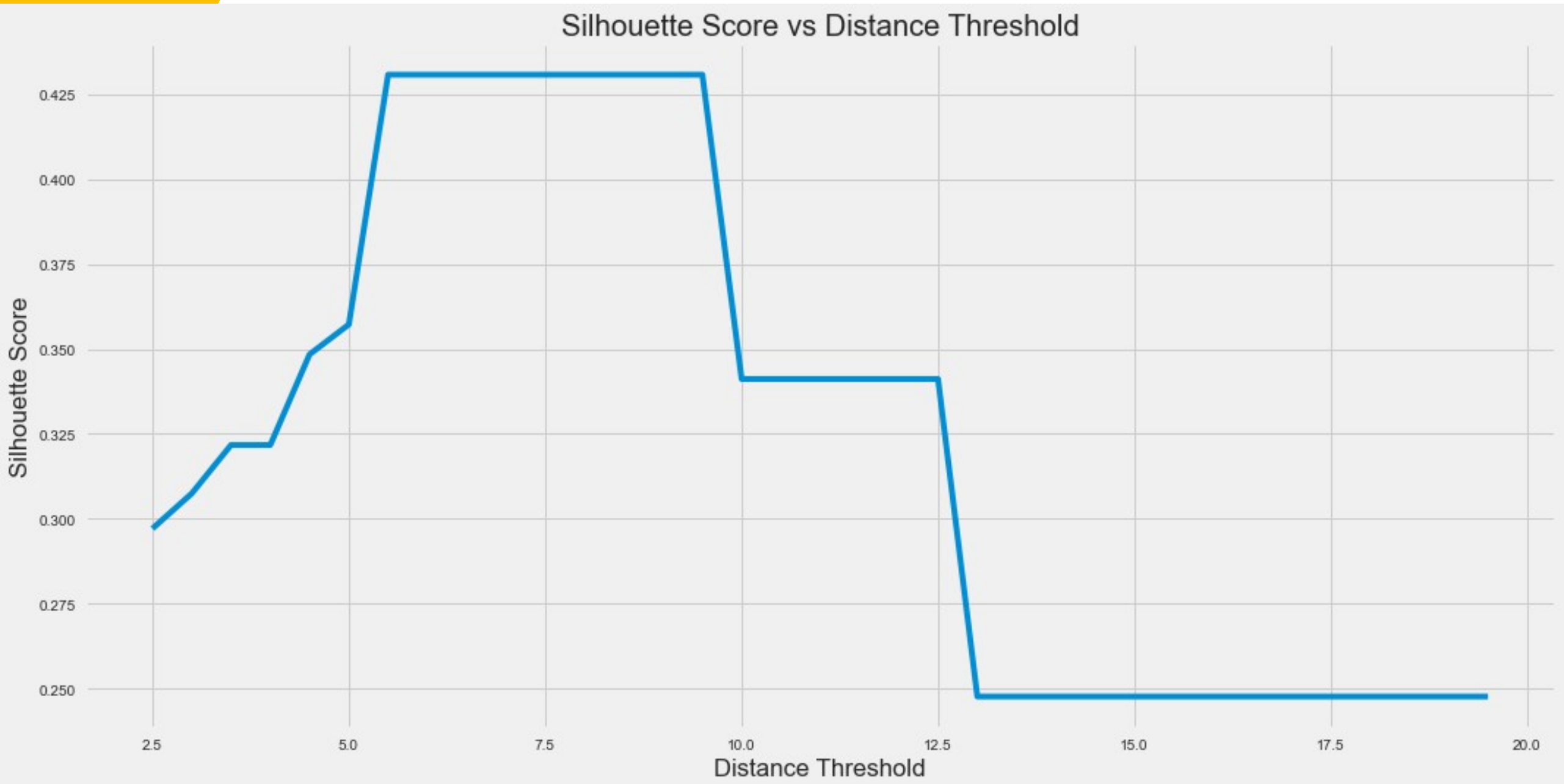
```
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.cluster import KMeans

# create a pipeline combining a scaler that takes in continuous features and a clustering algorithm
km = Pipeline(
    steps=[
        ('preprocessor', ColumnTransformer(transformers=[
            ('standardisation', scaler, [
             'Age', 'Spending Score', 'Annual Income'])
        ])),
        ('clustering', KMeans(algorithm='elkan',
                              init='k-means++', max_iter=500, n_clusters=6))
    ]
)

# fit with the original dataset
km.fit(customers)
print(km)
```

