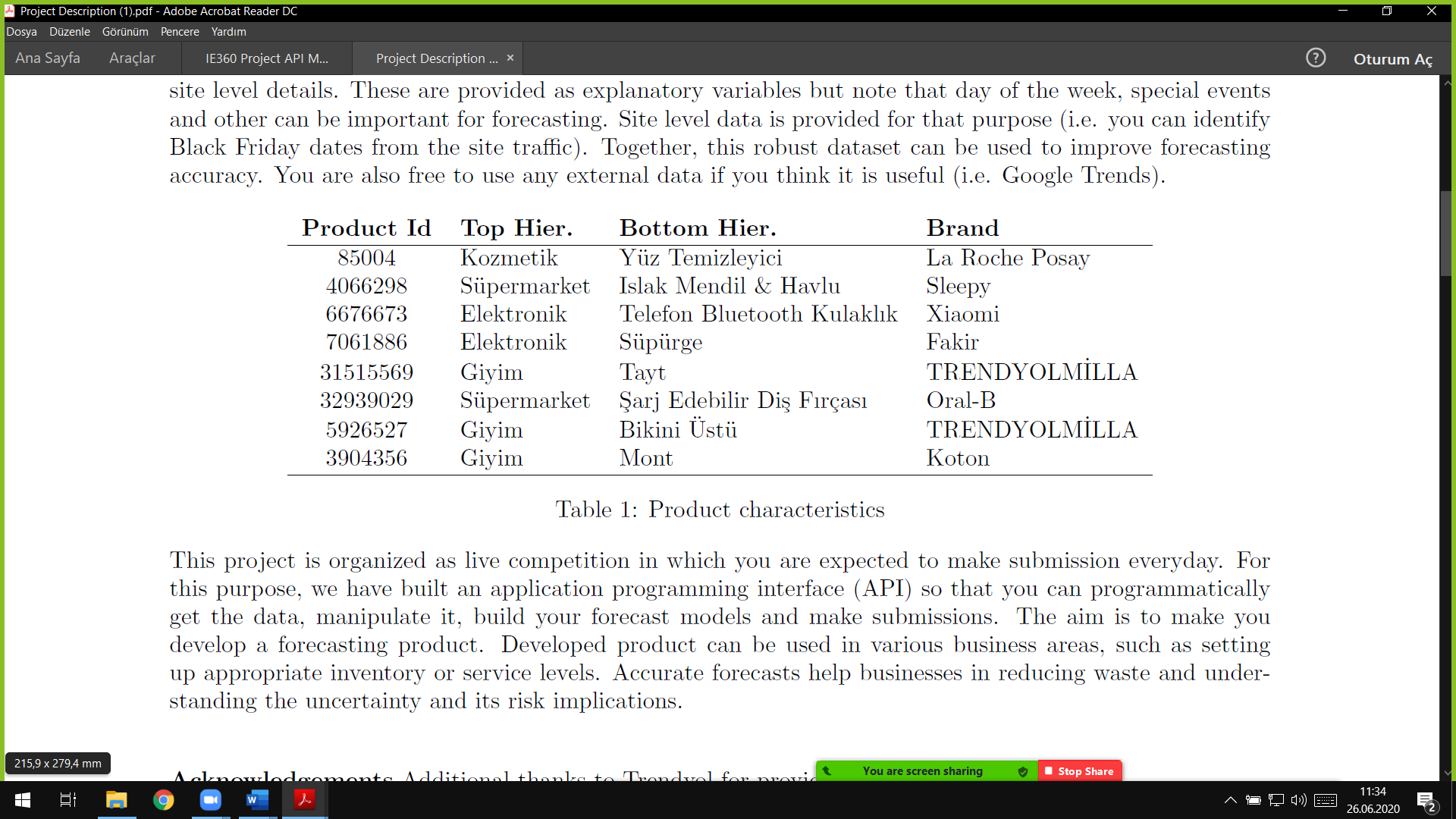
**1.Introduction**

**1.1 Problem Description**

How many products will Trendyol sell each day? Forecasting the sales quantity of a product for an onlineretail is a challenging task. Forecasting heavily relies on the historical data and modeling skills. In this project, it is expected to develop forecasting methods to predict the next day's sales quantity of the products in *Table 1.* Many decisions such as pricing, inventory placement and etc. require reliable forecasts and inaccurate business forecasts could result in actual or opportunity losses. The data covers daily sales of eight products of Trendyol and includes product, category level, brand and site level details. These are provided as explanatory variables. Together, this robust dataset can be used to improve forecasting accuracy. Accurate forecasts help businesses in reducing waste and understanding the uncertainty and its risk implications.

**1.2 Descriptive Analysis of Data**

**1.2.1 Sales Periods**

There are specific campaign periods that prices drastically drop and thus increasing the sold counts. Since the campaign periods are pre-determined by the company, the advertising strategies are effective in social media, TV commercials, billboards along with e-mailing and other communication channels. “Black Friday” as “Efsane Günler” in Trendyol, is the peak point of consumption curve obtained in the time horizon. Since it has a specific date every year and company spend enormous amounts of money to advertise it, consumers wait to buy some products that they want to buy for some time. Along with the steep decrease in prices and effort in marketing, consumer behavior tends to buy more and more. There are some other sale periods that have significant effect on sold counts. In 2019, November 9-11, November 25-29 “Efsane Günler”, and in 2020, February 4-6, February 14 Valentine’s Day, March 10-12 are the major campaign periods. These peak points tend to manipulate the forecasting model, therefore suitable handling methods are applied for each product at each campaign period.

**1.2.2 Stock-out Points**

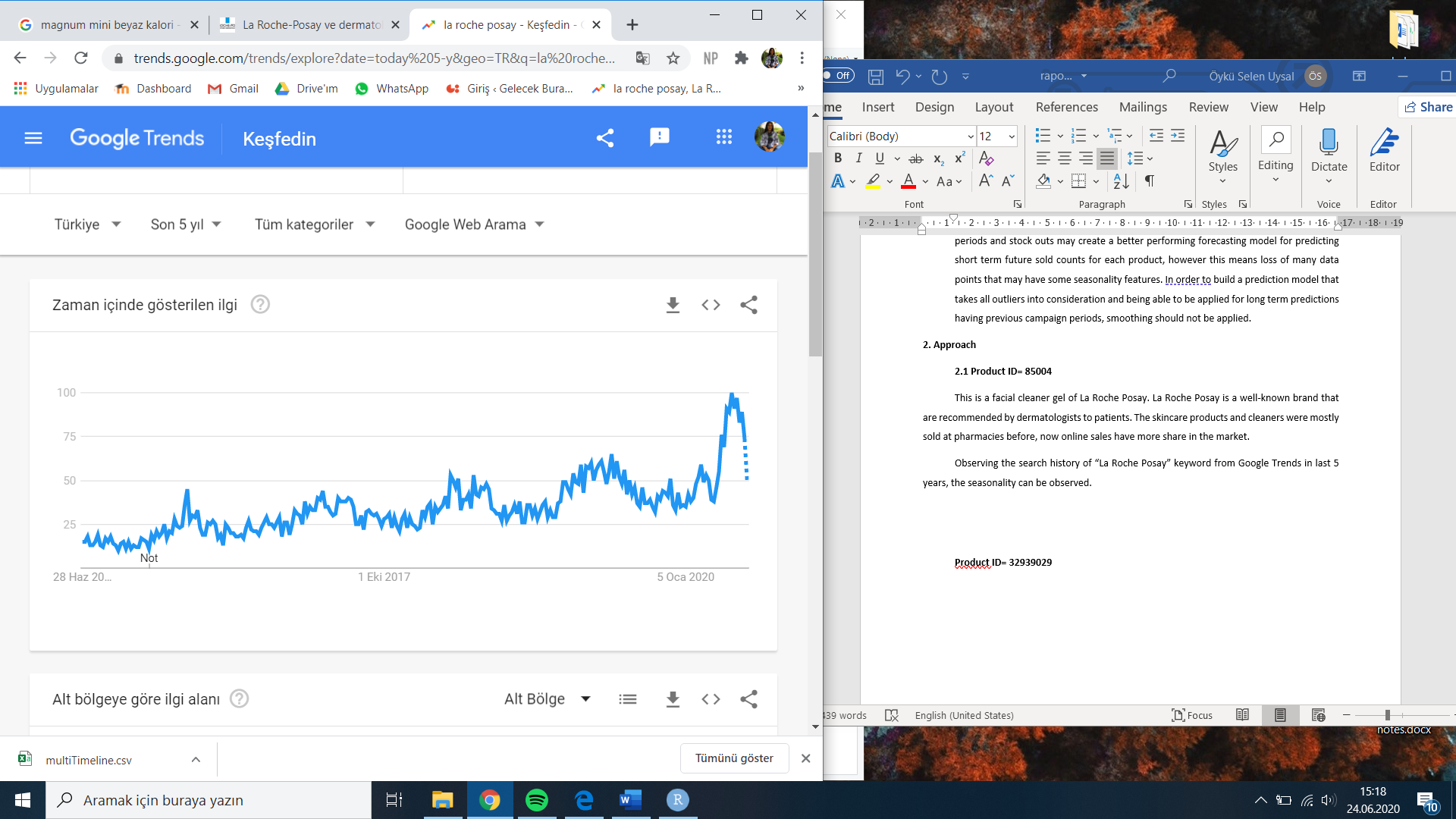
There are days that the product is not eligible to buy at the site, that may be due to inventory problems or some conflict with supplier. Those days are indicated in the dataset with Price value being equal to -1 and sold count is 0. There are pointwise stock-outs as well as periodic stock-outs for different product types. Not all products are sold for the entire time horizon, some of them are being sold starting from a specific date. This situation needs to be handled when building models not to manipulate it.

