INDR422 TERM PROJECT ASSIGNMENT

BERKE KARAMANLI (0079669)
ÖMER TATLICI (0079201)
INSTRUCTOR: FİKRİ KARAESMEN

1. INTRODUCTION

Throughout the course of INDR422, we learnt how to forecast different datasets by using various methods with different complexities on our homework materials, lecture notes and labs. After equipped with necessary knowledge in the semester, we are now able to conduct forecasting models by working on a real-world and up-to-date dataset.

2. DATASET AND SUPPORTING PREDICTOR DESCRIPTIONS

In this project, we are aiming to find the most accurate forecasting technique to predict Istanbul's daily water consumption provided by the Municipality of Istanbul (IBB). The dataset contains water consumption in terms of m^3/day for the years 2011-2023. Along with consumption data, the dataset contains daily water increase in Istanbul's major 10 dams. We will also use this information as a predictor to indicate whether it rained in Istanbul or not since dam increase indicates rain or snow, or in other terms, precipitation. Additionally, we considered daily, weekly and yearly lagged observations as important predictors and used them throughout the prediction process.

We also supported the main dataset with daily average temperature in Istanbul, column named as *temp*. The weather data was sourced from Visual Crossing Corporation (2024). We believe the temperature affects water consumption as we may follow seasonal changes by temperature averages. Another source used throughout prediction process is the yearly temperature dataset provided by Turkish Statistical Institute (TUIK). We also used yearly trend predictors.

We finally constructed dummy variables that indicate what day of the week and what month of the year the datapoint is in.

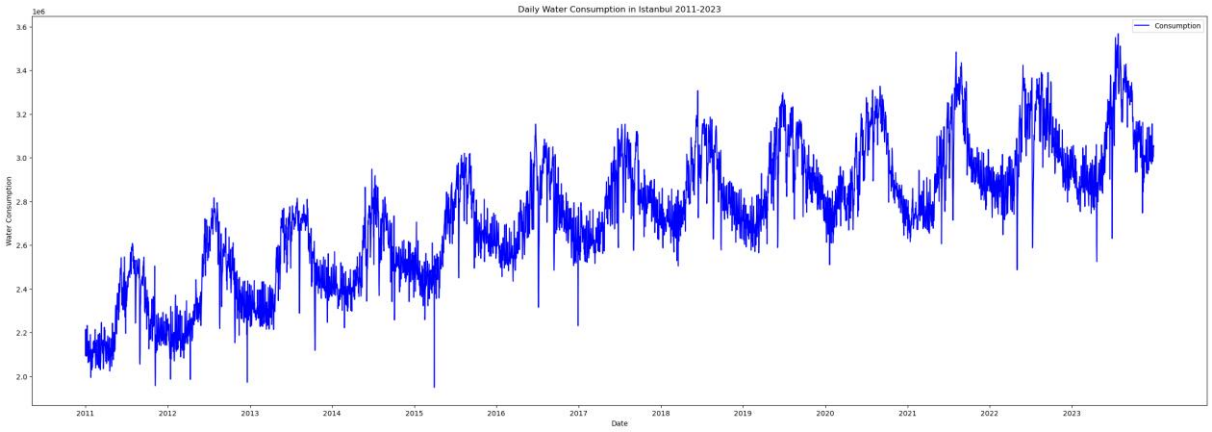

Figure 1: Water Consumption Data Visualization

From Fig. 1, we can clearly observe yearly seasonality with the seasonal peak at summer and bottom at winter. Also, a positive trend is clearly visible.

3. FORECAST METHODS

Naive-1 Forecast

We initiated our forecasts for the dataset by using observations 1 day behind. The forecast and consumptions almost match as seen in Fig. 2. It is almost impossible to distinguish consumption and forecast lines.

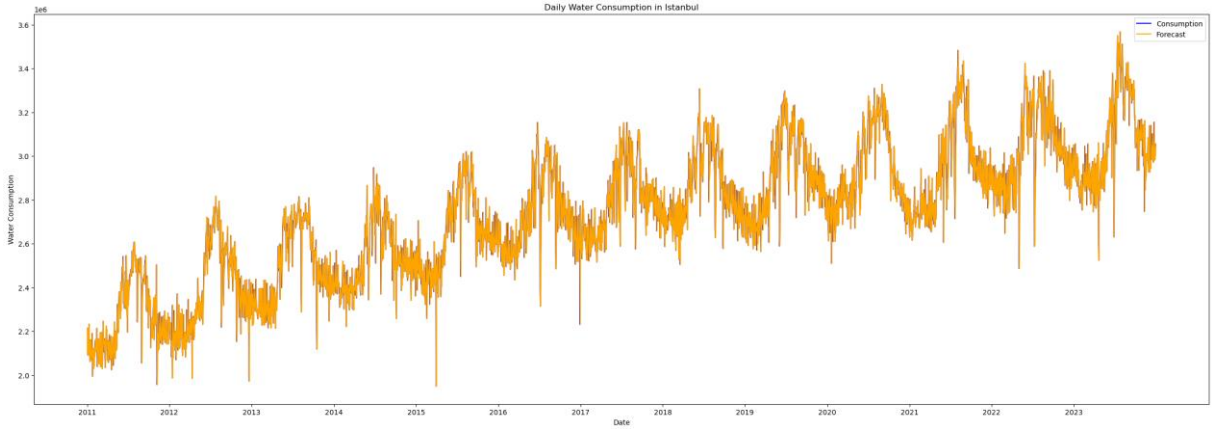

Figure 2: Naive-1 Forecast vs Observed Consumption Data

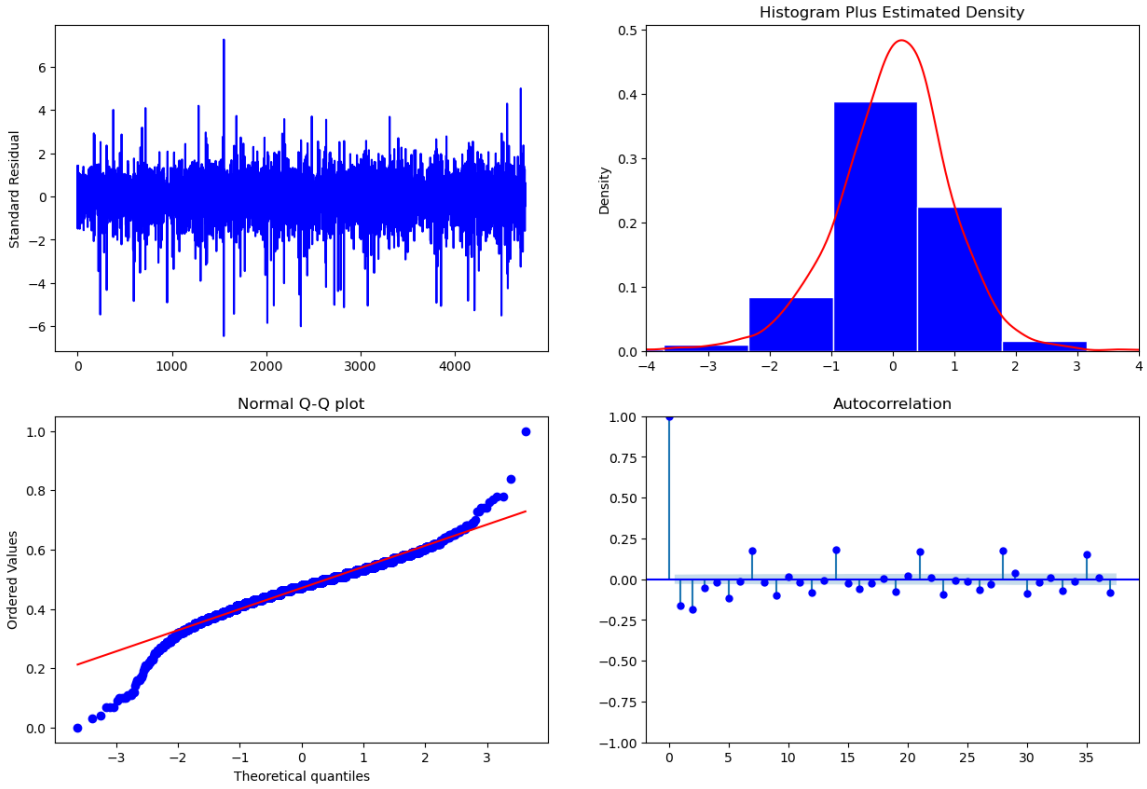

Figure 3: Residual Plots of Naive-1 Forecast

As seen from Fig. 3, the residual of the forecast is close to zero aside from the recurring lags weekly. We can assume the residual to be random and rely on the model.

```
MSE: 6870578764.29
RMSE: 82888.95
MAE: 60558.74
MAPE: 2.24%
```

As seen from the error metrics on the left, Naive-1 forecast provides accurate results. This means that the overall change from one day before is not large. It is safe to say that the first benchmark forecast set a high standard in terms of error metrics.

Naive-7 Forecast

Second naive forecast example for setting a benchmark is the data from one week before as it might be a beneficial tool to catch weekly seasonality. Forecast plot is made below in Fig. 4 and we start to see a shift towards the right. While it is still able to catch weekly seasonality, it cannot adapt to lower or higher weeks.

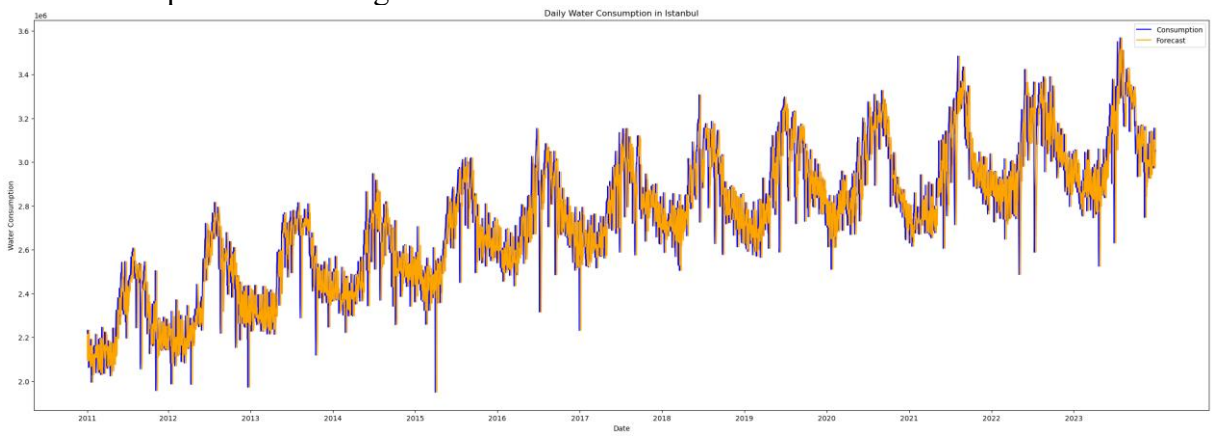

Figure 4: Naive-7 Forecast vs Observed Consumption Data

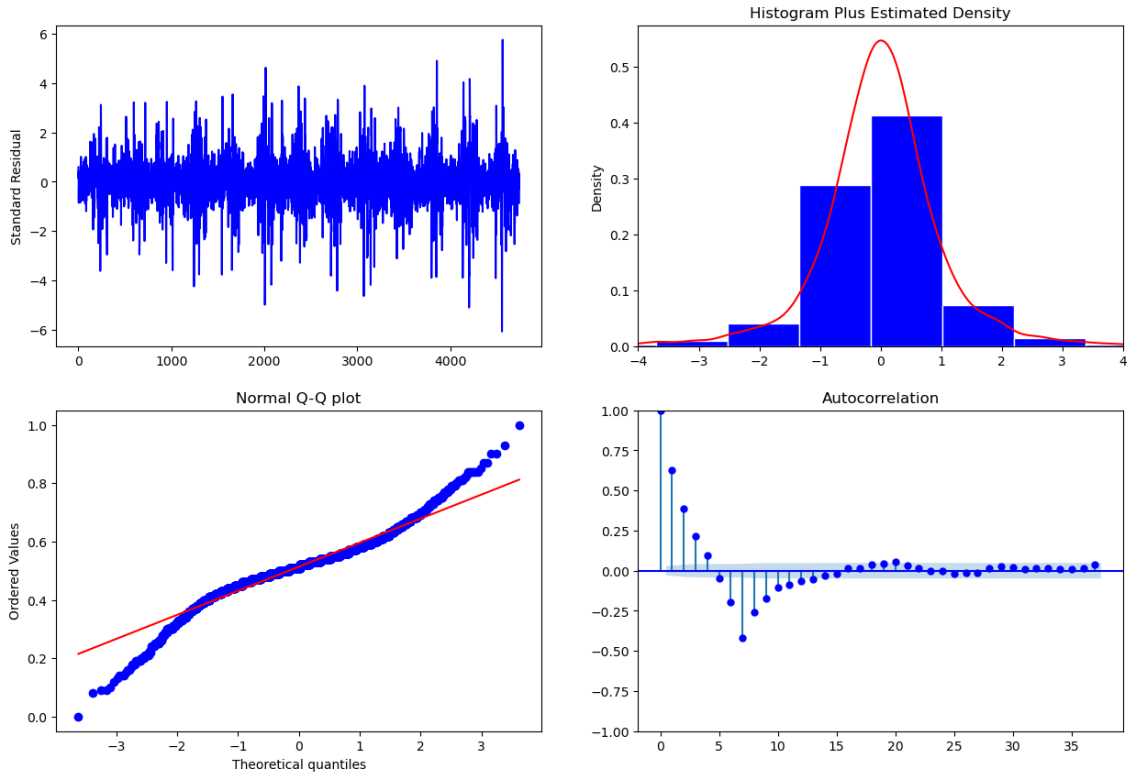

Figure 5: Residual Plots of Naive-7 Forecast

As seen above in Fig. 5, the residual distribution is still normal, but we can see the weekly lag and the trend on the autocorrelation plot more clearly. Aside from the autocorrelation plot, we are still able to rely on our model.

```
MSE: 15208237188.85
RMSE: 123321.68
MAE: 86367.51
MAPE: 3.15%
```

To comment on the performance of Naive-7 forecast based on error metrics, we sacrificed a bit of accuracy as MAPE increased by 1%. In conclusion, the metrics are still more than acceptable as MAPE is 3.15%.