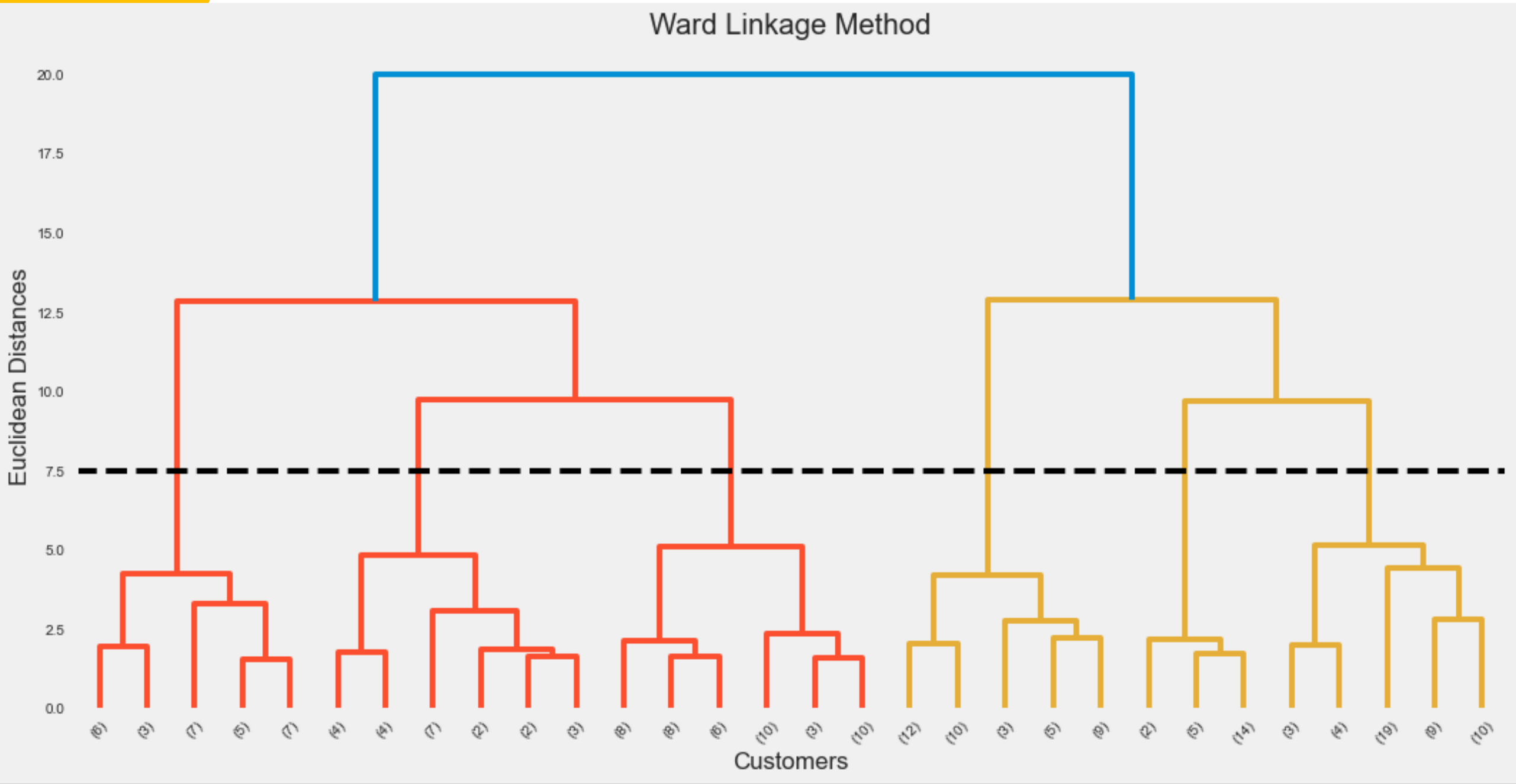
Agglomerative Hierarchical Clustering

Instead, Hierarchical Clustering iteratively merges two clusters until there is only one cluster left. Since it can organize any sort of data into a hierarchy, it can be visualized as a tree-like diagram called a dendrogram. Hierarchical Agglomerative Clustering usually has three parameters –



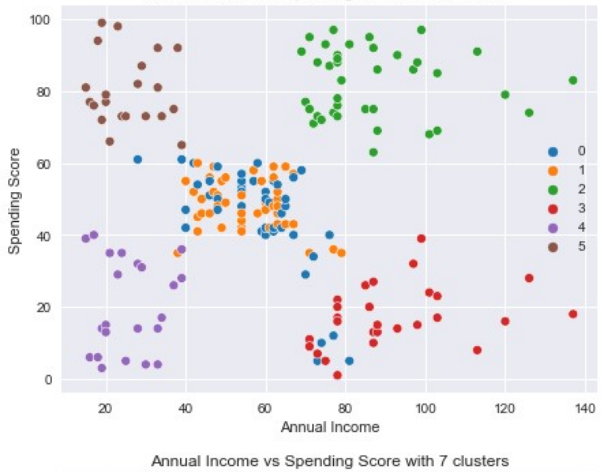

Ward Linkage Method
Euclidean Distances
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Customers
Average Linkage Method
Euclidean Distances
2.5
2.0
1.5
1.0
0.5
0.0
Customers
Complete Linkage Method
Euclidean Distances
6
5
4
3
2
1
0
Customers
Single Linkage Method
Euclidean Distances
1.0
0.8
0.6
0.4
0.2
0.0
Customers