There is a question rising here, smoothing outlier data points such as campaign periods and stock outs may create a better performing forecasting model for predicting short term future sold counts for each product, however this means loss of many data points that may have some seasonality features. In order to build a prediction model that takes all outliers into consideration and being able to be applied for long term predictions having previous campaign periods, smoothing should not be applied.

**2. Product ID= 85004**

**2.1 Product Based Analysis**

This is a facial cleaner gel of La Roche Posay. La Roche Posay is a well-known brand that are recommended by dermatologists to patients. The skincare products and cleaners were mostly sold at pharmacies before, now online sales have more share in the market.

Observing the search history of “La Roche Posay” keyword from Google Trends in last 5 years, it is expected to have an increase in visit and sold counts of the product in Spring and Summer seasons each year. This year, due to worldwide COVID-19 pandemic consumer preferences and behavior changed all in a sudden. As İstanbul Chamber of Commerce CEO Şekib Avdagiç announced, in April, more than 3 million credit-card is used for the first time for e-commerce activities. For dermatological skincare products in specific, the quarantine period and social isolation may have led people shift their shopping platform from pharmacies to e-commerce websites, in this case Trendyol.

*Figure 1*

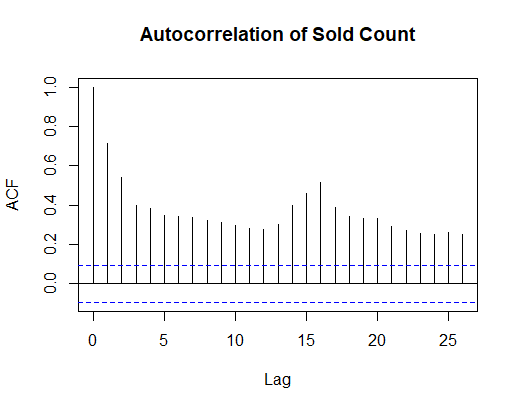
graph of la roche posay facial cleaning gel sold count data in time
There might be a subtle correlation between Google Trends search history and sold count data obtained from Trendyol especially for the last 3 months. This alone does not prove anything however it may help to choose between different forecasting models or combination of those models.

*Figure 2*

**2.2 Approach**

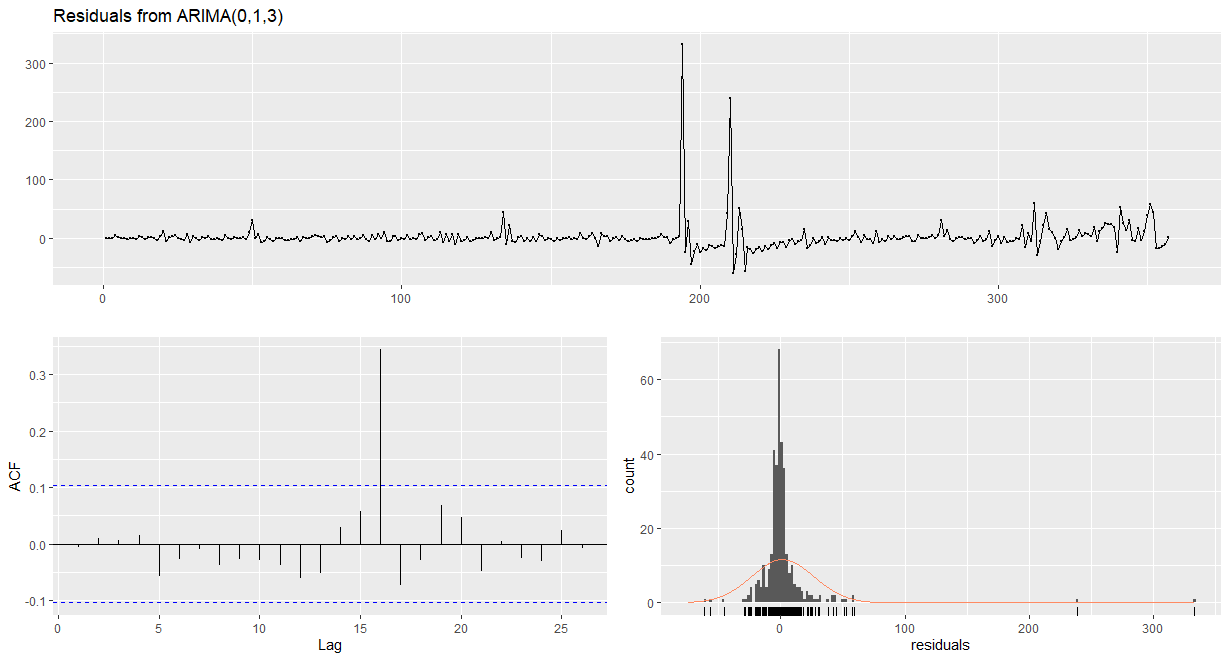
To begin with, sold count versus date plot is obtained to visualize the dataset. As expected, the campaign dates are peak points that can be accepted as outliers in some models. In this part, ARIMA and Linear Regression models are built in order to create forecasting models. For ARIMA model, the “outlier” points are smoothed in different ways to be explained in detail later. For LR models peak points are not smoothed, not to lose the information that they contain that affect the graph of la roche posay facial cleaning gel sold count data in time
importance of Price variable in the sold count values. 5 stock-out points that are “2019-05-08, 2019-05-09, 2019-06-26, 2019-07-13, 2019-09-01” are detected and smoothed by taking simple average or regarding the opposite effect of price, weighted average of neighbor data points. Both price and sold count values are smoothed.