Naive-365 Forecast

We finalize naive benchmark forecasts by using datapoint from 1 year (365 days) before. We can observe the lack of catching positive trend as our forecasts line is exactly under the observation line in Fig. 6.

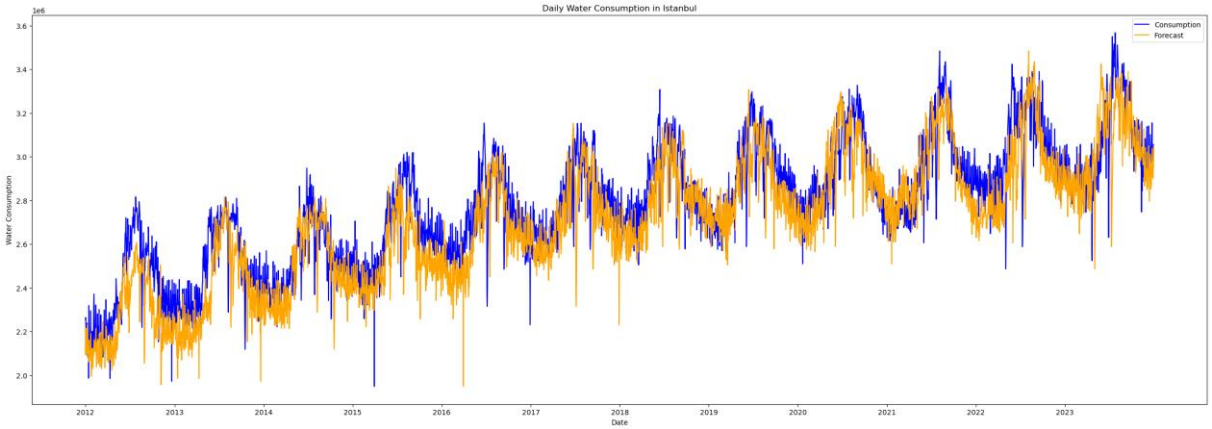

Figure 6: Naive-365 Forecast vs Observed Consumption Data

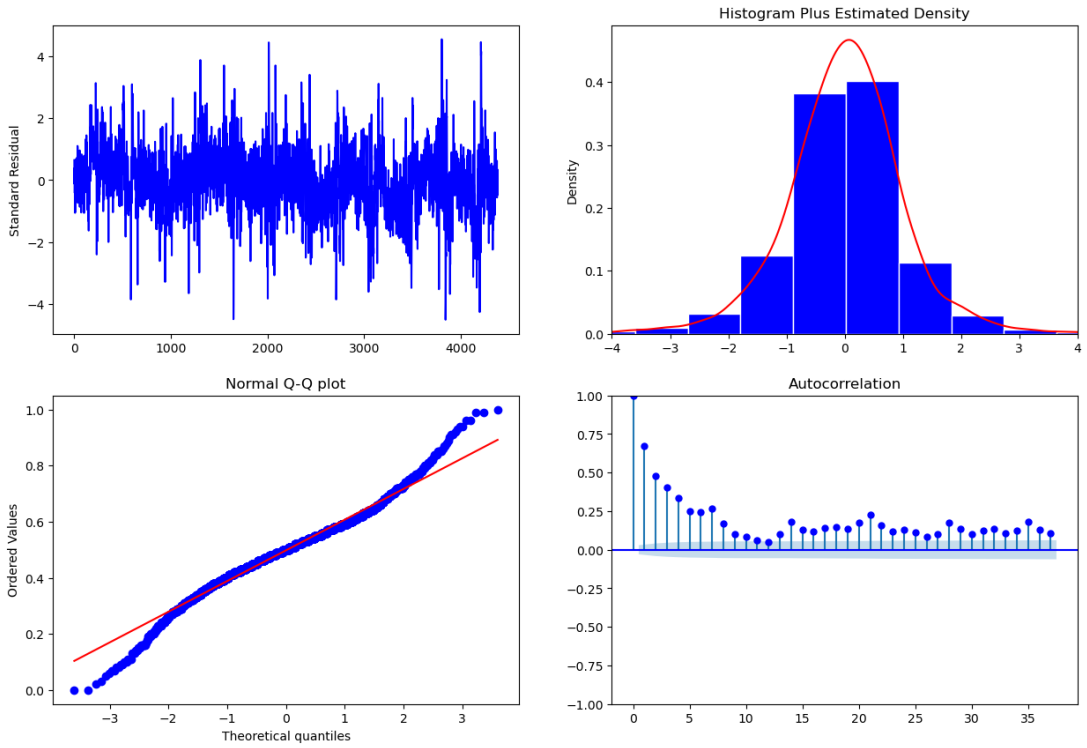

Figure 7: Residual Plots of Naive-365 Forecast

Fig. 7 shows that even though the residuals are still close to being normally distributed, there is a visible positive trend on the autocorrelation plot with weekly seasonality inside. We are still able to rely on our forecasting model.

```
MSE: 25122513887.19
RMSE: 158500.83
MAE: 122577.62
MAPE: 4.41%
```

From the error metrics on the left, we deviated a bit more from the initial benchmark models as MAPE increased again by nearly 1%. Even with the deviation, the model's error metrics conclude acceptable forecasting errors.

Moving Average over 7 Periods

The next model is conducted by using moving averages over 7 periods which takes the mean of most recent 7 observations. Fig. 8 shows that our forecasts are more stable as taking the mean decreased the variance, but this forecast is unable to reach the peaks and bottoms.

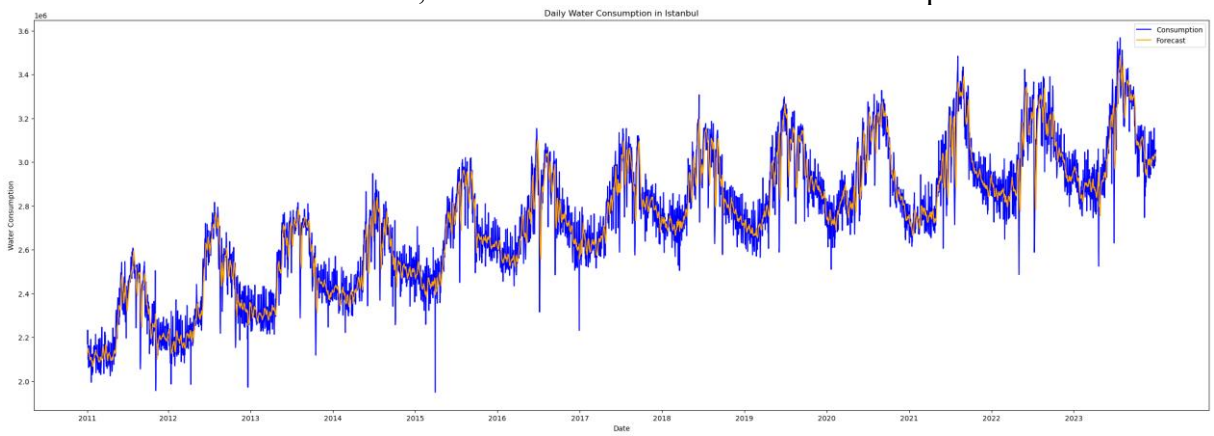

Figure 8: Moving Average-7 vs Observed Consumption Data

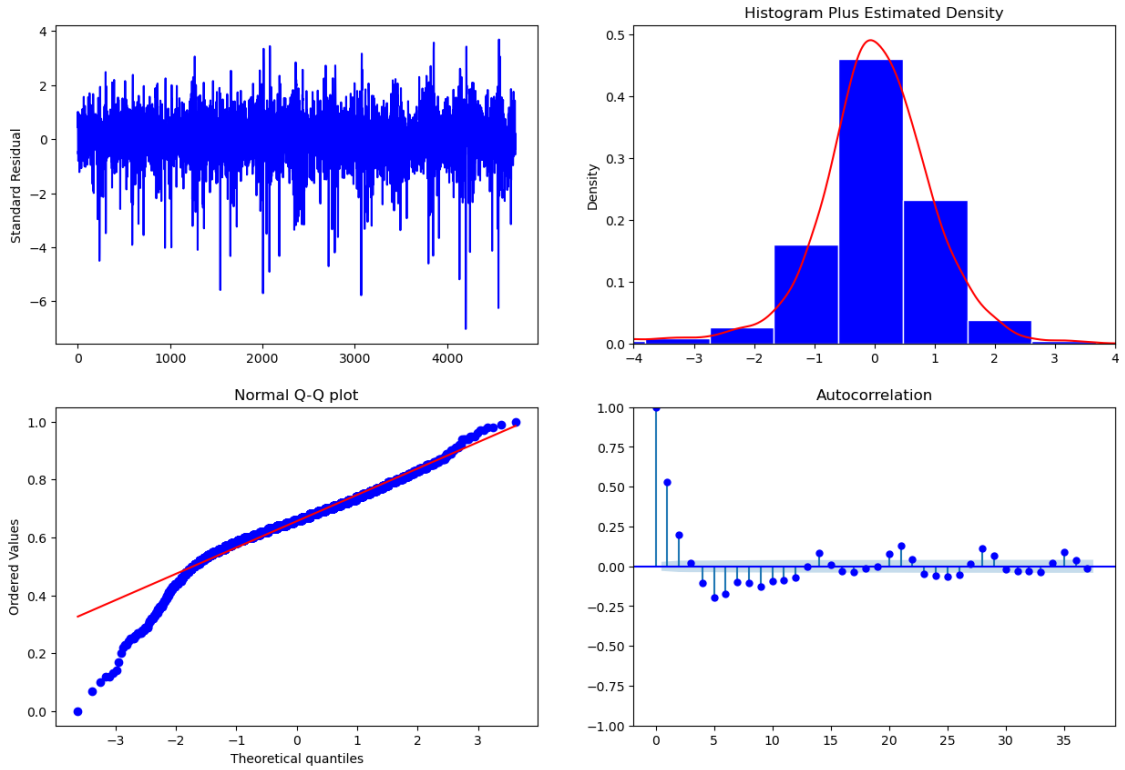

Figure 9: Residual Plots of Moving Average-7 Forecast

The plots in Fig. 9 shows that the residual is again close to being normally distributed but we are still able to observe weekly seasonality in autocorrelation plot. We can still rely on our forecasts.

```
MSE: 8441031382.07
RMSE: 91875.09
MAE: 66364.10
MAPE: 2.44%
```

The error metrics show that the forecast is accurate as we have 2.44% MAPE which is great. Taking the average of previous week shows positive forecasting results.

Moving Average over 30 Periods

We know take the average of the previous 30 observations (approximately 1 month) to forecast one period ahead. Fig. 10 shows us the forecasts are even more stable than the previous moving average forecast as increasing moving average size leads to less variance.

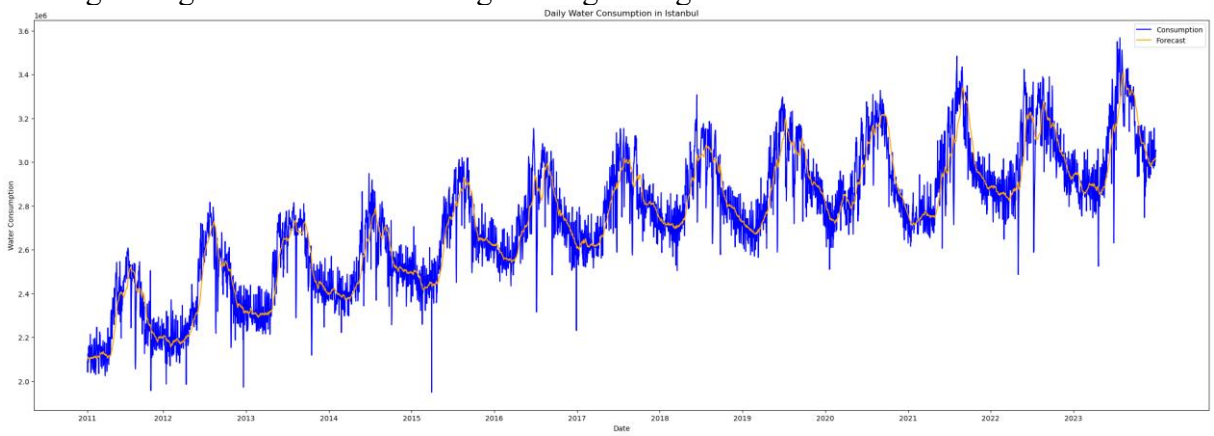

Figure 10: Moving Average-30 vs Observed Consumption Data

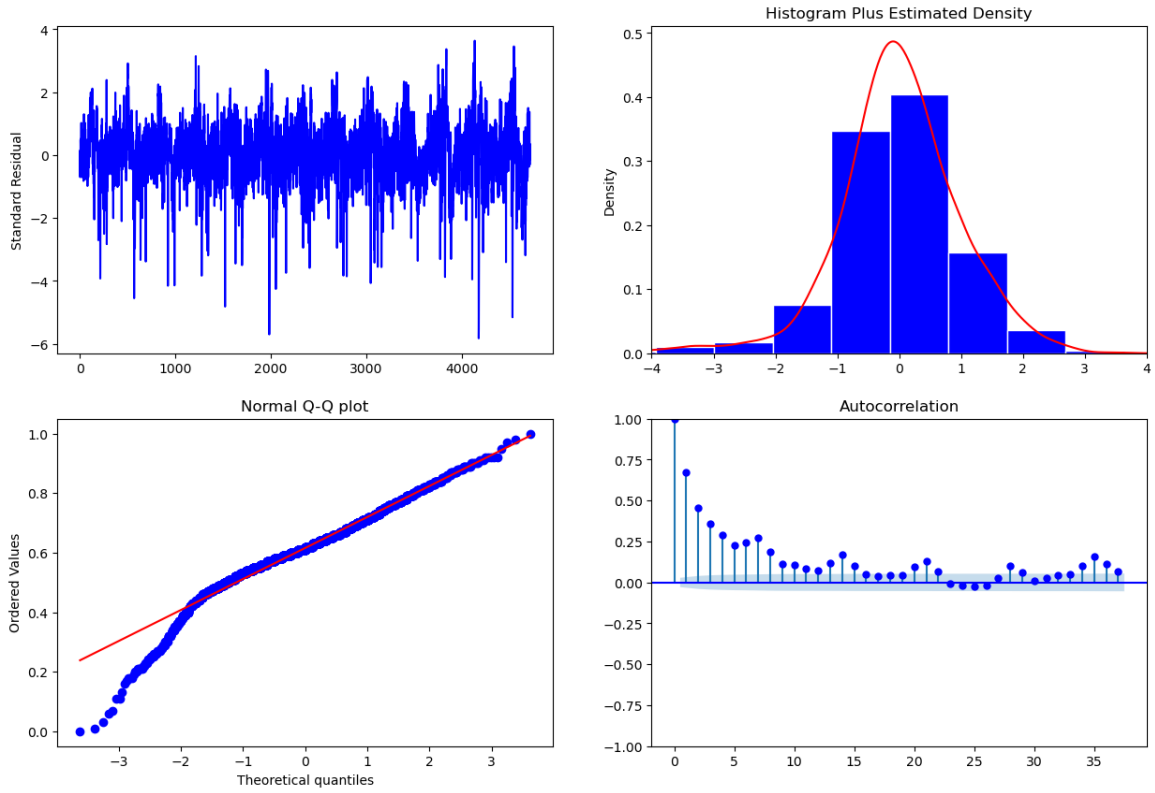

Figure 11: Residual Plot of Moving Average-30 Forecast

Fig. 11 shows resemblance with Fig. 9 since we now have more moving average terms. The only difference is the autocorrelation plot which is more unstable as monthly moving average may have not stabilized the lag severity. Other than that, we can rely on our forecasts.

```
MSE: 10969332117.93
RMSE: 104734.58
MAE: 77300.48
MAPE: 2.82%
```