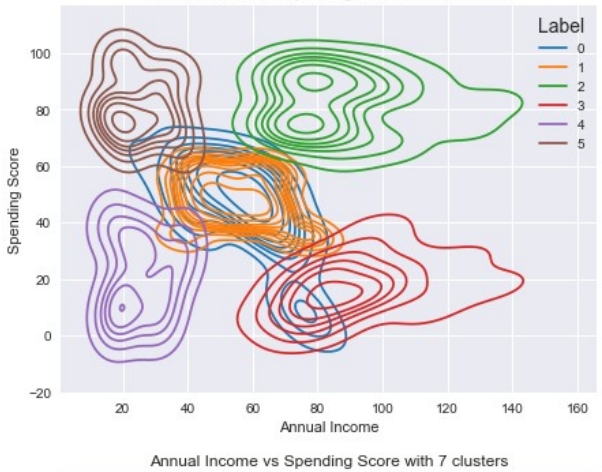

Silhouette Score vs Distance Threshold
Silhouette Score
0.425
0.400
0.375
0.350
0.325
0.300
0.275
0.250
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Distance Threshold

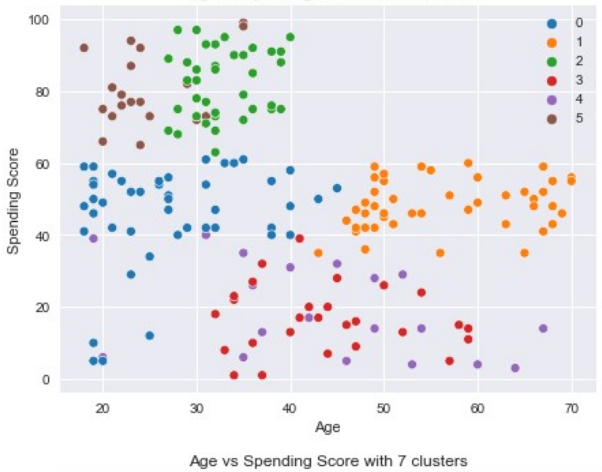

Ward Linkage Method
Euclidean Distances
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
(6) (3) (7) (5) (7) (4) (4) (7) (2) (2) (3) (8) (8) (6) (10) (3) (10) (12) (10) (3) (5) (9) (2) (5) (14) (3) (4) (19) (9) (10)
Customers

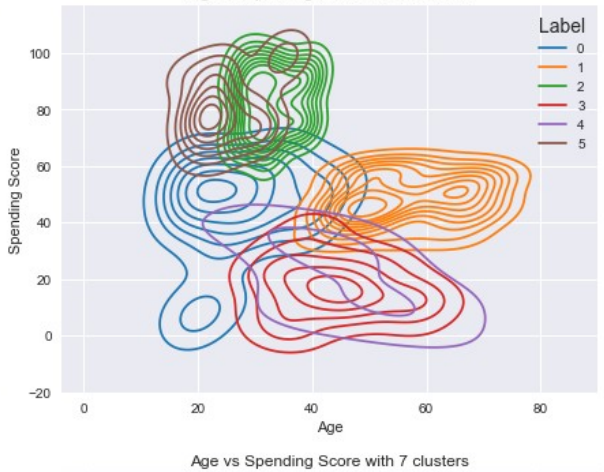

Silhouette Score Elbow for AgglomerativeClustering Clustering
elbow at k = 6, score = 0.420
silhouette score
fit time (seconds)
k
0.42
0.40
0.38
0.36
0.34
0.32
0.010
0.009
0.008
0.007
0.006
0.005
0.004
2
4
6
8
10
12
14