*Figure 3*

Autocorrelation plot demonstrates the strong autocorrelation at lag 1 and 16. This highly stems from the campaign periods’ spacing. By smoothing extreme points but not totally erase the increasing trend, it is aimed to represent the result of 16 days lag autocorrelation thus decrease in price periods.

*Figure 4*

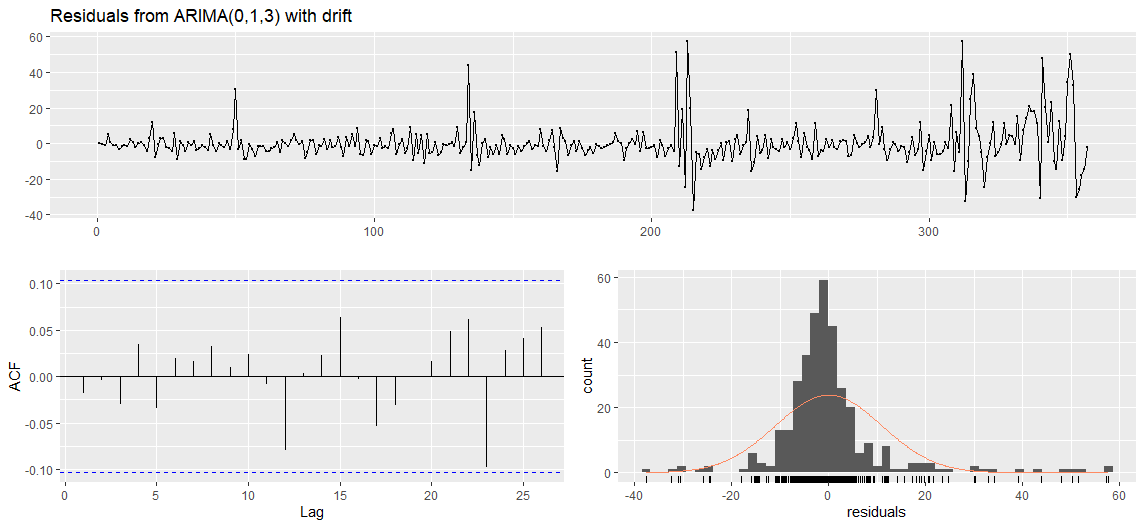
**2.2.1 ARIMA Model**

Smoothing only stock-out points did not perform well when no manipulation is made on the campaign periods. Residuals function gives the plots below:

*Figure 5*

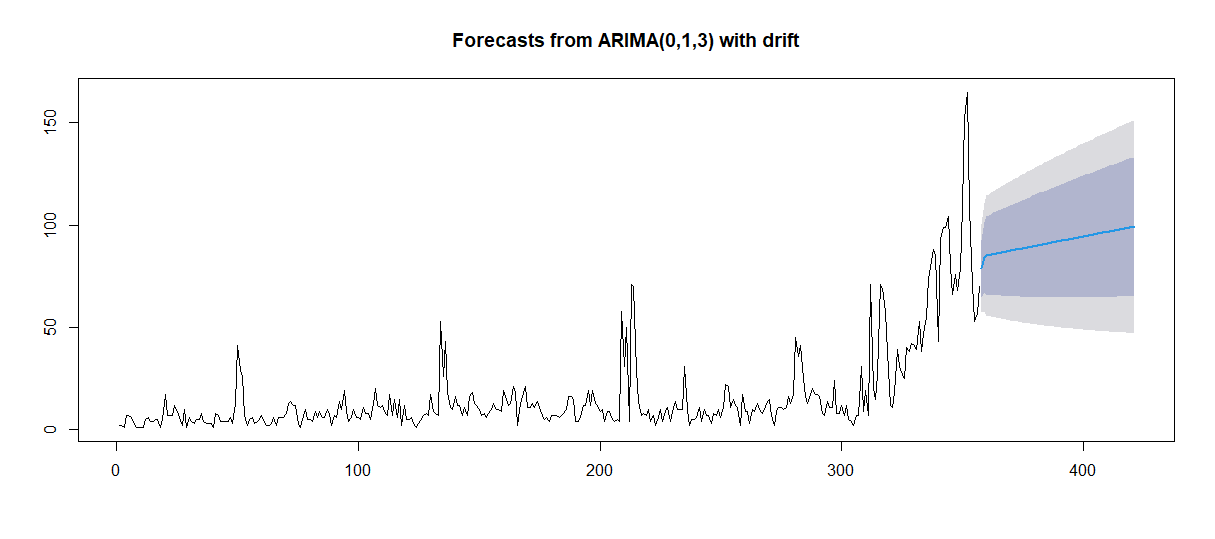
There is a significant autocorrelation at lag 16 for residuals and the distribution of residuals is away from normal distribution. Applying this model would not be reasonable, therefore the sold count values of the campaign periods are smoothed. Prices are not smoothed for the ARIMA model since it takes only sold count and event date information as input.

Each campaign period has different sold\_count changes pre-campaign and post-campaign periods therefore some of them smoothed by taking average of the 2 days before and after sold count numbers, where some of them are assigned the previous day’s sold count value. Aim was to softening the extremity while keeping the rise effect in the series.

Before applying models, the dataset is split into train and test sets. Random sampling would not be applicable for forecasting time series, therefore it is decided to split the dataset by picking a time point. In this case, last 64 observations are taken as test set observations. Auto.arima() function is used to built model and the below plots are obtained with Residuals() function on the model:

*Figure 6*

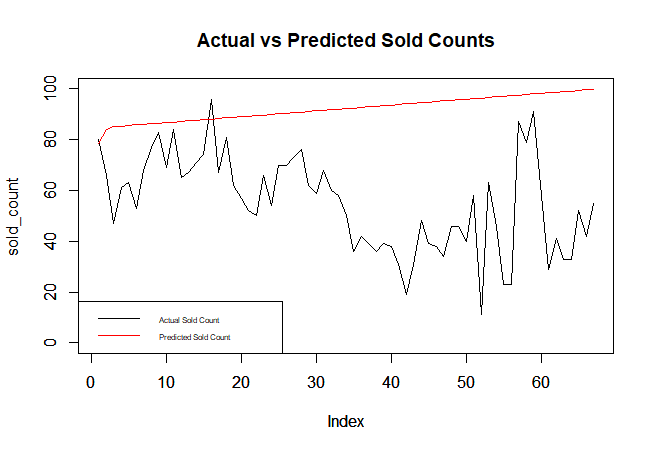
The autocorrelation levels of error terms are in between the boundaries that are acceptable, distribution is close to normal and therefore it can be taken as White Noise.

The forecast obtained from this model predicts as below:

*Figure 7*

The predictions are performed poorly, considering the performance measures obtained below:

RMSE 42.83006 MAE 37.41569 MAPE 100.07



*Figure 8*

The split point has a critical importance here, as of May the sales began to decline and the increasing trend started to diminish. However, the model could not foresee this. Here, the weakness of time series model appears and the need for a model like linear regression is needed.