Even though MAPE did not increase significantly, RMSE increased by 10%. This is an issue but there is nothing alarming in terms of percentage error.

SARIMA Forecast

The next forecasting model consists of a SARIMA model. We divided the dataset into train and test sets as train set ends at the end of 2020 and test set starts after the beginning of 2021. Before implementing the model, we need to take seasonal and trend differences in order locate the auto regressive (AR) and moving average (MA) terms. After taking trend difference and weekly differences, our data looks stable as seen in Fig. 12.

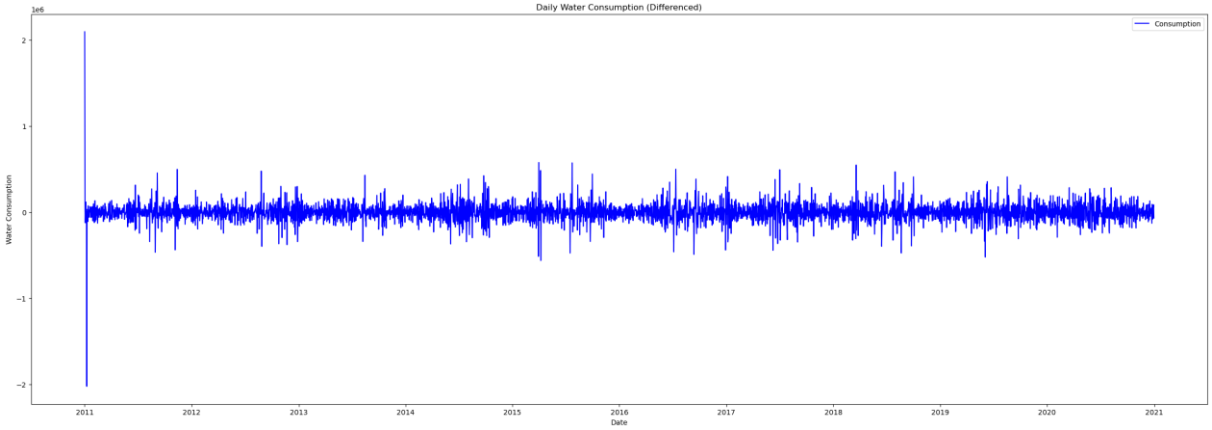

Figure 12: Water Consumption Data, Train Set Differenced Twice

This stability now allows us to observe autocorrelation function (ACF) and partial autocorrelation function (PACF) plots, shown in Fig. 13. We can observe a spike in ACF that is followed by a tail in PACF. This pattern indicates that we need an MA term to accurately forecast.

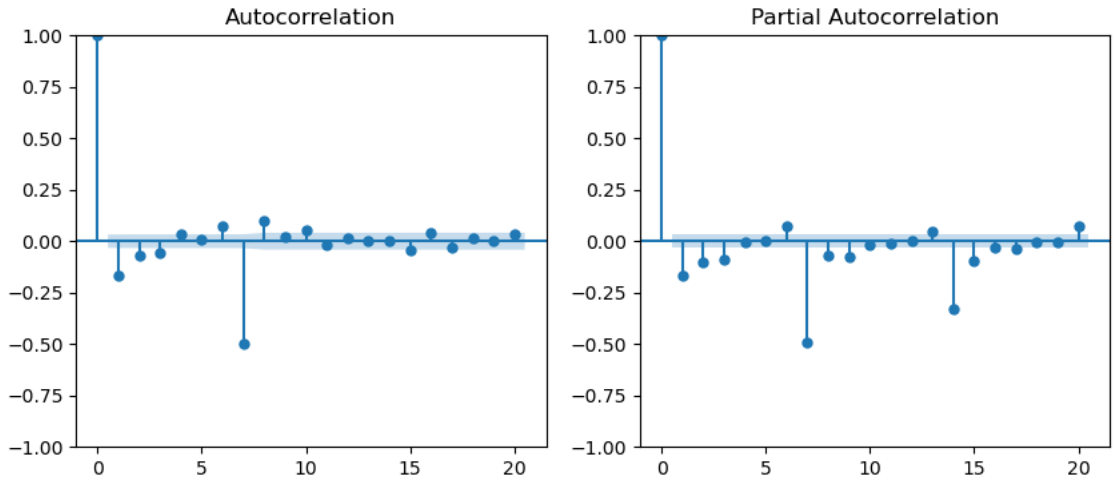

Figure 13: ACF and PACF Plots of Differenced Data

We decided to use **SARIMA(0, 1, 1)x(0, 1, 1, 7)** since we differenced twice with one being seasonal and we used MA terms to stabilize data both seasonal and regularly. Fitting of the model to the train set are as follows:

SARIMAX Results

Dep. Variable:	consumption	No. Observations:	3653
Model:	SARIMAX(0, 1, 1)x(0, 1, 1, 7)	Log Likelihood	-46456.682
Date:	Sun, 26 May 2024	AIC	92921.364
Time:	19:46:46	BIC	92946.169
Sample:	01-01-2011	HQIC	92930.199
	- 12-31-2020		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	4.8463	178.805	0.027	0.978	-345.605	355.298
ma.L1	-0.2675	0.017	-15.336	0.000	-0.302	-0.233
ma.S.L7	-0.8950	0.012	-74.652	0.000	-0.919	-0.872
sigma2	1.041e+10	4.92e-05	2.12e+14	0.000	1.04e+10	1.04e+10

Ljung-Box (L1) (Q):	2.68	Jarque-Bera (JB):	5766.51
Prob(Q):	0.10	Prob(JB):	0.00
Heteroskedasticity (H):	1.04	Skew:	-1.04
Prob(H) (two-sided):	0.54	Kurtosis:	8.80

Figure 14: SARIMA Model Fitting Results on the Train Set

Both MA terms are significant after fitting and intercept term is insignificant, but we disregard it and the final equation becomes:

$$y_t = 4.8463 + \varepsilon_t - 0.2675\varepsilon_{t-1} - 0.8950\varepsilon_{t-7}$$

This fit can be visualized in Fig. 15 where our model is accurate in forecasting the training set. We see that initially; the forecast was consistently overestimating the datapoints but became more and more accurate.

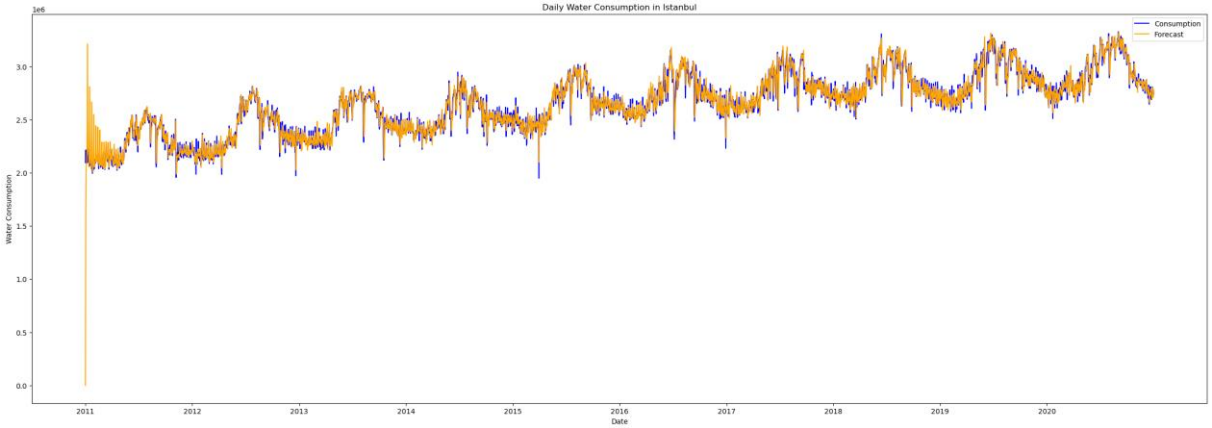

Figure 15: SARIMA model vs Observed Consumption in Train Set

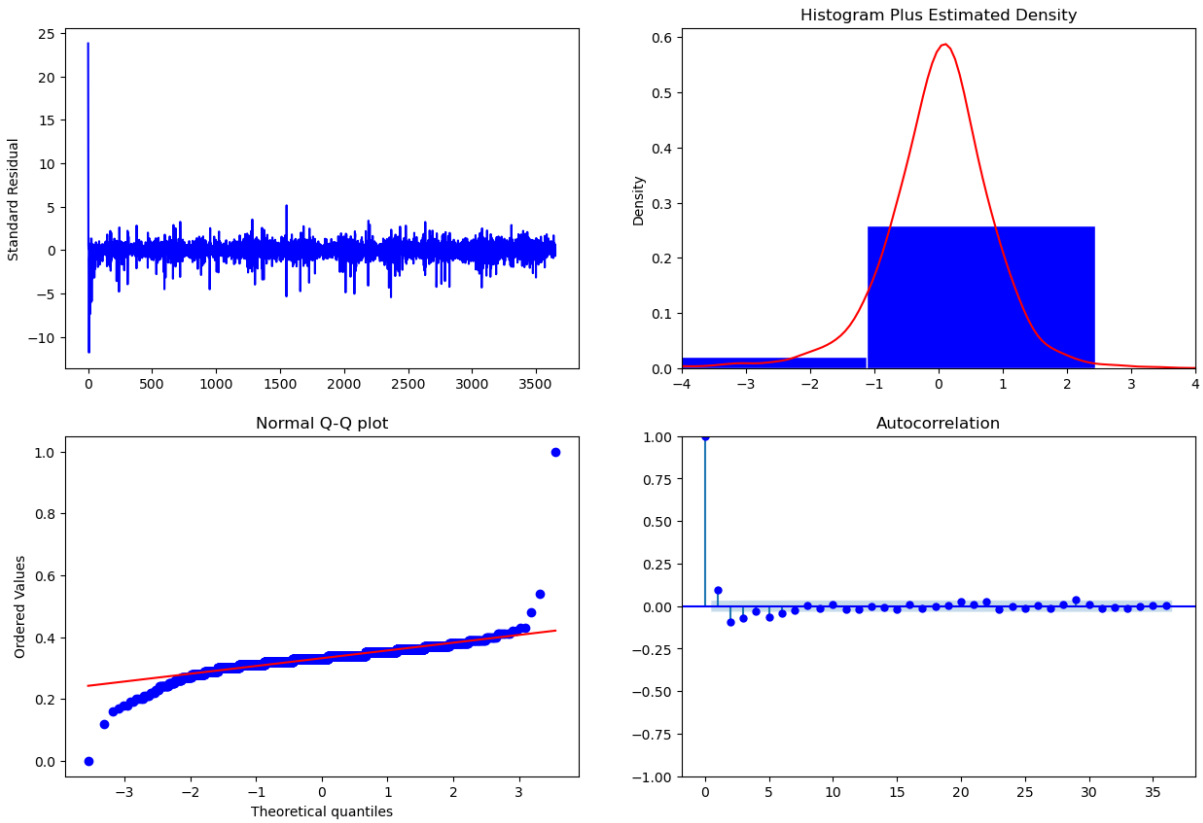

Figure 16: Residual Plots of SARIMA Model

As we can observe in Fig. 16, the residual diagnostics provide sufficient visual information in concluding the residual distribution is close to normal. We almost eliminated autocorrelation and residual quantiles match with normal line. We can confidently rely on our forecast model.

```
MSE: 7775245380.82
RMSE: 88177.35
MAE: 56004.25
MAPE: 2.14%
```

Training set error metrics also show that our model is successful in fitting without any significant error as MAPE is 2.14%. We can move on with investigating test set performance.