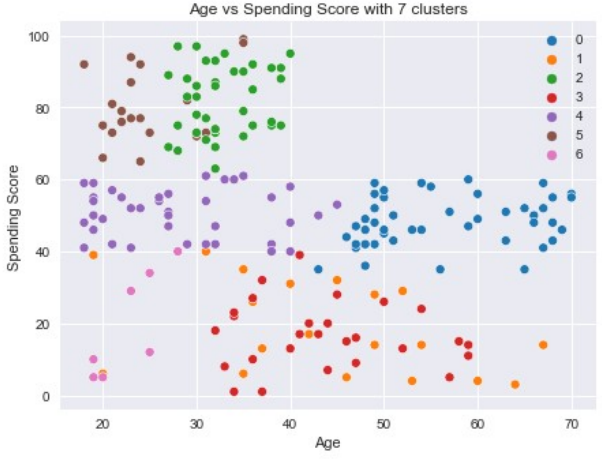
Agglomerative Clustering

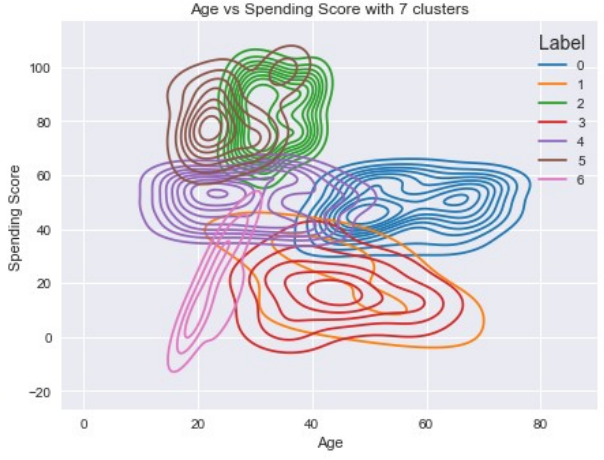








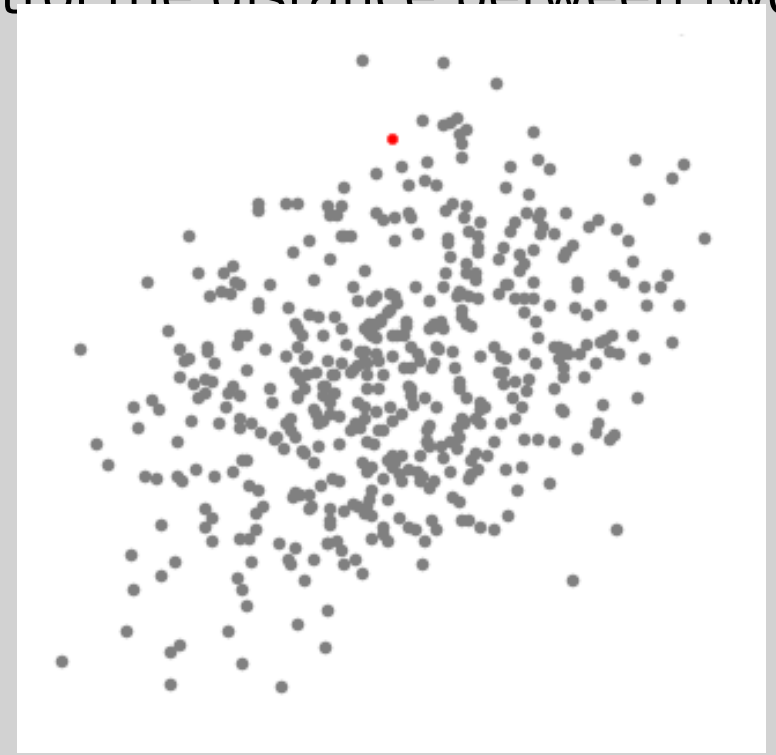


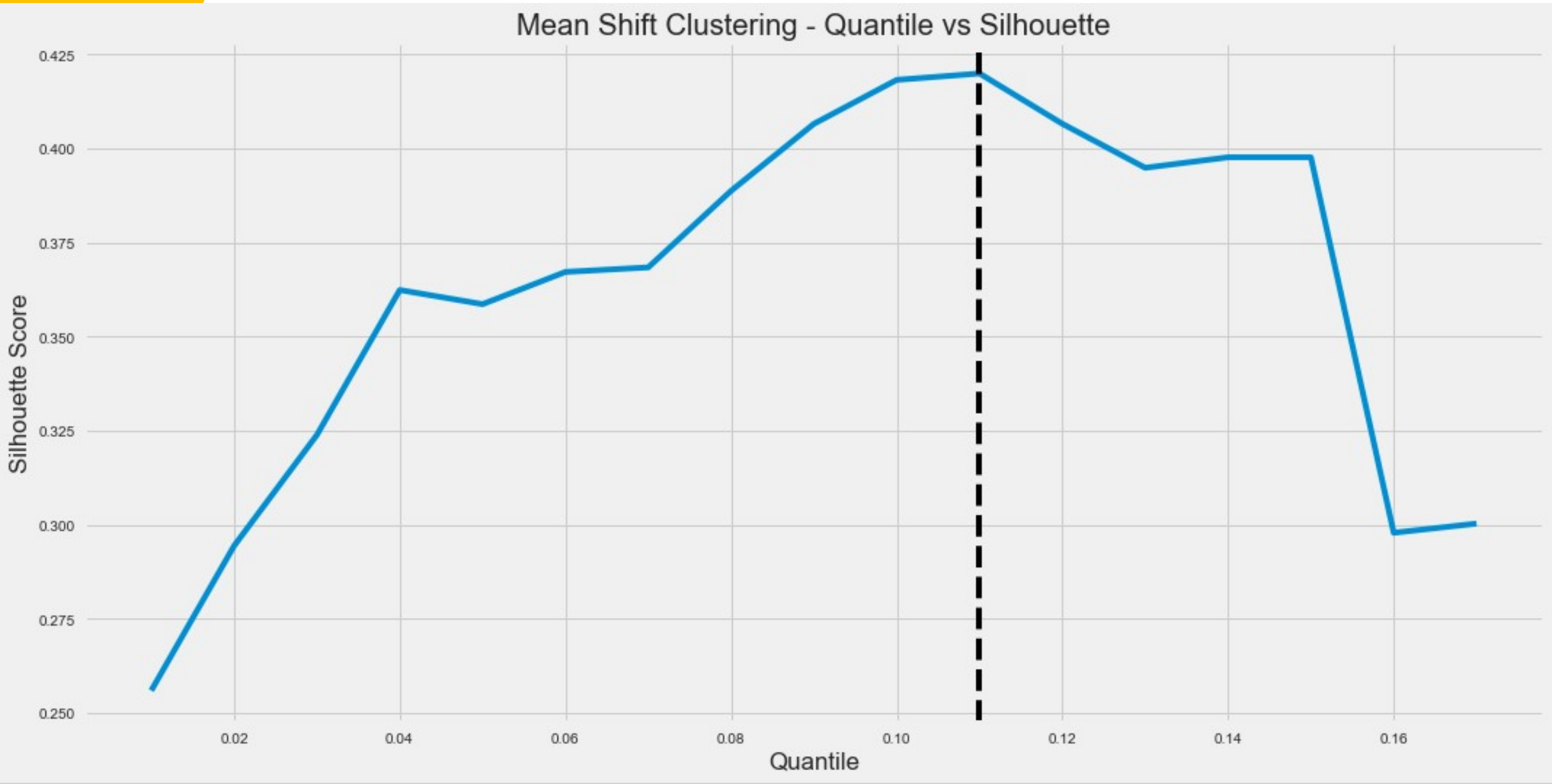


```
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.cluster import AgglomerativeClustering

# create a pipeline combining a scaler that takes in continuous features and a clustering algorithm
AC = Pipeline(
    steps=[
        ('preprocessor', ColumnTransformer(transformers=[
            ('standardisation', scaler, [
             'Age', 'Spending Score', 'Annual Income'])
        ])),