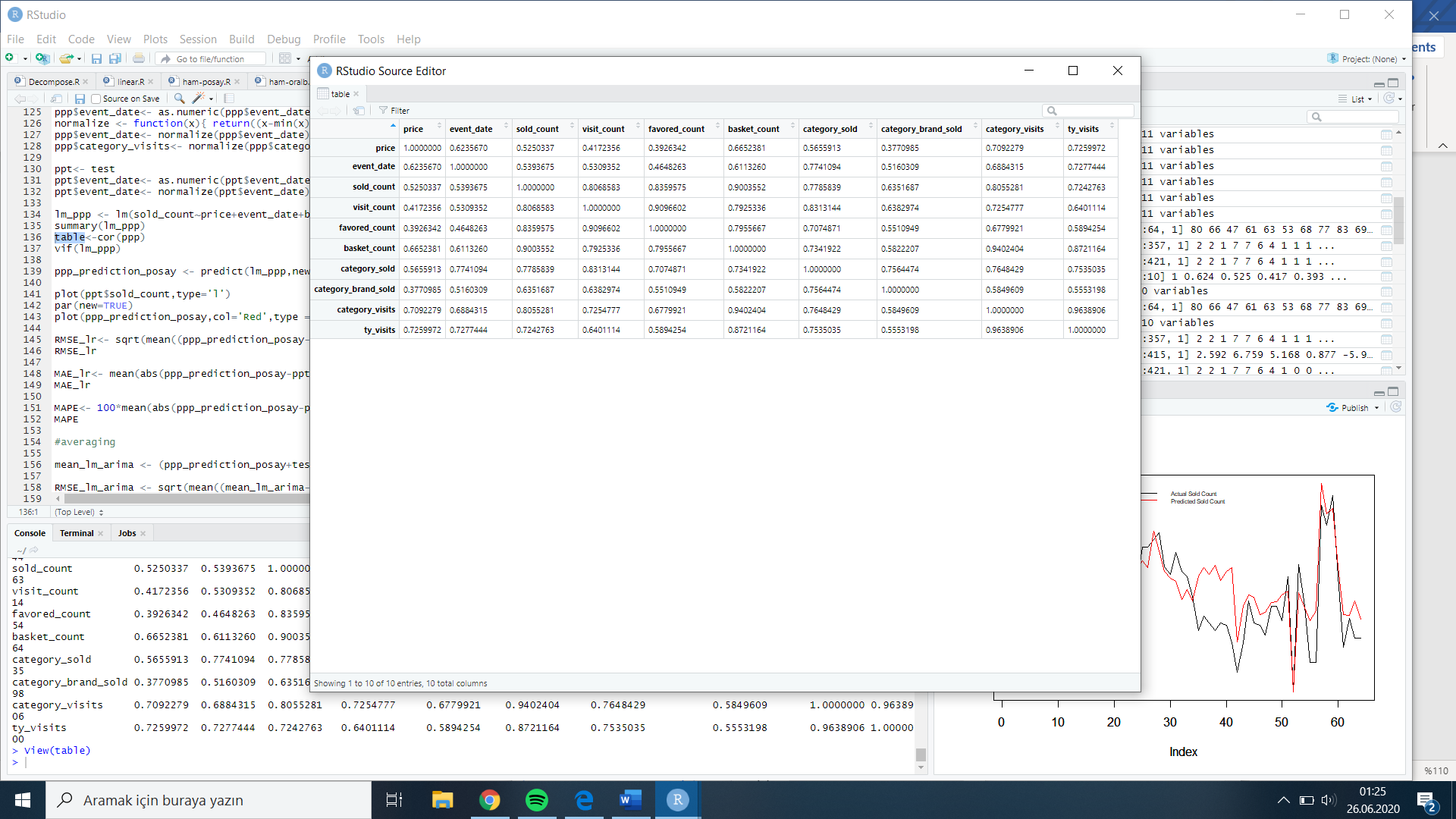
**2.2.2 Linear Regression Model**

In order to take other significant variables’ influence and correlation on sold count values into account, linear regression model is used. As it was in previous model, the dataset is split into train and test sets, latter having last 67 observations.

Here first model is built only excluding product\_content\_id column, however this model performed poorly due to some issues like category\_sold variable’s high significance in model. That is, when there is a sudden increase in this variable the model starts to increase the predicted sold count values as 4-5 times larger than previous ones which deteriorate the prediction performance.

The performance measures below state the poor performance.

RMSE 400.7634 MAE 335.6999 MAPE 702.4724

In order to overcome this problem, it is decided to eliminate some columns in the dataset given to model as input. cor() function is useful here to check correlation of input variables. TY\_visits, category\_visits and basket\_count columns are highly correlated. To support the claim, vif() function is used, “the VIF of a predictor is a measure for how easily it is predicted from a linear regression using the other predictors.” (http://Rdocumentation.org). Also, until November 25,2019 date all basket\_count data is 0 which manipulates the model.



*Table 2*

*Table 3*

In the new model, Price, Visit Count and Favored Count columns are given as input variables.

The summary of the model is:

Residuals:

Min 1Q Median 3Q Max

-76.137 -5.288 -0.043 4.289 98.321

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 175.937491 21.144459 8.321 1.93e-15 \*\*\*

price -2.594057 0.301848 -8.594 2.76e-16 \*\*\*

visit\_count 0.036658 0.001709 21.453 < 2e-16 \*\*\*

favored\_count -0.036251 0.014919 -2.430 0.0156 \*

---

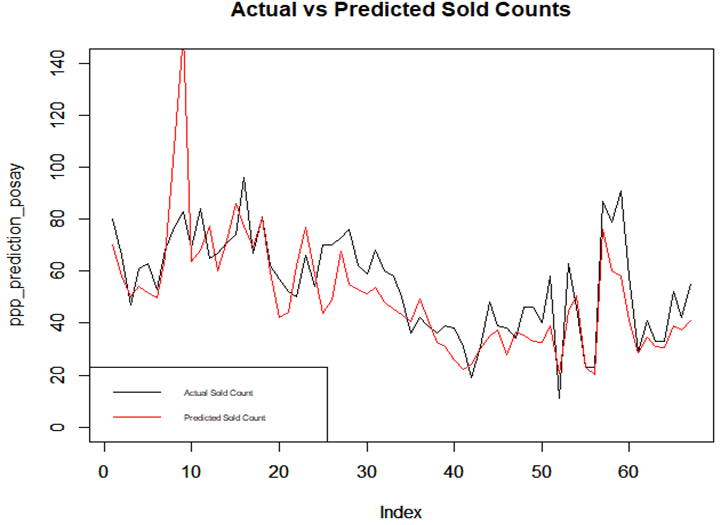
Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 13.88 on 353 degrees of freedom

Multiple R-squared: 0.8456, Adjusted R-squared: 0.8442

F-statistic: 644.2 on 3 and 353 DF, p-value: < 2.2e-16

Performance measures as stated below:

**** > RMSE 34.79341 MAE 16.93221 MAPE 36.21825

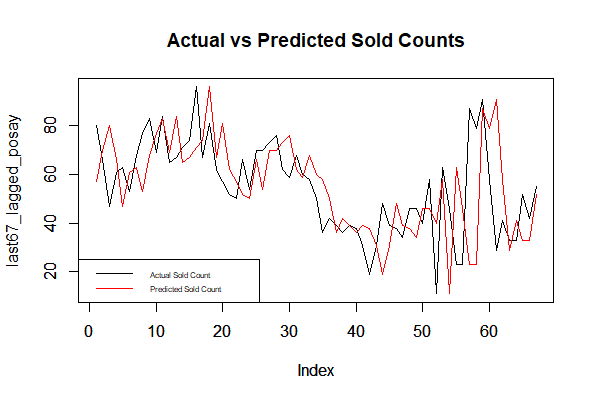
*Figure 9*

**2.2.3 Naive Model**

Naïve forecast is applied with lag 2. To compare better, the test set is chosen as the same size with

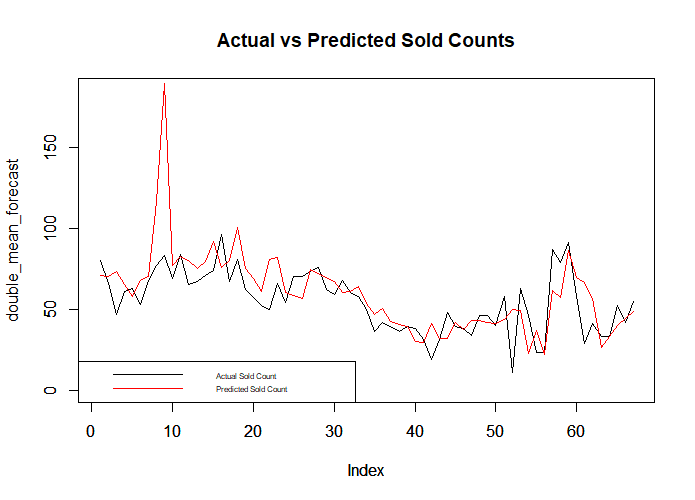
other models’ test set sizes which is 67. The naïve forecast performance measures are stated below:

> RMSE 19.35451 MAE 13.85075 MAPE 32.42971

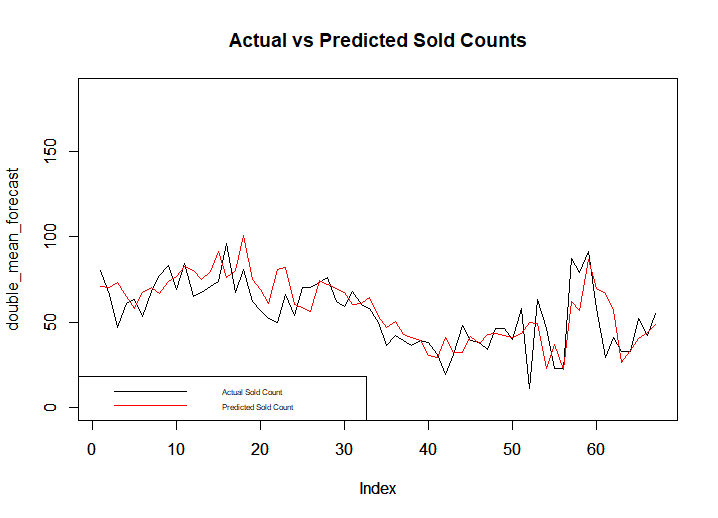


*Figure 10*

**2.2.4 Combined Model**

Here, averaging models is an option to compensate weaknesses of different model approaches. By taking simple average of LR model predictions and Naïve predictions, all performance measures are improved: > RMSE 19.22686 MAE 12.21977 MAPE 27.30915

*Figure 11*

There is an important point, the visit count and favored count data are valued in the linear model and in general the correlation between sold count and these variables. However, on the plot above it is seen that some problematic prediction occurred on the 8-9 indexes which correspond to April 28-29,2020. Examining these days, it is seen that there is a boom in visit count, basket count and favored count numbers. The boom pushes the model to predict accordingly on those days and linear model makes sold count predictions around 180 and 300 where actual sold count is around 60-80 those days. The problem may stem from an advertising issue which led an increase in visit counts of the product that days, however this situation distorts the predictions. When manually smoothing the boom, the RMSE value decreases around 5.5, MAE decreases 2 and MAPE decreases 2.3 points and the plot becomes:

*Figure 12*

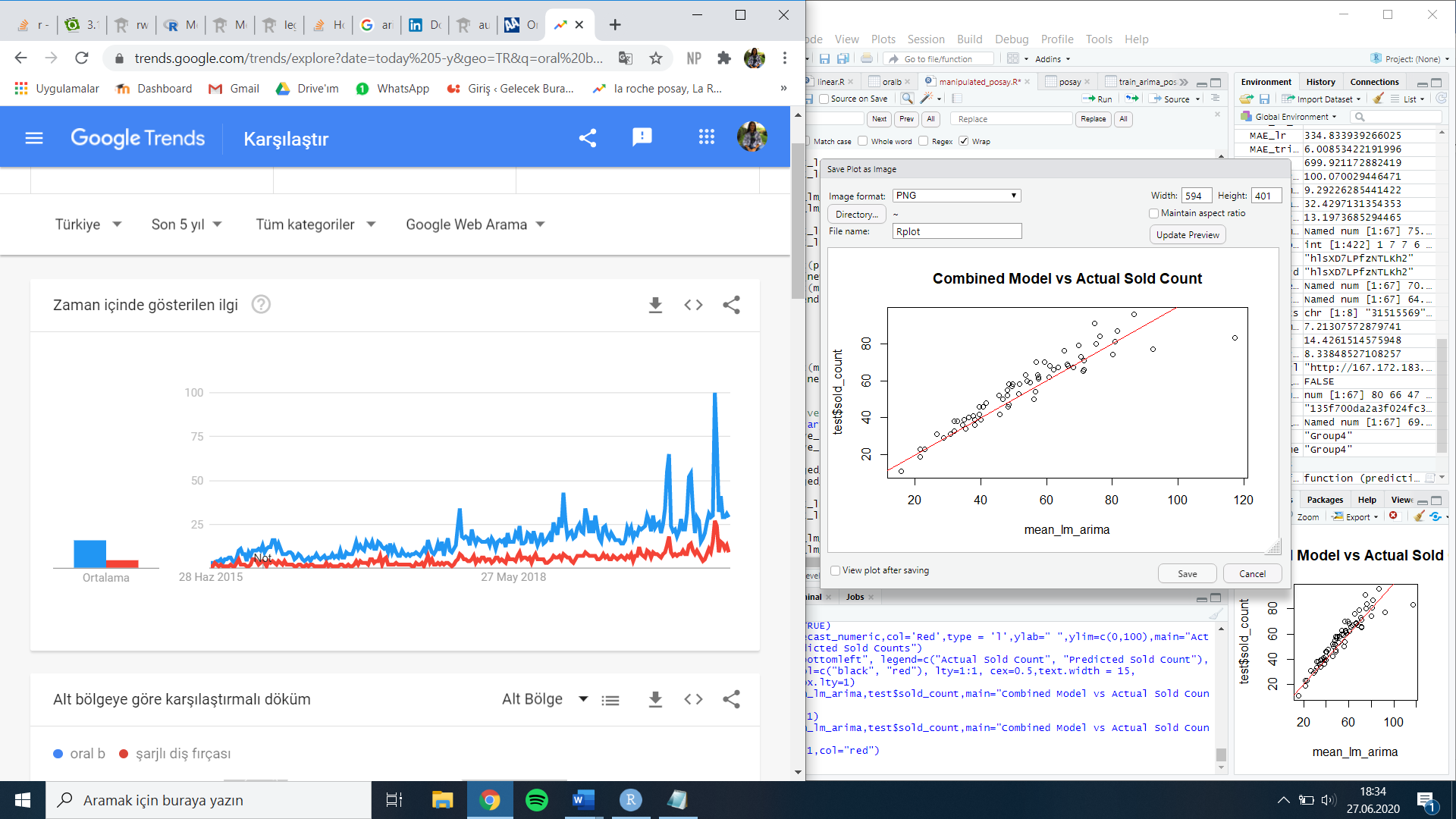
However, such manual intervention is not sustainable in the future of model. Therefore general ratio analysis should be done between different variables and also there must be some dummy variables that indicate whether advertising or marketing campaigns are influential on specific days.

**2.3 Result**

The combination of Naïve and Linear Regression models gave the best performance measures. Nevertheless, the model is too sensitive to change in visit count and favored count variables and this causes serious mispredictions. In order to use such model safely, the dataset should be analyzed well and all data should be trustable, without numerical errors etc.