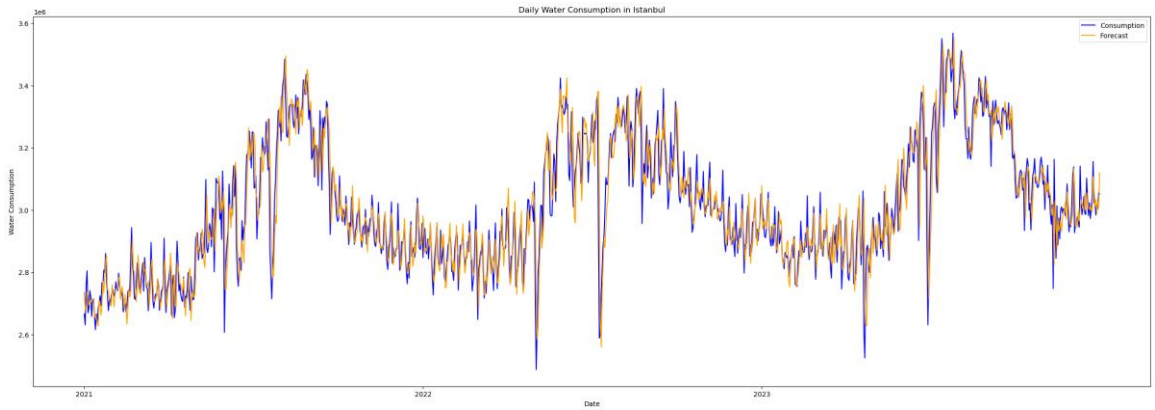

Figure 17: SARIMA Model vs Observed Test Data

```
MSE: 6554091618.02
RMSE: 80957.34
MAE: 57513.26
MAPE: 1.92%
```

Fig. 17 shows that our constructed SARIMA model carries its accuracy to the test set as forecast line is incredibly close to the observation line. This success can also be seen in the error metrics as we have even lower RMSE and an incredible MAPE with 1.92%.

Triple Exponential Smoothing

We first looked for the optimal parameters α, β, θ that gives us minimum error rate. And then we used that model to make our predictions. The top 10 optimal parameters can be seen in Fig. 18.

	ALPHA	BETA	GAMMA	RMSE	MAPE	MAE	MSE
78	0.7	0.1	0.7	101887.432952	2.794341	76490.557323	1.038105e+10
52	0.5	0.1	0.5	102380.665481	2.817962	77180.944537	1.048180e+10
77	0.7	0.1	0.5	102588.405179	2.820038	77155.628652	1.052438e+10
79	0.7	0.1	0.9	102646.393487	2.827955	77521.686246	1.053628e+10
51	0.5	0.1	0.3	103335.731228	2.844288	77788.898913	1.067827e+10
53	0.5	0.1	0.7	104382.630170	2.881957	79140.960982	1.089573e+10
26	0.3	0.1	0.3	106615.648212	2.914155	79784.496611	1.136690e+10
76	0.7	0.1	0.3	105972.778942	2.926643	80124.902654	1.123023e+10
104	0.9	0.1	0.9	107288.296421	2.955291	80894.230809	1.151078e+10
54	0.5	0.1	0.9	108037.435294	2.982028	82085.239329	1.167209e+10

Figure 18: Top 10 Triple Exponential Smoothing Forecasts with Metrics

We chose the best model to make our predictions. The predictions for Triple exponential smoothing can be seen in Fig.19. where the resulting prediction did a good job capturing the movement of our data, but we did better predictions in other models.

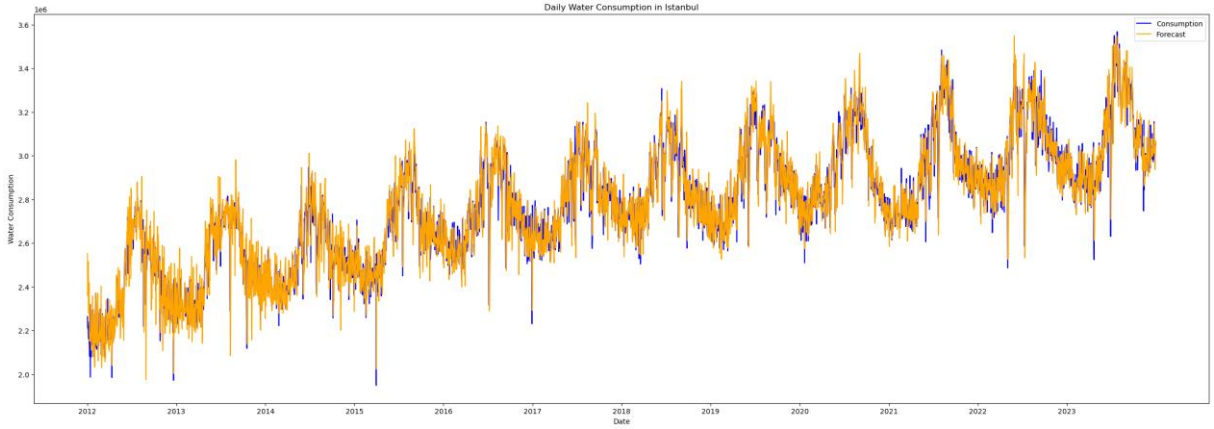

Figure 19: Best Triple Exponential Smoothing Forecast vs Observed Consumption Data

Also, when we check our residual graphs in Fig. 20, we can see that residuals are close to the normal but not close enough. Moreover, ACF graph helps us observe that there are some seasonal autocorrelations between the residuals, so we shouldn't rely on this model.

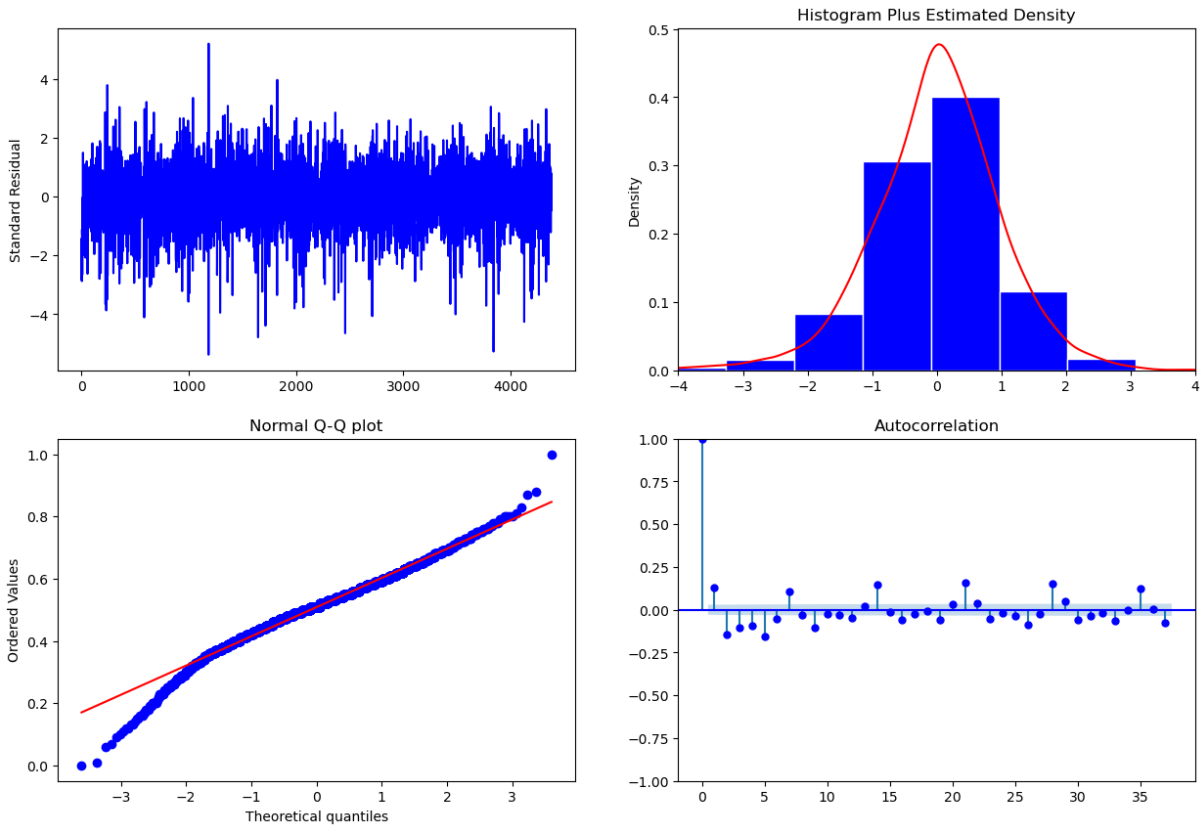

Figure 20: Residual Plots of the Triple Exponential Smoothing

```
MSE: 10381048993.466087
RMSE: 101887.43295159658
MAE: 76490.55732278358
MAPE: 2.7943412733637127%
```

Left hand side screenshot of the error metrics validates our discussion that our model shouldn't be chosen, error metrics are high compared to the other models. But generally, the resulting metrics are not far away from the observed data.

Ordinary Least Squares Regression Model

After working only with our observed dataset, we are now moving on with more complex forecasting methods by using ordinary least squares regression with the predictors described in Section 2. We now start by fitting our model to the DataFrame constructed with consumption dataset and predictors.

After fitting, we obtain the regression results in Fig. 21. The fit is successful because of the probability of the F-statistic is zero. Also, R-squared value is 0.937 which means our model can explain 93.7% of the variability of the data. To comment on the predictors, we can observe that the predictors *intercept, May and October* are insignificant according to the 5% significance level. The remaining predictors are significant. We may suspect overfitting with the magnitude of the R-squared statistic and the number of predictors used but this possibility is not clear as we need to progress and see the test set performance.

OLS Regression Results

Dep. Variable:	consumption	R-squared:	0.937
Model:	OLS	Adj. R-squared:	0.937
Method:	Least Squares	F-statistic:	2250.
Date:	Sat, 01 Jun 2024	Prob (F-statistic):	0.00
Time:	15:18:04	Log-Likelihood:	-45908.
No. Observations:	3653	AIC:	9.187e+04
Df Residuals:	3628	BIC:	9.202e+04
Df Model:	24		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.354e+05	1.48e+05	0.918	0.359	-1.54e+05	4.25e+05
temp	5790.4641	425.012	13.624	0.000	4957.178	6623.750
rain	-4.019e+04	2492.891	-16.124	0.000	-4.51e+04	-3.53e+04
popul	0.0580	0.011	5.160	0.000	0.036	0.080
YearTrend	1.889e+04	2413.757	7.827	0.000	1.42e+04	2.36e+04
Tuesday	-1.181e+04	4342.066	-2.720	0.007	-2.03e+04	-3297.777
Wednesday	1.689e+04	4401.968	3.837	0.000	8259.373	2.55e+04
Thursday	2.651e+04	4381.646	6.049	0.000	1.79e+04	3.51e+04
Friday	1.055e+04	4354.644	2.423	0.015	2011.799	1.91e+04
Saturday	3.231e+04	4378.204	7.380	0.000	2.37e+04	4.09e+04
Sunday	6.014e+04	4369.055	13.765	0.000	5.16e+04	6.87e+04
February	-1.697e+04	5753.087	-2.950	0.003	-2.82e+04	-5690.300
March	-2.415e+04	5739.090	-4.208	0.000	-3.54e+04	-1.29e+04
April	-2.311e+04	6210.852	-3.720	0.000	-3.53e+04	-1.09e+04
May	8847.3192	7313.347	1.210	0.226	-5491.362	2.32e+04
June	3.627e+04	8885.803	4.082	0.000	1.88e+04	5.37e+04
July	4.304e+04	9771.601	4.405	0.000	2.39e+04	6.22e+04
August	2.836e+04	9829.133	2.886	0.004	9092.890	4.76e+04
September	3.059e+04	8877.164	3.446	0.001	1.32e+04	4.8e+04
October	1364.1793	7162.709	0.190	0.849	-1.27e+04	1.54e+04
November	1.277e+04	6338.807	2.014	0.044	337.986	2.52e+04
December	2.105e+04	5782.673	3.641	0.000	9715.707	3.24e+04
lag_1	0.5229	0.012	41.933	0.000	0.498	0.547
lag_7	0.0197	0.009	2.162	0.031	0.002	0.037
lag_365	0.0091	0.002	3.945	0.000	0.005	0.014

Omnibus:	951.682	Durbin-Watson:	1.787
Prob(Omnibus):	0.000	Jarque-Bera (JB):	73713.744
Skew:	0.140	Prob(JB):	0.00
Kurtosis:	25.005	Cond. No.	1.96e+09

Figure 21: OLS Regression Results in Training Set

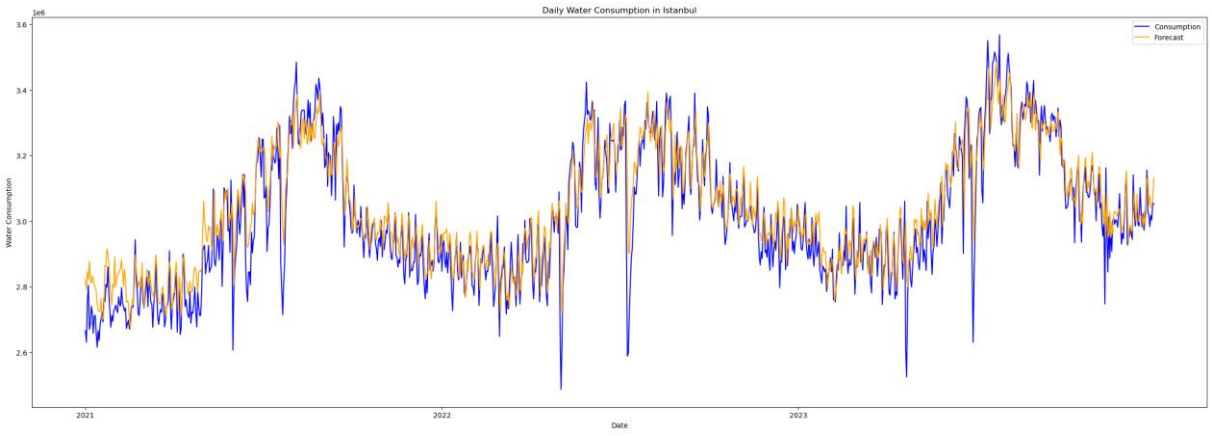

Figure 22: Regression Forecast vs Consumption Training Set

We can furthermore visualize the success of regression fitting on the training set by using Fig. 22. To further analyze the fitting performance of the training set, the residuals are close to normal as seen in Fig. 23. Even in the first lags, there is still some autocorrelation visible. But in overall, we can say that our residuals are not autocorrelated.