        ('clustering', AgglomerativeClustering(
            n_clusters=6, affinity='euclidean', linkage='ward'))
    ]
)

# fit with the original dataset
AC.fit(customers)
print(AC)
```

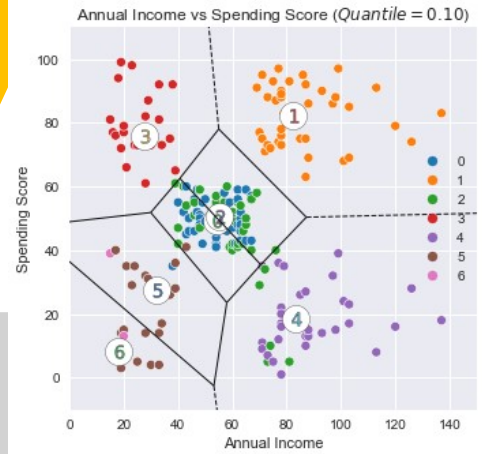
Mean Shift Clustering

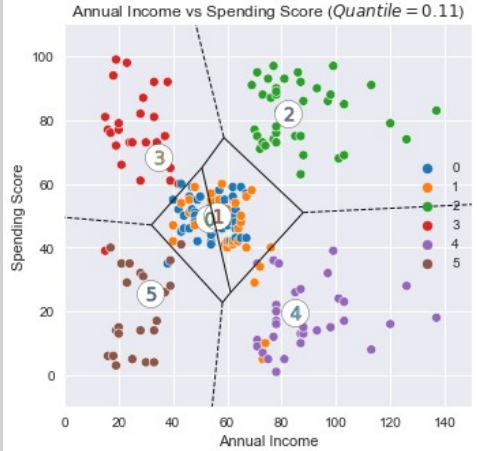
Mean Shift clustering algorithms aims to discover high density regions, using Kernel Density Estimation, by updating the locations of candidate centroids of a given region. This process is repeated until a final set of centroids is found. Mean Shift clustering has only one parameter – bandwidth, to control the distance between two centroids.