**3. Product ID= 31515569**

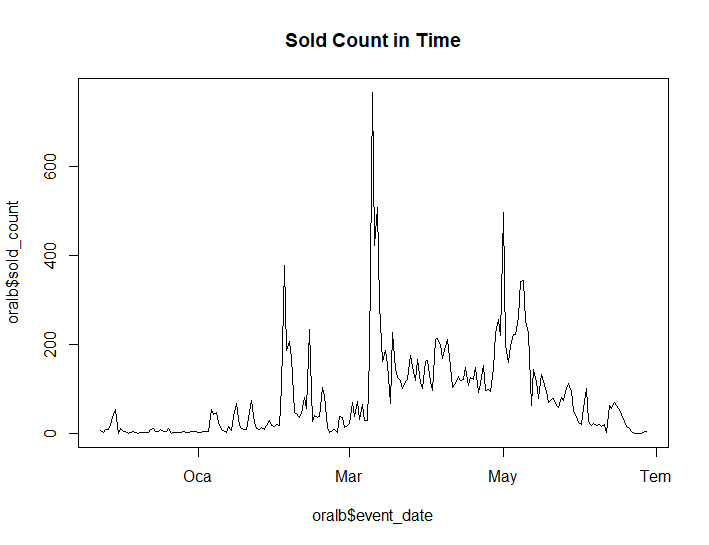
**3.1 Product Based Analysis**

This is a rechargeable toothbrush model of Oral-B. Product can be categorized as luxurious due to high prices and basic toothbrush alternatives with lower prices. In Google Trends search data, blue line represents the “Oral-B” keyword and red line represents the “şarjlı diş fırçası” for the last 5 years.

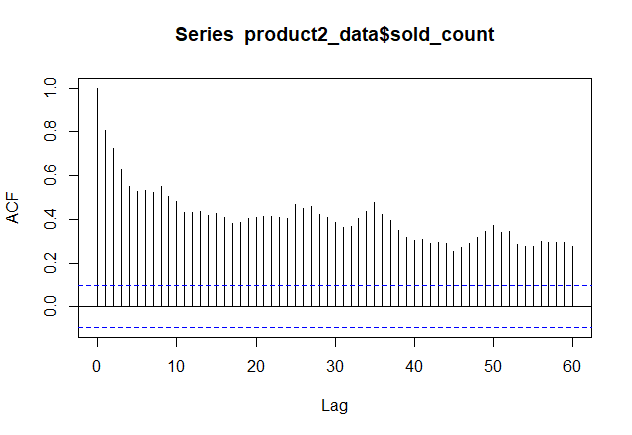
*Figure 13*

It seems like Black Friday and February campaigns has an influence on search, the peak points belong to November and February months. For the first time, May has more search counts than other months in Google Trends search history.

**3.2 Approach**

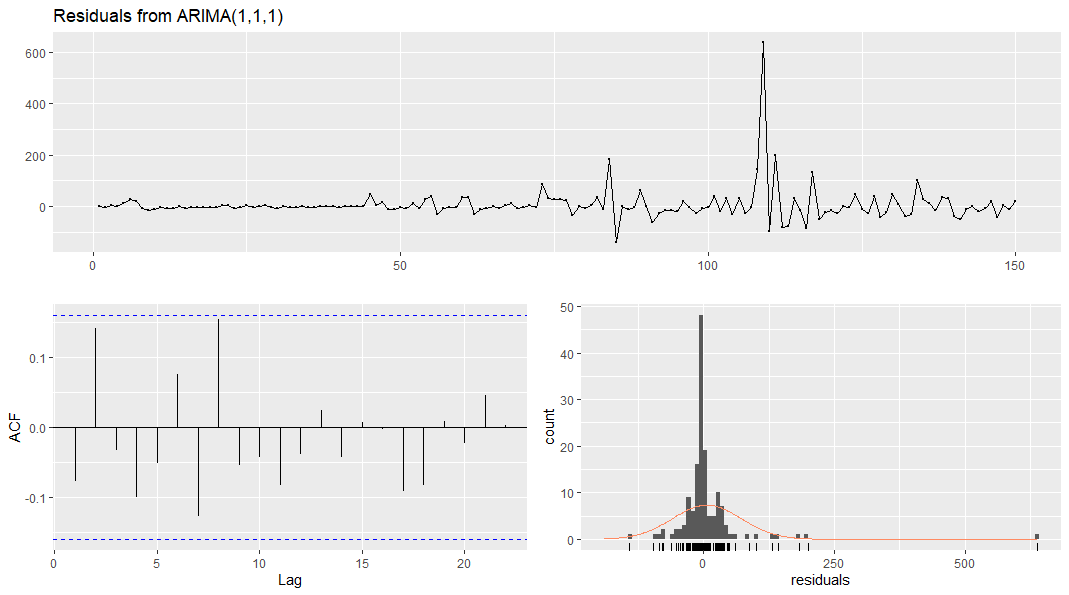
To begin with, sold count versus date plot is obtained to visualize the dataset. As expected, the February 4-6 and November 9-11 are peak points that are smoothed as outliers in some models. In this part, Naïve, ARIMA and Linear Regression models are built in order to create forecasting models. For ARIMA model, the “outlier” points are smoothed as equating the previous day’s sold count data. For LR model peak points are not smoothed, not to lose the information in effect of different variables. Only stock out points that are, in 2019 November 30, December 4,8 and 21, and in 2020, June 22-25 are intervened. Also, there is no sale until November 22, therefore the data until this date is deleted. The next plot shows the sold count in time:

*Figure 14*

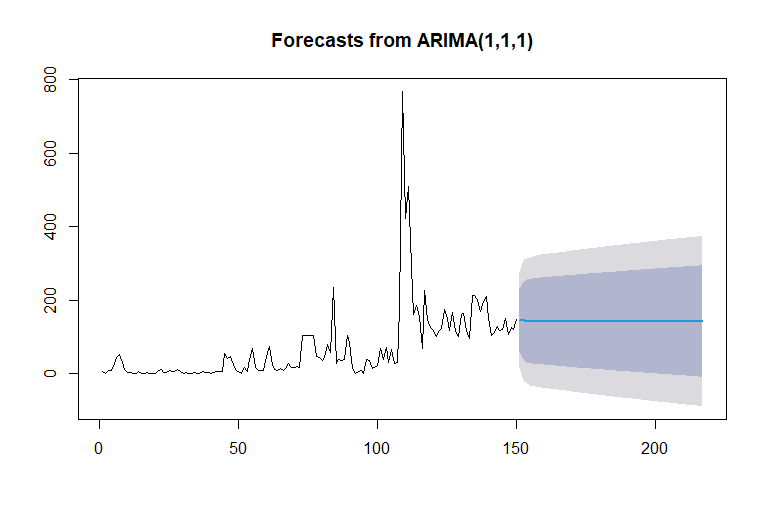
Autocorrelation function plot is below, it is seen that there is high autocorrelation values at different lag values.

*Figure 15*

**3.2.1 ARIMA Model**

Auto.arima() function is used to build model and the below plots are obtained with Residuals() function on the model:

*Figure 16*

The autocorrelation levels of error terms are in between the boundaries that are acceptable and therefore it can be taken as White Noise. The forecast obtained from this model predicts as below:

*Figure 17*

The predictions are performed poorly, considering the performance measures obtained below:

RMSE 134.3786 MAE 95.89926 MAPE 94.53622

**3.2.2 Linear Regression Model**

In order to take other significant variables’ influence and correlation on sold count values into account, linear regression model is used. As it was in previous model, the dataset is split into train and test sets, latter having last 67 observations.Just like in the product with id 85004, same reasoning is valid here when eliminating category sold, category brand sold, category visit and ty visit columns. Giving price, visit count basket count and favored count columns as input into the model, the below summary is obtained along with the performance measures:

Call:

lm(formula = sold\_count ~ price + visit\_count + basket\_count +

favored\_count, data = train\_oralb)

Residuals:

Min 1Q Median 3Q Max

-97.958 -13.588 2.484 9.898 106.032

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 265.025575 30.261927 8.758 4.75e-15 \*\*\*

price -2.127897 0.256441 -8.298 6.70e-14 \*\*\*

visit\_count 0.018465 0.005357 3.447 0.000742 \*\*\*

basket\_count 0.148239 0.016748 8.851 2.77e-15 \*\*\*

favored\_count -0.269255 0.053775 -5.007 1.58e-06 \*\*\*

---

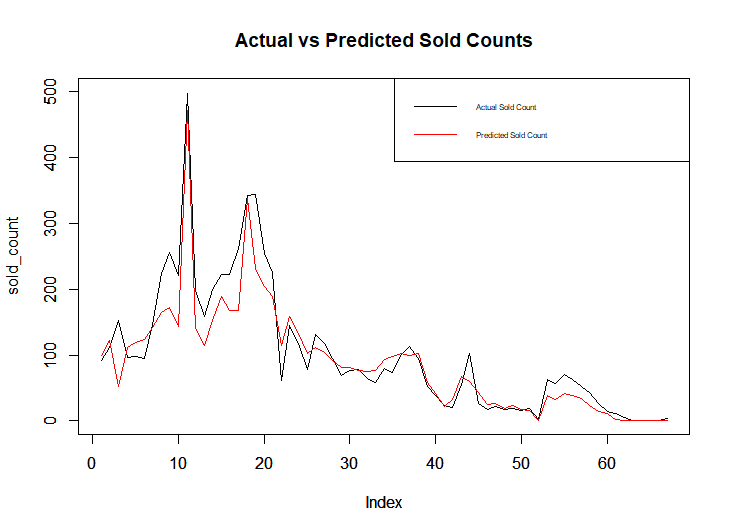
Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 24.68 on 145 degrees of freedom

Multiple R-squared: 0.9444, Adjusted R-squared: 0.9428

F-statistic: 615.1 on 4 and 145 DF, p-value: < 2.2e-16

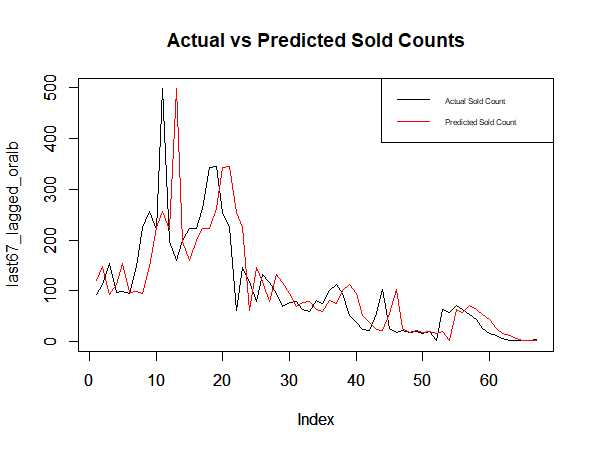
RMSE 34.11595 MAE 22.53744 MAPE 32.4096



*Figure 18*

**3.2.3 Naive Model**

Naïve forecast is applied with lag 2. To compare better, the test set is chosen as the same size with other models’ test set sizes which is 67. The naïve forecast performance measures are stated below: RMSE 72.76829 MAE 44.47761 MAPE 88.99037



*Figure 19*

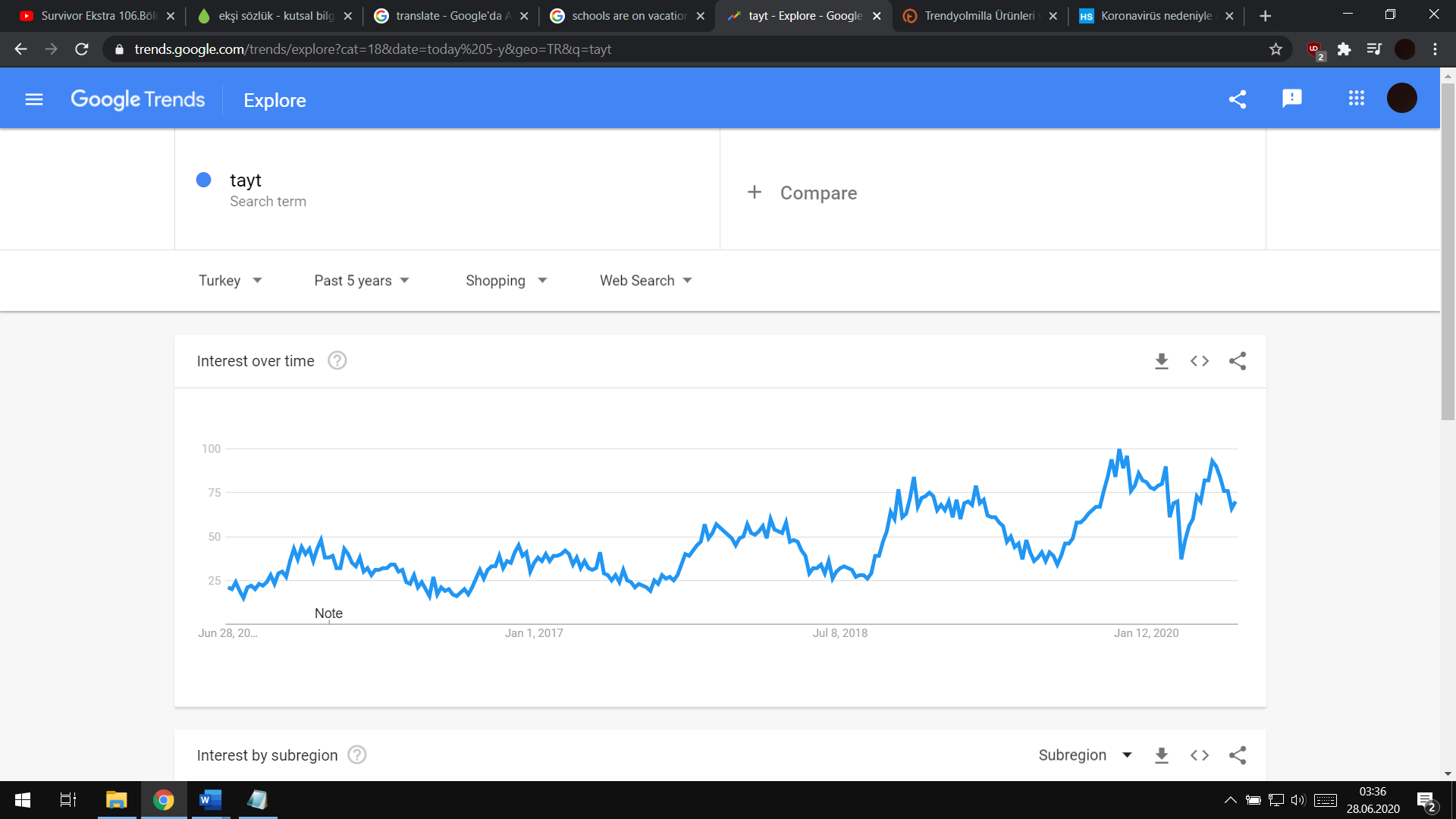
**3.3 Result**

Among the Naïve, ARIMA and Linear Regression models, LR gave the best performance measures. The capability of catching rapid changes in input variables is a significant ability of linear regression. ARIMA could not perform well, this may have stem from the lack of datapoints.