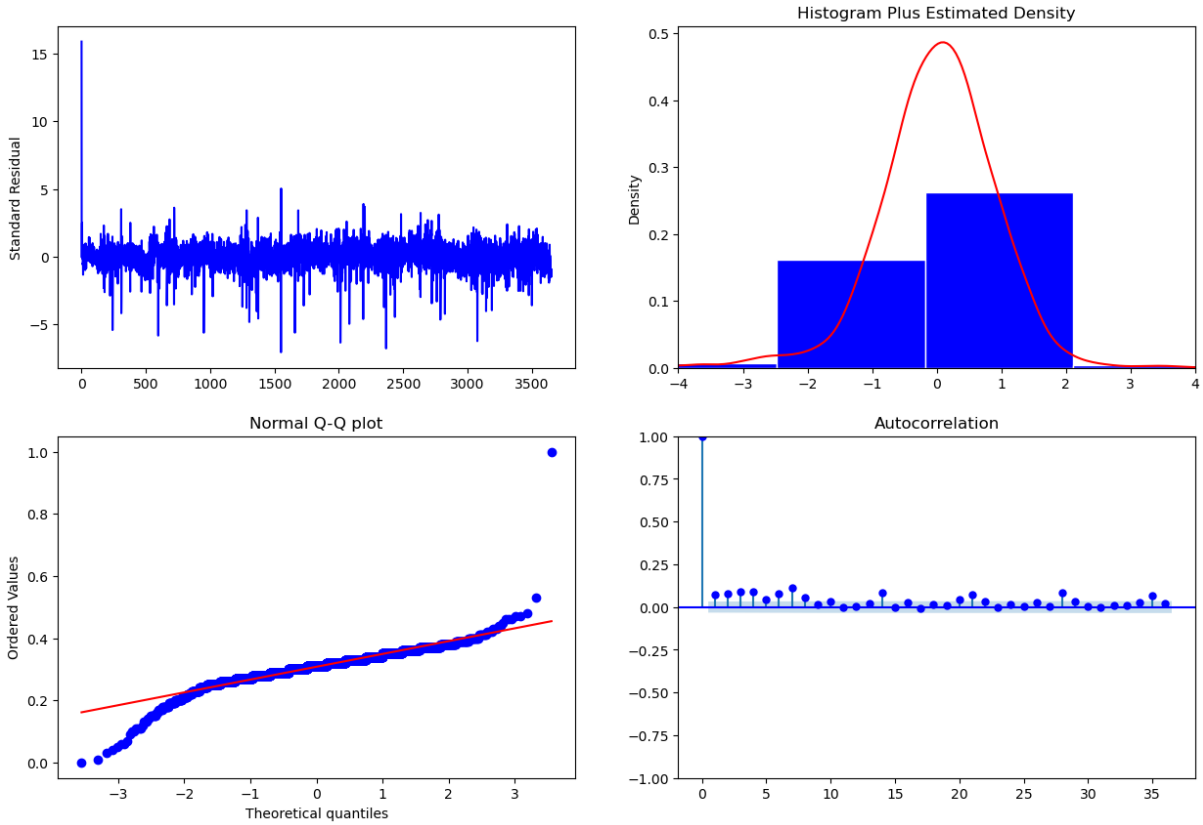

Figure 23: Residual Plots of Regression Forecast

```
MSE: 4822535312.78
RMSE: 69444.48
MAE: 48730.26
MAPE: 1.85%
```

The error metrics of the training set are significantly lower than the previous forecasting models as seen on the left. We observe great decrease in both RMSE and MAPE. We can confidently move on with testing our regression with the test set.

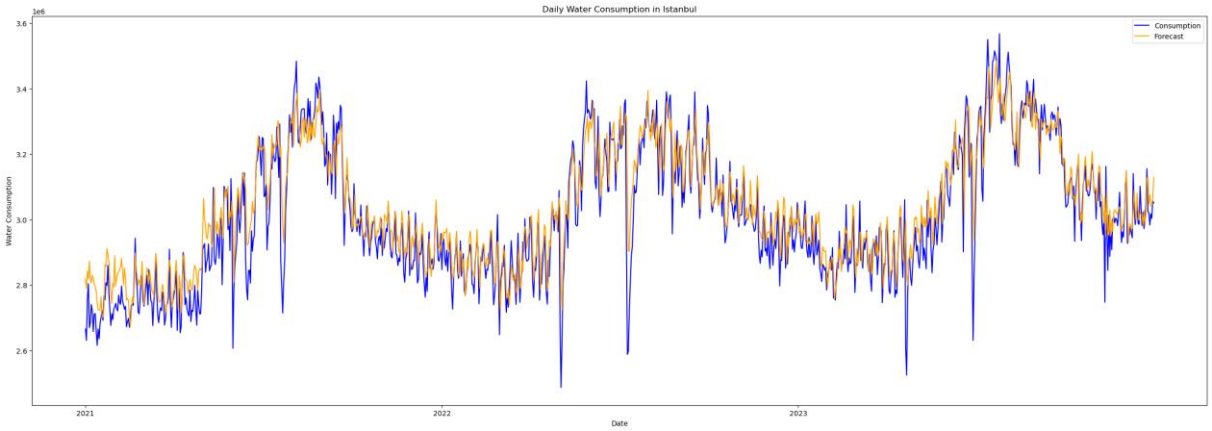

Figure 24: Regression Model vs Observed Test Set

```
MSE: 6451211063.18
RMSE: 80319.43
MAE: 58572.34
MAPE: 1.98%
```

Both Fig. 24 and the error metrics show outstanding results as the line representing the forecast almost perfectly aligns with test set data line. The accuracy is supported by low error metrics as we are now below 2% in terms of test set MAPE. We can conclude that that possibility of an overfitting is not significant as the ratio between number of predictors used, and the size of the dataset is too low, and the accuracy continues between training and set sets.

LASSO Reduction on our Regression Model

Even though we did not observe any major overfitting signs, we thought that confirming the possible existence of overfitting and reducing our model to fit by using less predictors is better for us to understand the behavior of both Lasso technique and our model.

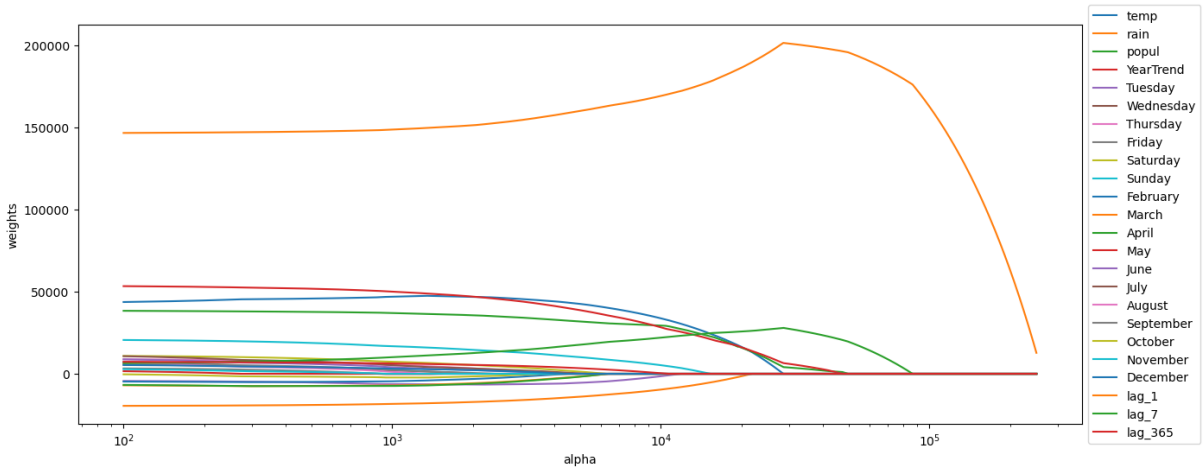

Figure 25: Behavior of Regression Coefficients Subject to Ranging Alpha Values

After applying a penalty term α to nonzero coefficients, we see the behavior of such coefficients with increasing α values on Fig. 25. We can conclude that most of the predictors play a crucial role in water consumption forecast. Furthermore, the existence of rain is so important that it survives until the end of our range of α values.

Following the plotting, we used 5-fold cross-validation to optimally find the best α and we found $\alpha^*=365.465$ with a CV score of 0.818. This means the slightly reduced model performs well.

```
coef of temp >> 45566.18346866848
coef of rain >> -19192.15070191835
coef of popul >> 37882.87362197383
coef of YearTrend >> 52293.52763964241
coef of Tuesday >> -4980.352100334398
coef of Wednesday >> 4448.898199287954
coef of Thursday >> 7760.915180404922
coef of Friday >> 2204.327138609404
coef of Saturday >> 9783.925337922468
coef of Sunday >> 19400.066208841672
coef of February >> -5034.972031997655
coef of March >> -7494.0738838087245
coef of April >> -7510.303949478556
coef of May >> 0.0
coef of June >> 6459.86788989768
coef of July >> 8043.0671767183485
coef of August >> 3939.322188411923
coef of September >> 4869.69735269151
coef of October >> -1627.4048140582888
coef of November >> 1382.0514331142215
coef of December >> 4268.234075076148
coef of lag_1 >> 147377.57479816215
coef of lag_7 >> 7803.2945530031475
coef of lag_365 >> 6951.36040023115
```

Figure 26: Coefficients after adjustment of Alpha

We see that only coefficient of May vanishes in Fig. 26. We then adjust the formula for regression fitting and here are the results in Fig. 27:

OLS Regression Results

Dep. Variable:	consumption	R-squared:	0.937
Model:	OLS	Adj. R-squared:	0.937
Method:	Least Squares	F-statistic:	2348.
Date:	Sat, 01 Jun 2024	Prob (F-statistic):	0.00
Time:	15:24:43	Log-Likelihood:	-45909.
No. Observations:	3653	AIC:	9.187e+04
Df Residuals:	3629	BIC:	9.201e+04
Df Model:	23		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.364e+05	1.48e+05	0.925	0.355	-1.53e+05	4.26e+05
temp	6070.4791	356.476	17.029	0.000	5371.565	6769.393
rain	-4.019e+04	2493.049	-16.121	0.000	-4.51e+04	-3.53e+04
popul	0.0576	0.011	5.129	0.000	0.036	0.080
YearTrend	1.878e+04	2412.230	7.786	0.000	1.41e+04	2.35e+04
Tuesday	-1.171e+04	4341.612	-2.698	0.007	-2.02e+04	-3202.251
Wednesday	1.698e+04	4401.562	3.859	0.000	8354.286	2.56e+04
Thursday	2.656e+04	4381.677	6.062	0.000	1.8e+04	3.52e+04
Friday	1.064e+04	4354.225	2.444	0.015	2106.897	1.92e+04
Saturday	3.234e+04	4378.438	7.385	0.000	2.38e+04	4.09e+04
Sunday	6.006e+04	4368.787	13.747	0.000	5.15e+04	6.86e+04
February	-1.986e+04	5232.532	-3.796	0.000	-3.01e+04	-9604.746
March	-2.764e+04	4961.141	-5.572	0.000	-3.74e+04	-1.79e+04
April	-2.764e+04	4954.702	-5.578	0.000	-3.74e+04	-1.79e+04
June	2.836e+04	6019.602	4.712	0.000	1.66e+04	4.02e+04
July	3.426e+04	6539.793	5.239	0.000	2.14e+04	4.71e+04
August	1.951e+04	6563.442	2.973	0.003	6643.918	3.24e+04
September	2.272e+04	6038.041	3.762	0.000	1.09e+04	3.46e+04
October	-4693.6085	5121.813	-0.916	0.360	-1.47e+04	5348.310
November	7971.9447	4947.692	1.611	0.107	-1728.588	1.77e+04
December	1.759e+04	5023.179	3.501	0.000	7738.436	2.74e+04
lag_1	0.5233	0.012	41.978	0.000	0.499	0.548
lag_7	0.0217	0.009	2.424	0.015	0.004	0.039
lag_365	0.0089	0.002	3.893	0.000	0.004	0.013

Omnibus:	956.311	Durbin-Watson:	1.786
Prob(Omnibus):	0.000	Jarque-Bera (JB):	74768.945
Skew:	0.152	Prob(JB):	0.00
Kurtosis:	25.162	Cond. No.	1.96e+09

Figure 27: OLS Regression Fit into the Training Set After Lasso (May not included)

The regression is once again significant based on F-statistic probability and we observe no change in both R-squared and AIC to compare multiple metrics, so we can comment the same as the unreduced regression. To comment on the predictors, we can observe that the predictors *intercept, October and November* are insignificant according to the 5% significance level. The remaining predictors are significant.

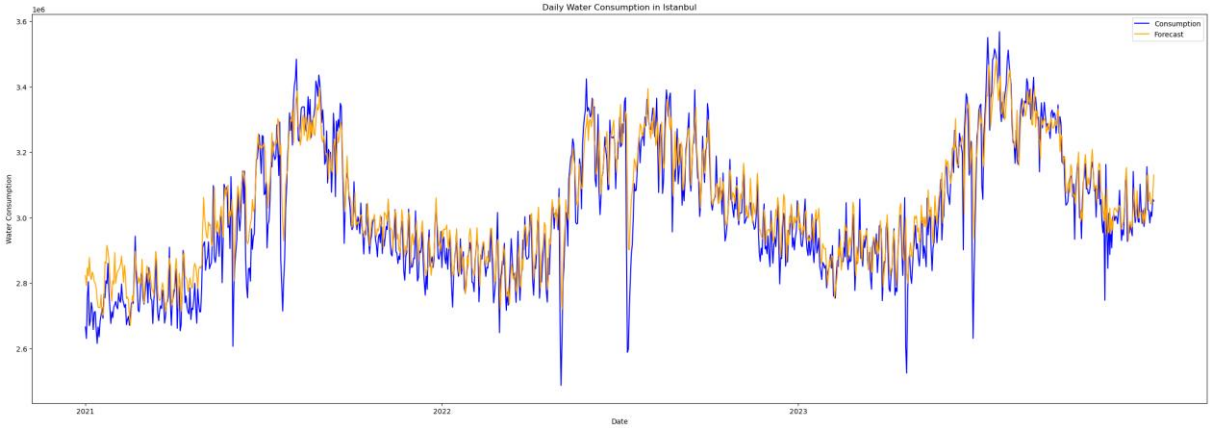

Figure 28: Reduced Regression Forecast vs Observed Test Set

```
MSE: 6429152761.74
RMSE: 80182.00
MAE: 58535.75
MAPE: 1.98%
```

Fig. 28 once again shows almost identical forecast versus observation lines. The minority of the model reduction may be the reason behind this. The error metrics on the left confirms this as RMSE and MAPE are almost identical. This is absurdly close as we are working with water consumption quantities more than a million.