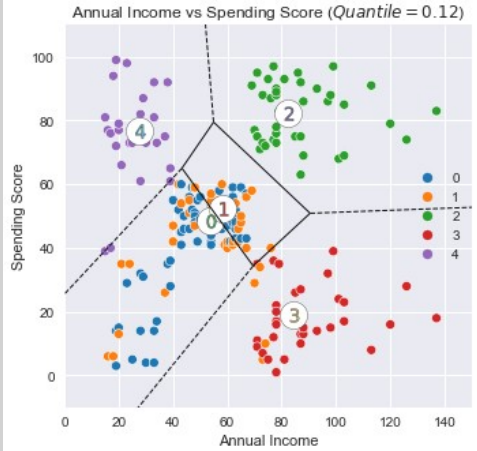



Mean Shift Clustering - Quantile vs Silhouette
Silhouette Score
0.425
0.400
0.375
0.350
0.325
0.300
0.275
0.250
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
Quantile

Mean Shift Clustering





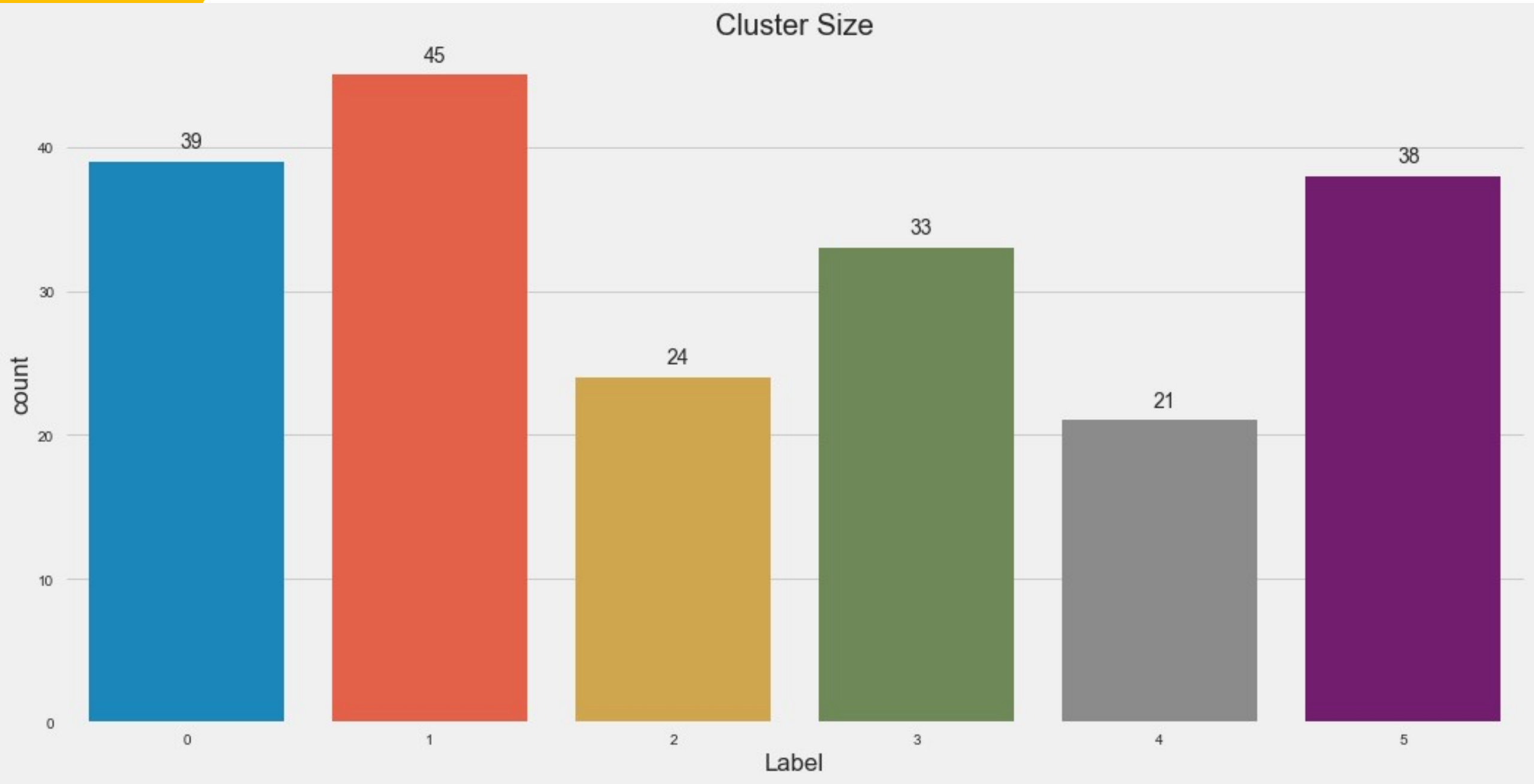


```
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.cluster import MeanShift

# create a pipeline combining a scaler that takes in continuous features and a clustering algorithm
MS = Pipeline(
    steps=[
        ('preprocessor', ColumnTransformer(transformers=[
            ('standardisation', scaler, [
             'Age', 'Spending Score', 'Annual Income'])
        ])),
        ('clustering', MeanShift(bandwidth=1.0752644580687445, bin_seeding=True))
    ]
)

# fit with the original dataset
MS.fit(customers)
print(MS)
```

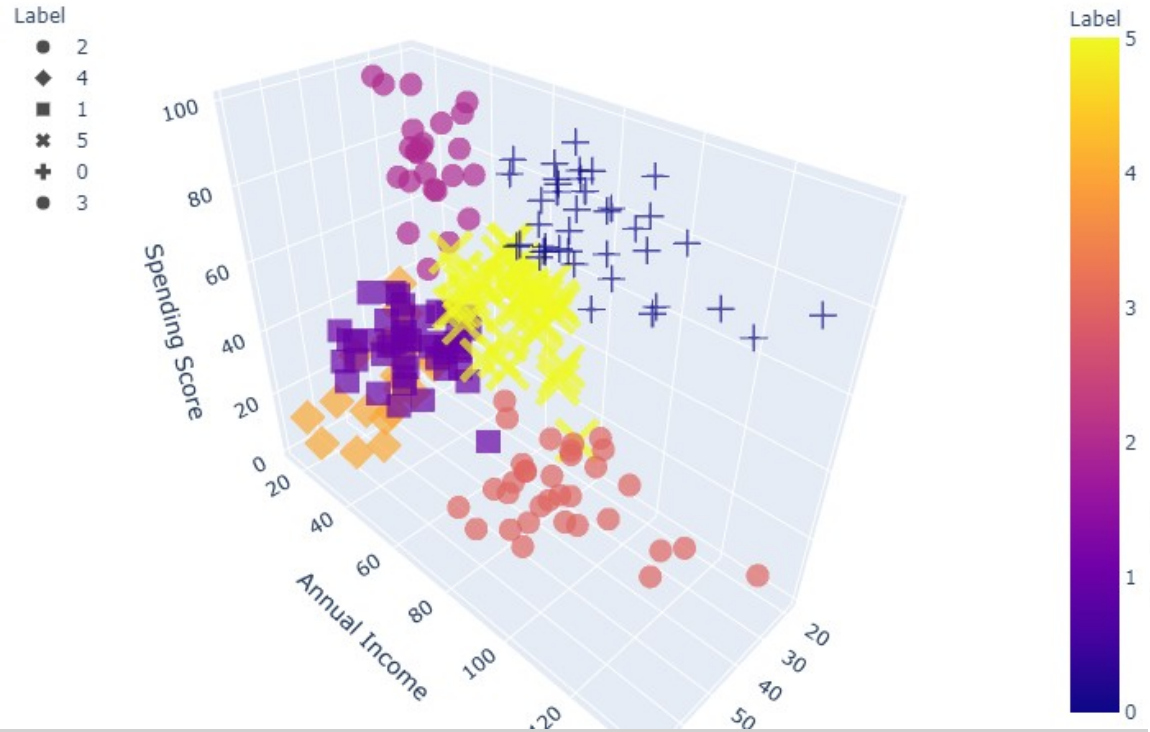
Results

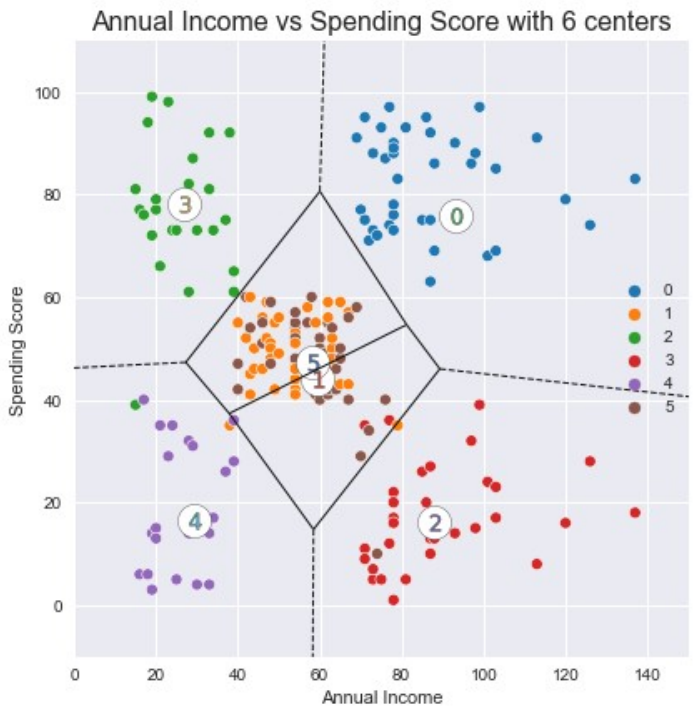
K-Means Clustering, k = 6, is the best performing algorithm among all of the clustering algorithms. As such, K-Means Clustering is the most suitable algorithm to perform customer segmentation.