K-Nearest Neighbor (KNN) Model

We experimented with 3 different KNN models with K= 1, 5, 10. Figures 29, 30 and 31 show that all three models provide nearly equal predictions in the test set but all three have a common shortcoming: they cannot exceed predictions surpassing the maximum observation in the training set. This results in peaks not being forecasted.

1-NN:

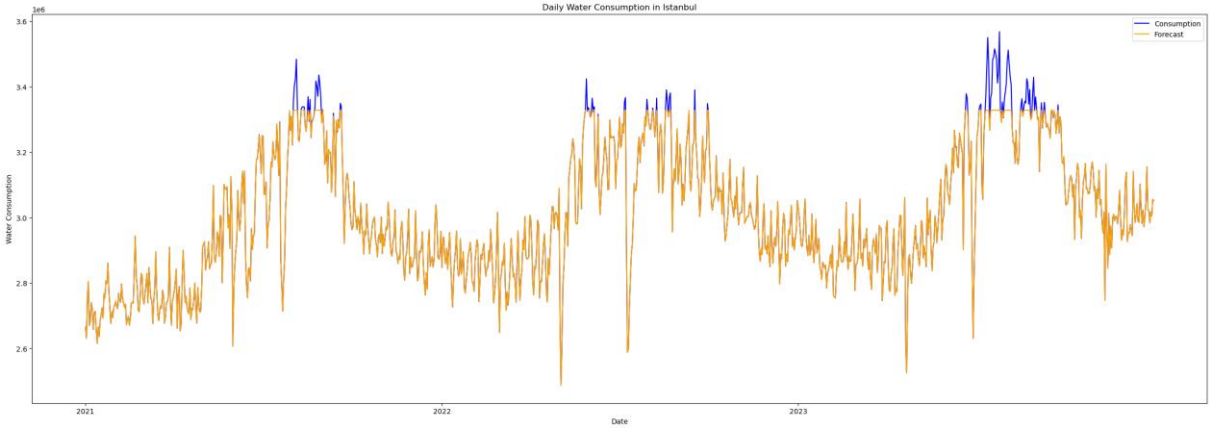

Figure 29: 1-NN Model vs Observed Test Set

```
MSE: 519246673.10
RMSE: 22786.98
MAE: 4955.42
MAPE: 0.15%
```

5-NN:

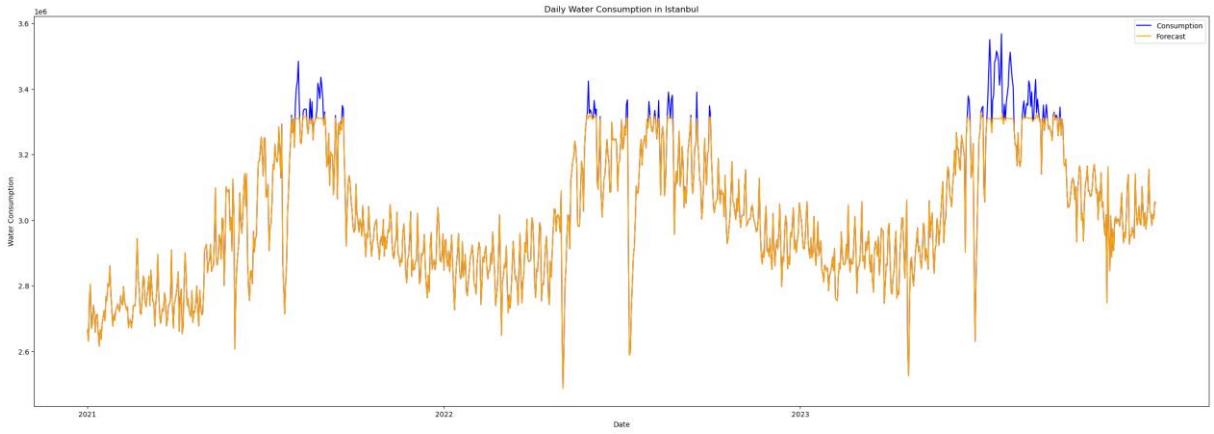

Figure 30: 5-NN Model vs Observed Test Set

```
MSE: 704218530.94
RMSE: 26537.12
MAE: 6069.49
MAPE: 0.18%
```

10-NN:

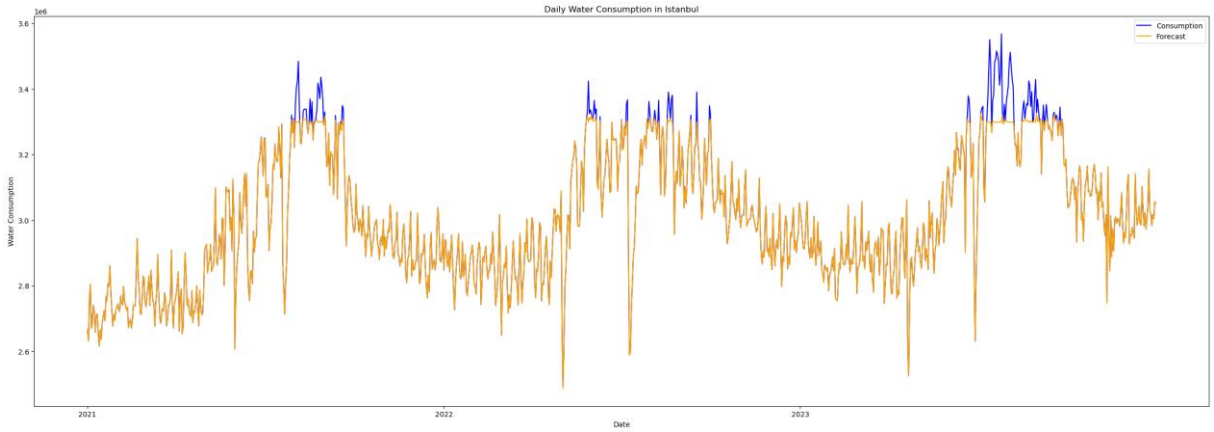

Figure 31: 10-NN Model vs Observed Test Set

```
MSE: 833570305.18
RMSE: 28871.62
MAE: 6964.97
MAPE: 0.20%
```

Among all three models, we observed the best error metrics from **1-NN** but as we said, this set of models lack the ability to observe data with increasing trend. We should not rely on this model on long term datasets with increasing trends.

Decision Tree Regressor

We fitted a decision tree with our train data and test data. We can see that we managed to capture the trend and seasonality in most cases, but we can see that this model is not good at capturing rapid decreases in some periods as we can see in Fig. 32.

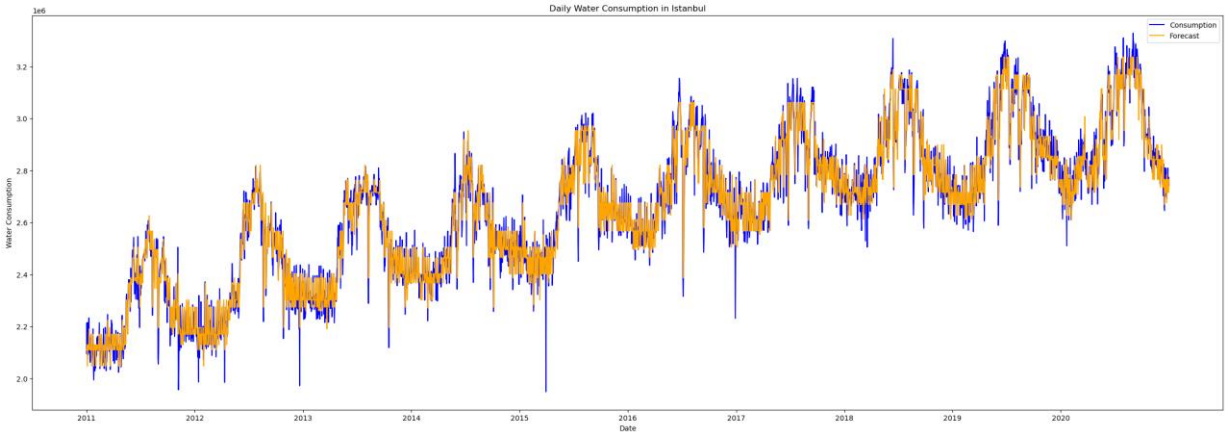

Figure 32: Decision Tree Regressor vs Observed Train Set

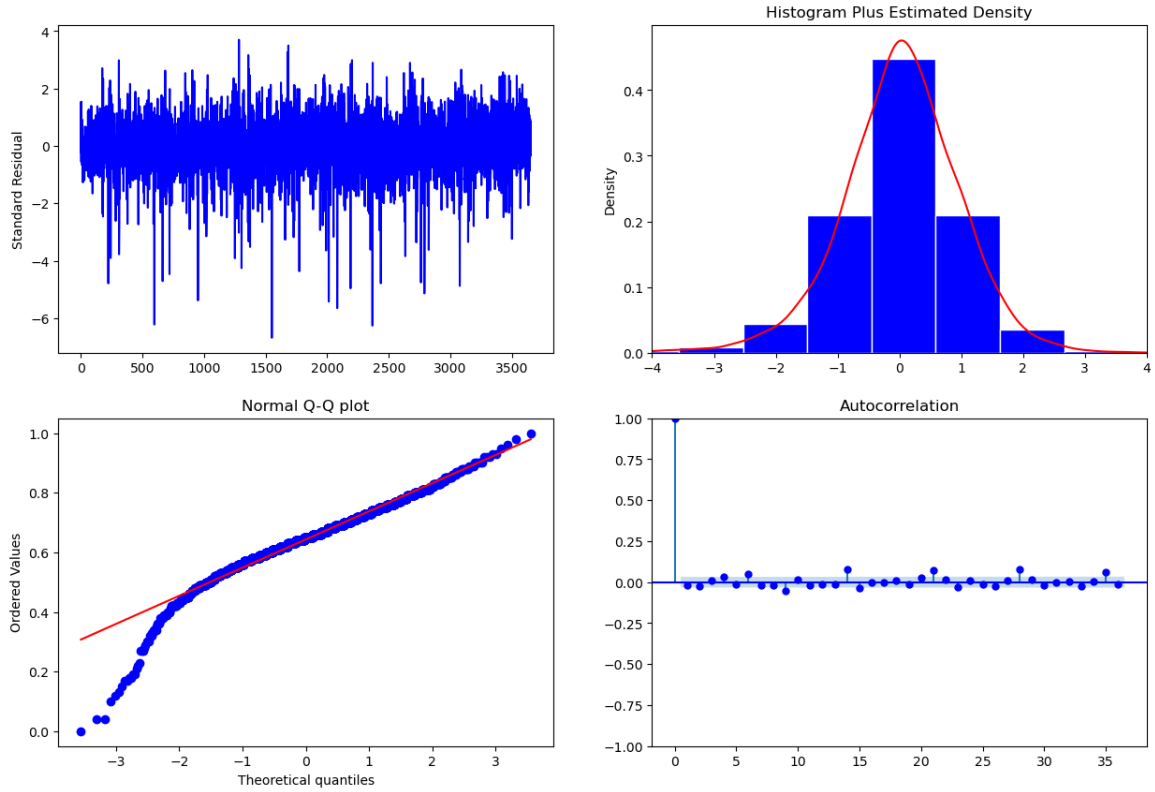

Figure 33: Residual Plots of Decision Tree Regressor

We can see in Fig. 33, our residuals are not autocorrelated which is good, and we can observe that residuals are close to normal which is also good. We can rely on this model.

*Resulting decision tree is on the next page due to its size.

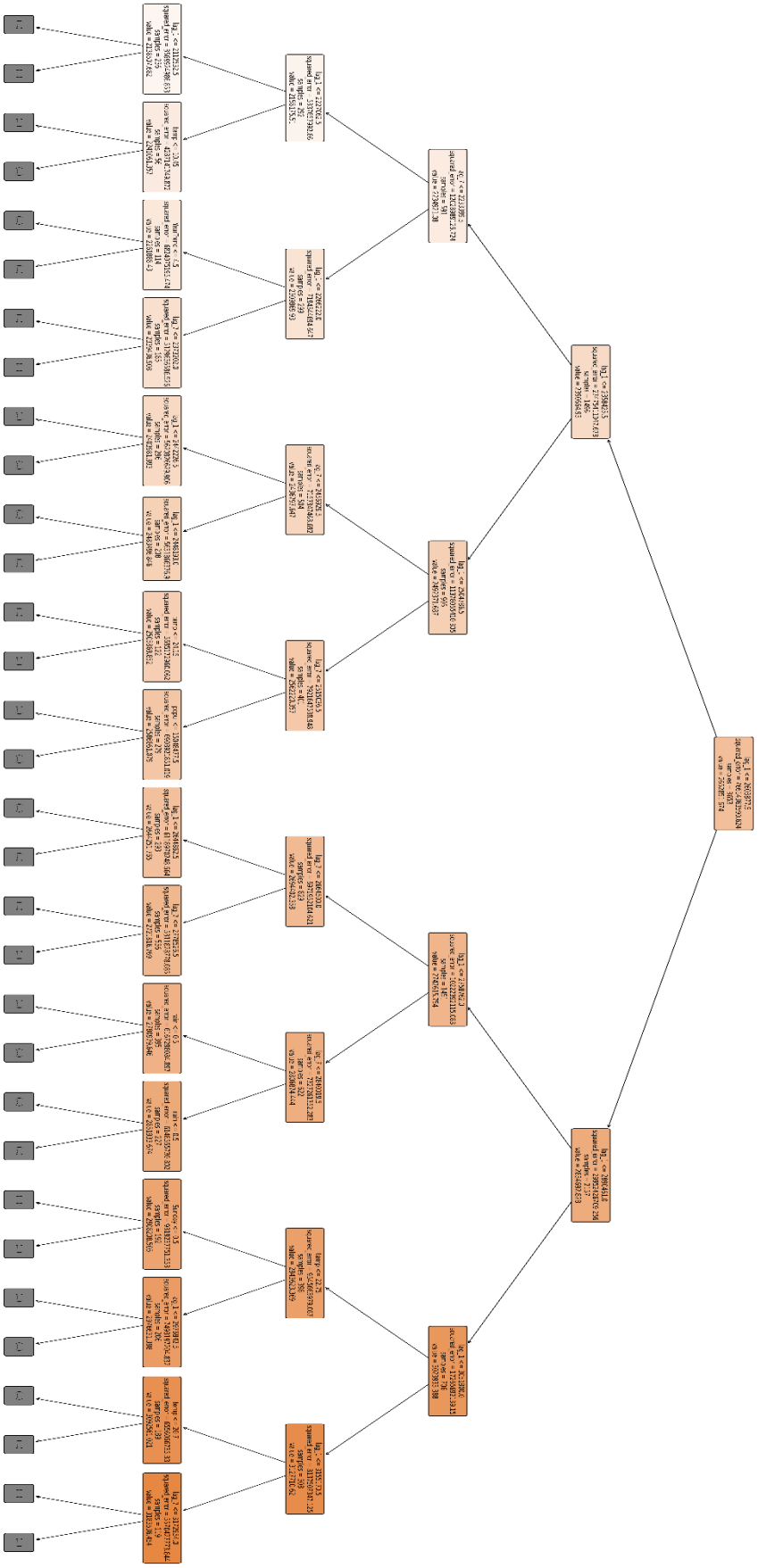

Figure 34: Decision Tree Explicit Form

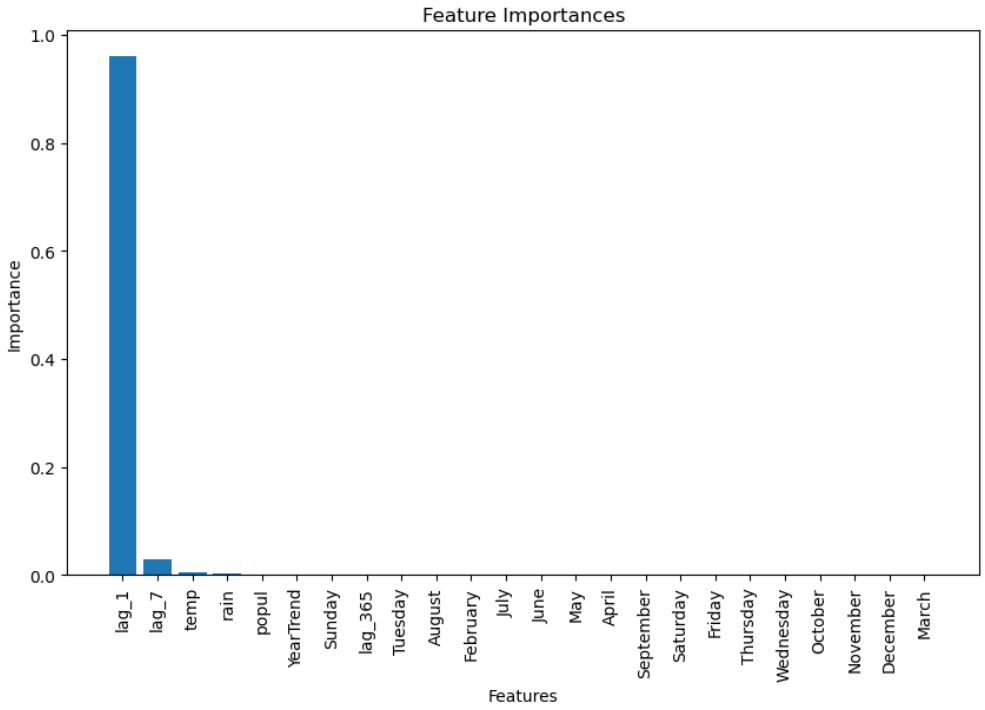

Figure 35: Importance Graph for Decision Tree Predictors

We can observe from the importance graph on Fig. 35, the most important predictor for the classification is the lag-1 predictor and flowing that lag-7.

Error metrics for train set:

```
MSE: 4586013069.127512
RMSE: 67720.10830711592
MAE: 50072.61122840787
MAPE: 1.903358280828546 3%
```

Error metrics for test set:

```
MSE: 13858387853.48282
RMSE: 117721.65414010636
MAE: 88779.48307555949
MAPE: 2.89069472675113%
```

We can observe that the decision tree model did worse on the test set. This is a clear sign for overfitting, and we can conclude that we shouldn't use this model. The difference between the error rates can be explained in error metrics above.

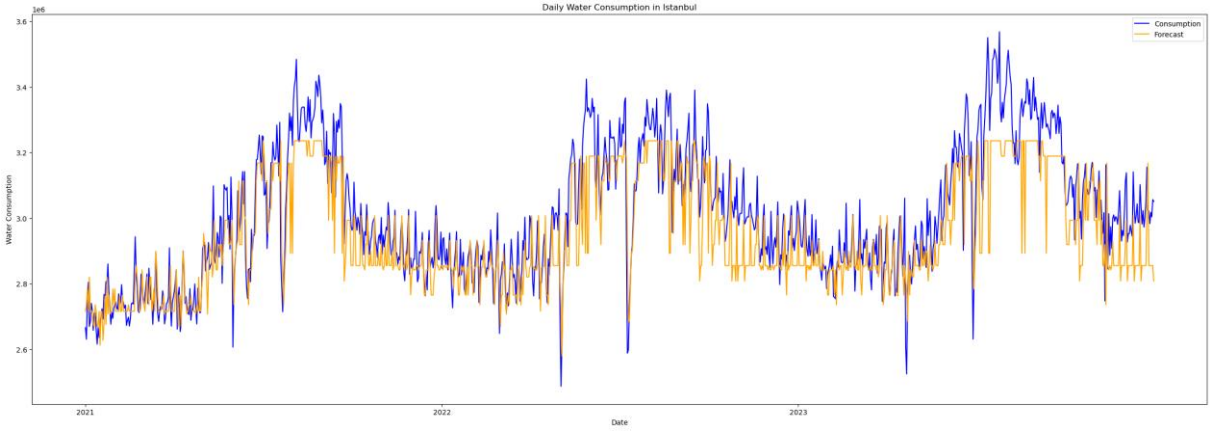

Figure 36: Decision Tree Regressor vs Observed Test Set

In Fig. 36, we can see that our model is able to capture the general movement of the data but makes wrong classifications in some obvious parts. The model is not able to capture

negative trends correctly and this is the reason of the difference between error metrics in test set and train set.

Random Forest

We fitted a random forest to our data with using five-fold cross validation. We can see in the Fig. 37 our model is able to capture the trend and seasonality in our data.

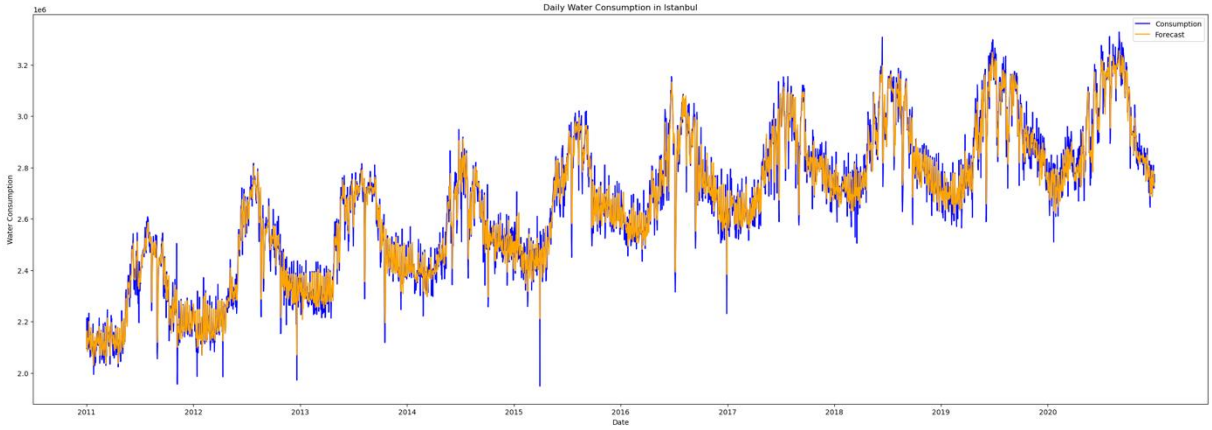

Figure 37: Random Forest vs Observed Training Set

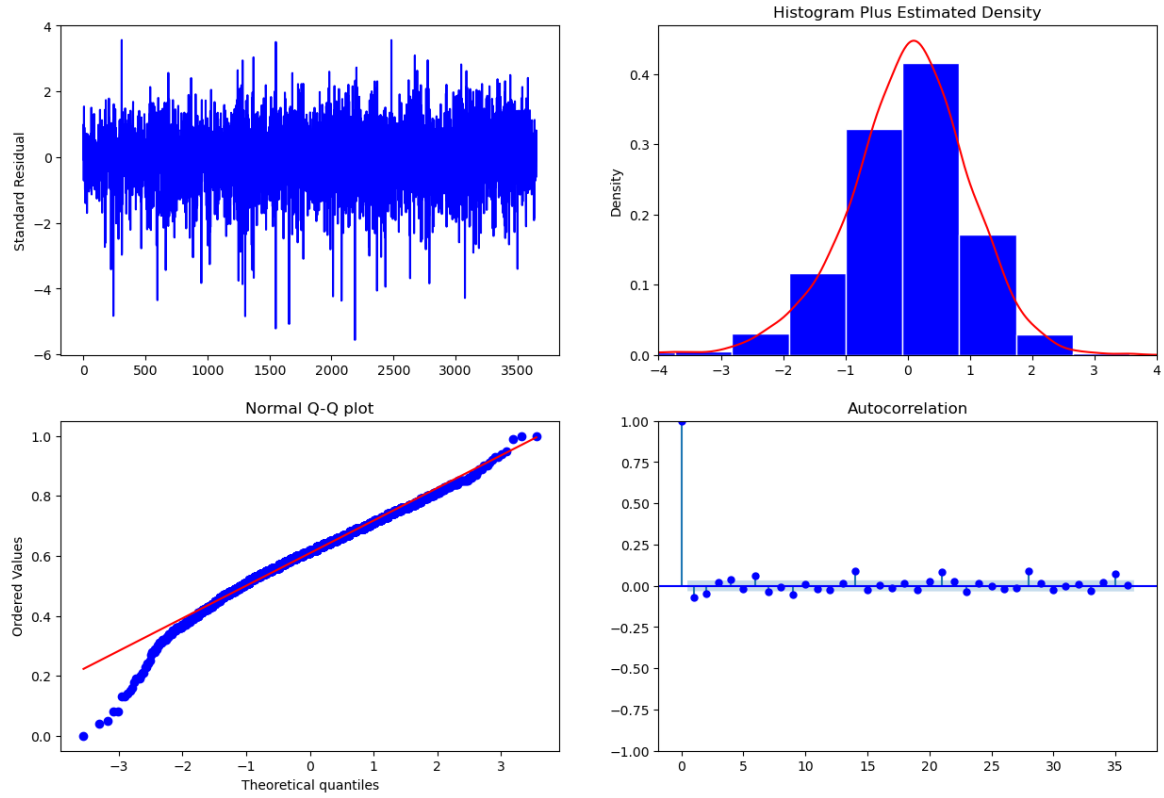

Figure38: Residual Plots of Random Forest

We can see in our residual graphs Fig. 38 our residuals are close to normal and no autocorrelated which is a desired result, we can understand that our model is good so far and we can rely on the random forest model.

```
MSE: 2551107460.358915
RMSE: 50508.488993028834
MAE: 38564.797351533896
MAPE: 1.46285595606446%
Best Depth:  8
```

We can see our error metrics for training set in the error metrics on the left. We found that the optimal depth of the random forest should be 8 and we constructed the model according to that.

As we can see in Fig. 39, random forest model couldn't be able to do good forecasts as in train set. We can see that our forecast in the test set is below the actual consumption almost all the days. This is a sign for overfitting, and we can conclude that from here we shouldn't use this model to make accurate forecasts.

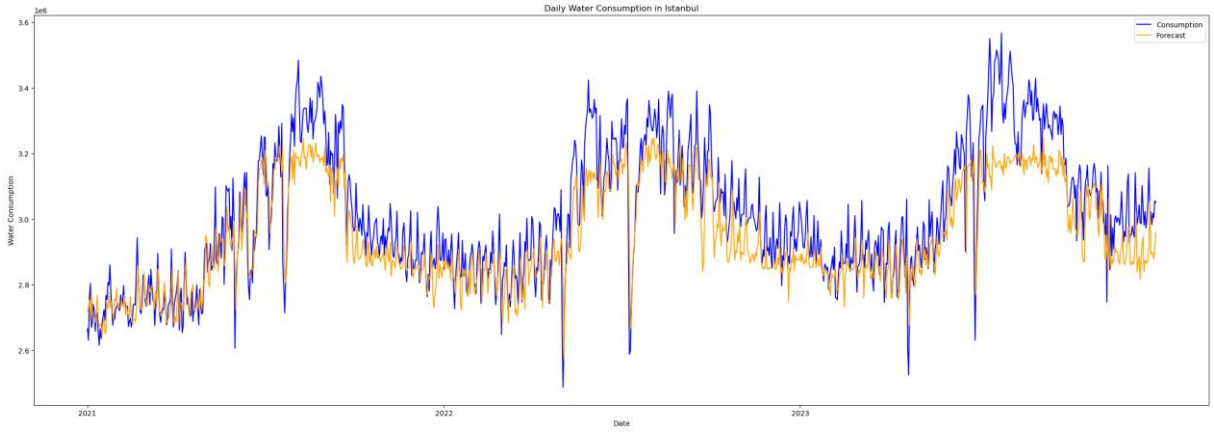

Figure 39: Random Forest vs Observed Test Set

```
MSE: 11784125746.657915
RMSE: 108554.71314806149
MAE: 84821.41103772682
MAPE: 2.753272080583888%
```

As we can see on the left, our model's error metrics is worse in the test set as we claimed visually.