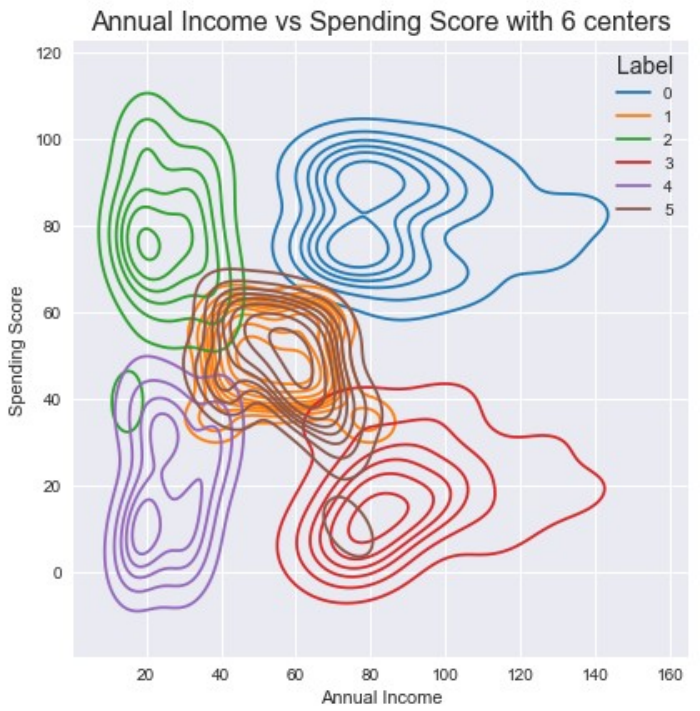
	KMeans	Agglomerative	MeanShift
Silhouette Score	0.446311	0.430749	0.437942
Calinski Harabasz Score	160.475858	147.029982	154.579019
Davies-Bouldin Score	0.764581	0.834040	0.774959


Cluster Size
45
39
38
33
24
21
40
30
20
10
0
count
0
1
2
3
4
5
Label

Results


Label
2
4
1
5
0
3
Spending Score
Annual Income
Label
5
4
3
2
1
0


Annual Income vs Spending Score with 6 centers
Spending Score
Annual Income
0
1
2
3
4
5


Annual Income vs Spending Score with 6 centers
Label
0
1
2
3
4
5
Spending Score
Annual Income


Age vs Spending Score with 6 centers
Spending Score
Age
0
1
2
3
4
5


Age vs Spending Score with 6 centers
Label
0
1
2
3
4
5
Spending Score
Age

Cluster	Proportion of Sample	Description	Interpretation
0	19.5%	High Annual Income High Spending Score Young Age	High Income and High Spending Scores. High chance this group are Executives, earning big money. Being the **second largest** cluster, the mall owners should prioritize targeting this group to increase their earnings.
1	22.5%	Middle Annual Income High Spending Score Young Age	This group could be possibly retirees or older workers. Having the largest indicates that the most frequent customers are retirees or old workers. The Mall Owner should pivot their strategies to target older people since it is their **largest** customer base.
2	12%	High Annual Income Low Spending Score Middle Age	High annual income but low spending score. These people are like executives, but they are much thriftier, which explains their small proportion of the sample.
3	16.5%	Low Annual Income High Spending Score Young Age	It is high likely that these group of customers are students, having a low income while having high spending. The mall owner could target this group for profit, since they are more vulnerable. Possible business ideas is adding entertainment outlets.
4	10.5%	Low Annual Income	Low income and Spending Score, this group is significantly poorer. It would unethical and unprofitable to target this group.

Wisdom

Some suggestions after analysing the clusters.

- Employ more F&B (restaurants and eateries) and Necessities (supermarkets and department stores) retail outlets which caters to old people or families, since there is a large customer base of them.
- Since there are a larger segment has either high or moderate spending scores, the mall owners can rope in MNC Franchise to ensure customer loyalty to the mall; due to brand loyalty.
- The mall owners should target the Executives for advertisements with luxury products since they are the most likely to generate large profit.
- Since there are more families and older people, the mall owners could provide incentives for them, such as weekend discounts, free parking etc., to ensure that they are more likely to come to the mall.