Neural Network

We fitted a neural network with 2-layers and to find the optimal epoch size, batch size and the activation function, we trained multiple networks to find the best parameters that gives the minimum error. After a long time of training multiple networks, we get the output below:

```
Optimal epoch size:  110
Optimal batch size:  10
Optimal activation function:  relu
```

With using those parameters, we trained our network with train set. We can observe that in Fig 39, our neural model was able to capture the general movement of our data like seasonality and trend, but the model is not able to capture rapid decreases.

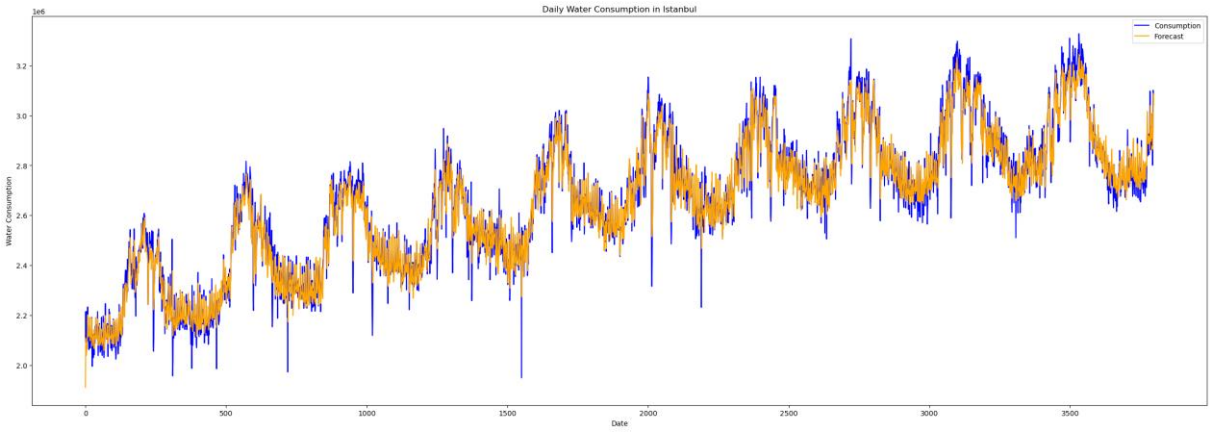

Figure 39: Neural Network vs Observed Training Set

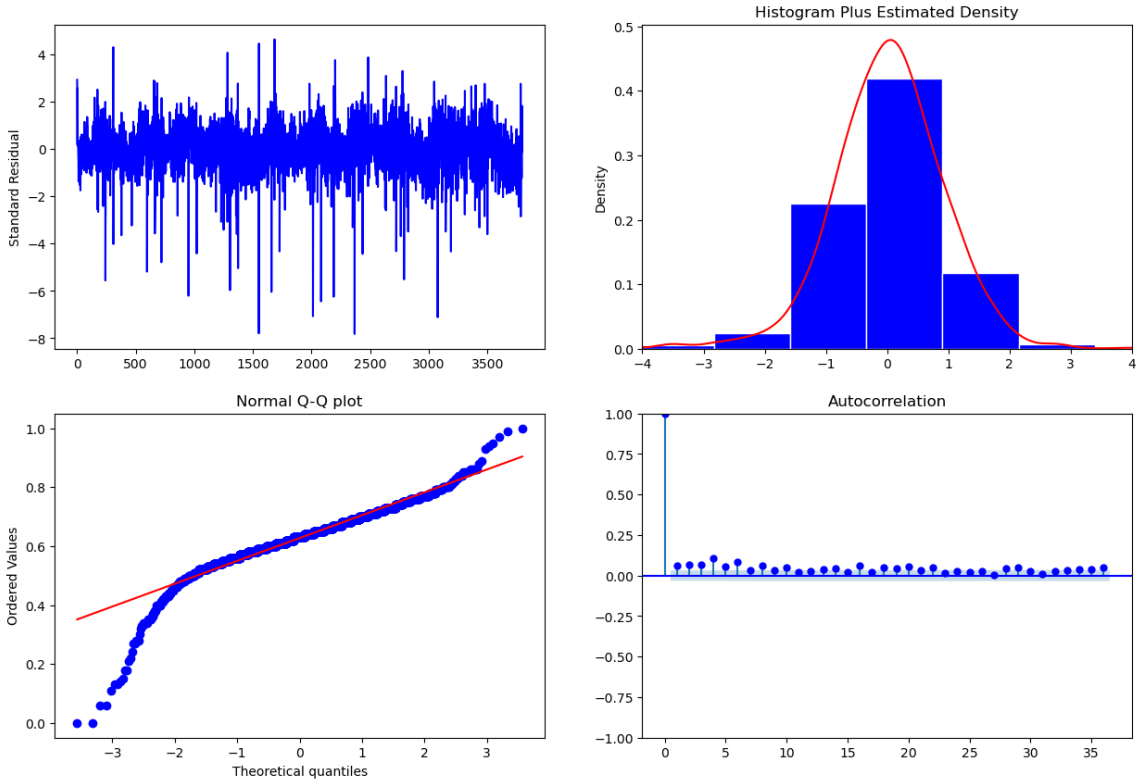

Figure 40 Residual Plot of Neural Network

Also we can see in Fig. 40, our residuals are close to normal and no autocorrelation observed in the plots which is a desired property, we can rely on our model.

```
TRAIN MAPE:  1.667471404902847
TRAIN MSE:  3698481402.3812904
TRAIN RMSE:  60815.14122635325
```

We can see that in the error metrics, we had a very good metrics in the train set. We were expecting this since neural networks can capture more complex relations between the features.

As we can see in Fig .41, our model did a little worse on the test set. The forecasts are not able to capture the peaks of the data.

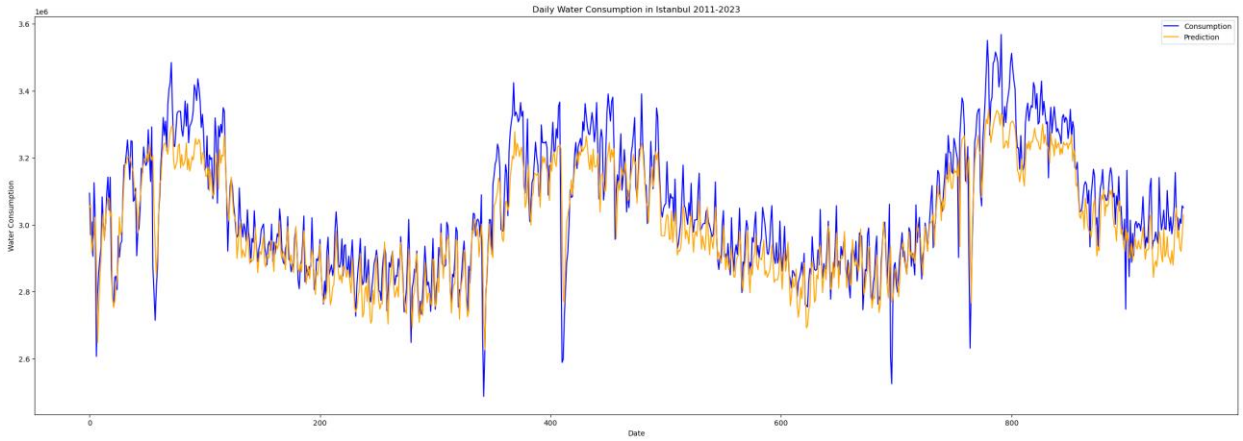

Figure 41: Neural Network vs Observed Test Set

```
MAPE_test = 1.976%
MSE_test = 6530944399.191
RMSE_test = 80814.259
```

We can assess in the error metrics on the left, we did worse on the test, and we can conclude that this is a sign for overfitting so we shouldn't use that model.

4. FORECASTING METHODS SUMMARY

Method	Spec.	RMSE Train	MAPE Train	RMSE Test	MAPE Test
Naive-1	-	82888.95	2.24%	-	-
Naive-7	-	123321.68	3.15%	-	-
Naive-365	-	158500.83	4.41%	-	-
MA-7	-	91875.08	2.44%	-	-
MA-30	-	104734.58	2.82%	-	-
SARIMA	(0,1,1)(0,1,1,7)	88177.35	2.14%	80957.34	1.92%
Triple Exponential Smoothing	$\alpha = 0.7$, $\beta = 0.1$, $\theta = 0.7$	101887.43	2.79%	-	-
Regression	Can be seen in Fig.	69444.48	1.85%	80319.43	1.98%
Reduced Lasso Regression	Can be seen in Fig.	-	-	80182.00	1.98%
1-KNN	-	-	-	22786.98	0.15%
Decision Tree	-	67720.11	1.90%	117721.65	2.89%
Random Forest	Best Depth = 8	50508.48	1.46%	108554.71	2.75%
Neural Network	Epoch Size = 110, Batch Size = 10, Activation Function = 'ReLu'	60815.14	1.67%	80814.26	1.98%

Throughout this section, we experimented with many forecasting methods from primal methods such as naive and all the way to complex models like neural networks. Every model had its own shortcomings that are explained further in their respective subsections. But to decide which model to proceed with in linear programming, we decided to use Lasso Reduced Regression. Even though 1-NN has the best error metrics, it lacks the ability to catch positive trends, which is not preferable in our case. We could also choose SARIMA, but it being linearly limited also does not allow us to use it. Final candidate was the neural network, but the difference in error metrics between training set and test set raises suspicion for a possible overfit and pushes us from this choice. In conclusion, we implemented Lasso reduced linear regression in our prescriptive problem.

5. PRESCRIPTIVE PROBLEM

Among the predictions we made, we thought that the best prediction method will be the reduced regression model with lasso (reduced indicator of May). Since the water distribution is a basic and very important service, we claimed that we should avoid underage as much as possible. Water demand should be met as much as possible. Therefore, we assigned some costs for underage and overage c_u , c_o respectively. We picked $c_u = 19$ and $c_o = 1$ and the reason of doing that is to reach the critical factor of $\%95 = \frac{c_u}{c_u + c_o}$, meaning that we aimed to meet demand in $\%95$ of the days. We constructed an LP model in order to implement these. Here is the LP model:

Decision variables:

$$q_j : \text{ value for the predictor } j \text{ , } \forall j \in P$$

Parameters:

$$d_i : \text{ daily water consumption for day } i \text{ , } \forall i \in N$$
$$z_i^+ : \text{ a representation for underage amount for day } i \text{ , } \forall i \in N$$
$$z_{\bar{i}} : \text{ a representation for overage amount for day } i \text{ , } \forall i \in N$$
$$c_u : \text{ underage cost}$$
$$c_o : overage\ cost$$

Objective function:

$$min\ \frac{1}{n}\sum_{i=1}^{n} c_u z_i^+ + c_o z_i^-$$

Subject to:

$$z_i^+ \geq d_i - \left(q_0 + \sum_{j=1}^{p} q_j x_j \right) \quad i = 1,2,\dots,n$$

$$z_i^- \geq \left(q_0 + \sum_{j=1}^{p} q_j x_j \right) - d_i \quad i = 1,2,\dots,n$$

$$z_i^+, z_i^- \geq 0 \quad i = 1,2,\dots,n$$

We then solve this linear problem using GurobiPy with our train set. Result of the LP:

```
Optimal Objective Value on Training Set:  127122.15019450258
```

*This is the cost with the assigned values $c_u = 19$ and $c_o = 1$, more realistic cost estimations can be done via adjusting the c_u and c_o accordingly.
And here are the values of our decision variables, predictors and the equation for the optimal order quantity:

```
Optimal Order Quantity: 512262.10 + 7069.22 * temp + -34556.37 * rain + 0.04 * popul + 26658.26 * YearTrend +
    -8886.25 * Tuesday + 11998.01 * 'Wednesday' + 24936.69 * 'Thursday' + 2384.30 * 'Friday' +
    24074.13 * 'Saturday' + 68721.63 * 'Sunday' + -39679.83 * 'February' + -41469.40 * 'March' +
    -52987.34 * 'April' + 37146.77 * 'June' + 27466.43 * 'July' + 15742.94 * 'August' +
    9287.96 * 'September' + -21845.45 * 'October' + -15248.57 * 'November' + 5112.59 * 'December' +
    0.48 * 'lag_1' + 0.02 * 'lag_7' + 0.02 * 'lag_365'
```

Using these predictions, we can test our model in the test set. Here are the results for test set:

```
Average backordering and holding Cost on Test Set:  153190.70734330913
Number of times overage happened:  1078
Number of times underage happened:  17
Percentage of meeting water consumption: % 98.45
```

We can see that in our results in %98.45 of the days there were no underage observed. Since our aim was to reach the demand as much as possible %98.45 is a good rate.

Lastly, we want to mention that this prediction model can be used to decide optimal order quantity for any day with selecting appropriate parameter. But the data we have should be updated simultaneously to get more accurate results for the future predictions.

6. REFERENCES

INDR422 Spring 24 Lecture Notes and Lab Materials

Municipality of Istanbul Dataset for daily water consumption and dam increases: https://data.ibb.gov.tr/dataset/istanbul-barajlarina-dusen-gunluk-toplam-yagis-miktari/resource/762b802e-c5f9-4175-a5c1-78b892d9764b

Visual Crossing website for daily temperature averages: https://www.visualcrossing.com

Turkish Statistical Institute for yearly population of Istanbul: https://biruni.tuik.gov.tr/medas/