**4. Product ID =** **31515569**

**4.1 Product Based Analysis**

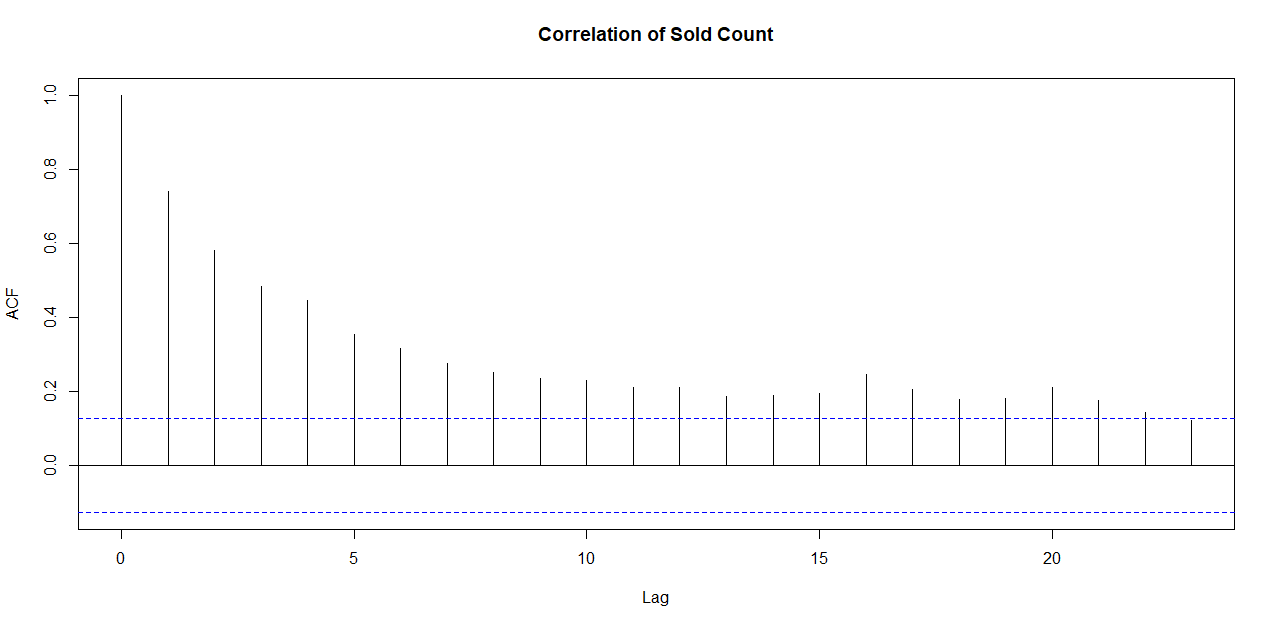
This product is tights of Trendyolmilla which is a brand created by Trendyol and it is sold only on Trendyol. Because the product is produced by Trendyol which is also the seller, the website probably shows this product more than other competitors to increase their profit.

Actually seasonality is not expected for this product but when the Google Trends data of “tayt” keyword in last 5 years is analyzed, there is a seasonality which is shown in figure 20. On the other hand, there is an unexpected decrease in the week which is the week first COVID-19 patient is detected and government gave a break for schools as a first prevention against COVID-19 in Turkey. However, on Trendyol data there is no clear seasonality or decrease in related week.

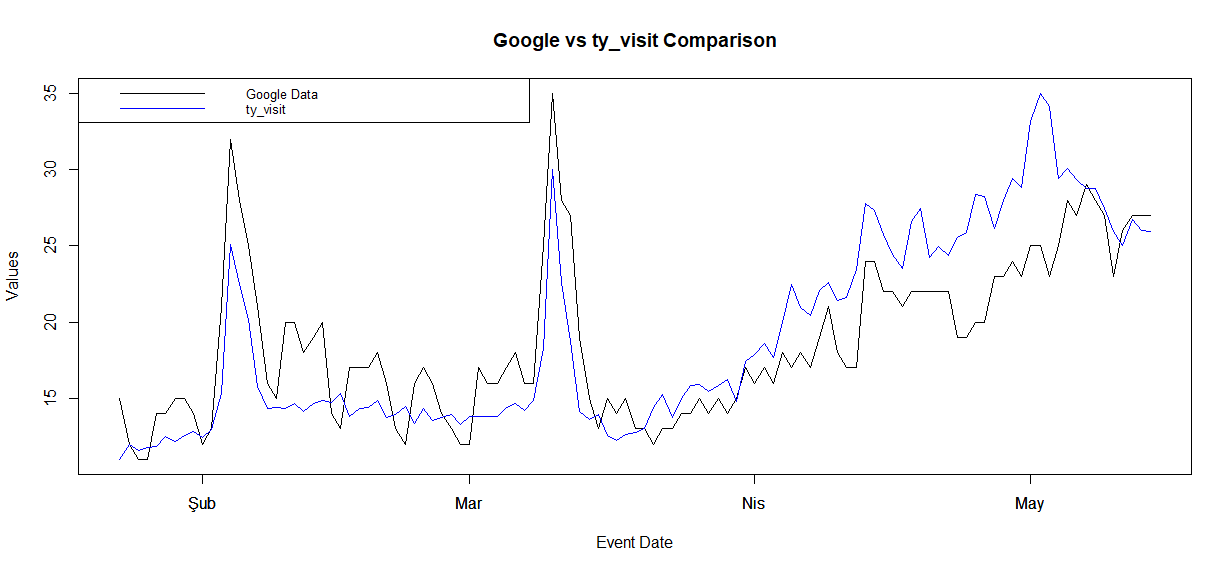
*Figure 20*

**4.2 Approach**

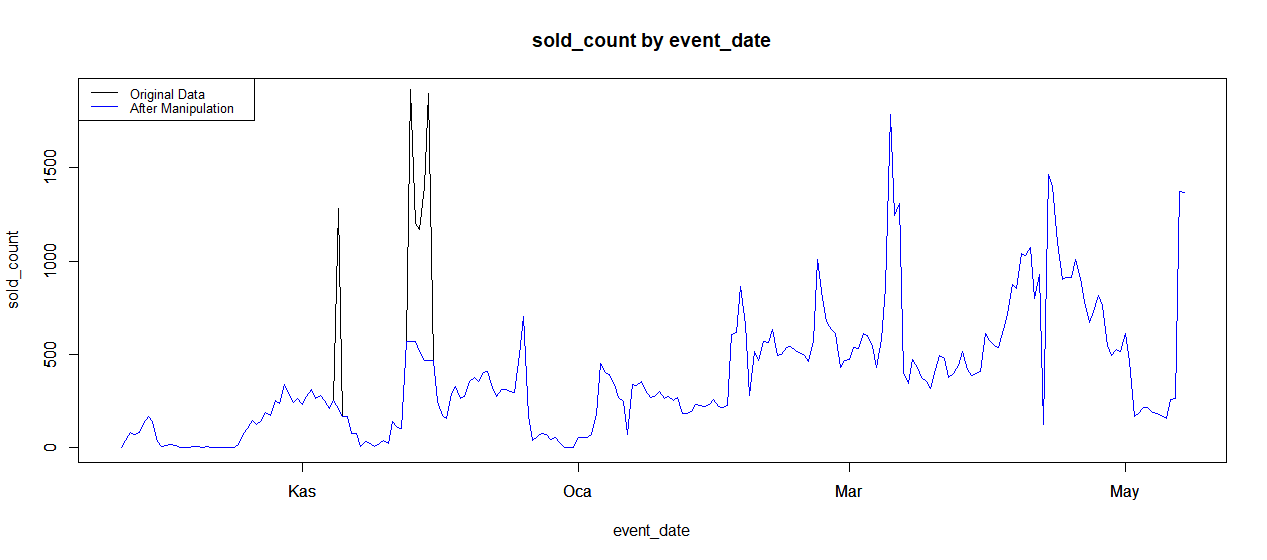
There is no data before September 21 of 2019 for the related data set (Probably sales of related product had not started yet). Therefore, data before 21st September of 2019 can be omitted when generating models. There are “-1” values for price column and sold\_count values are “0” for these days, however there are high sold\_count values for previous or next days which means that “product is out of stock”. There are 8 days like that, and these days can decrease quality of our models so they should be manipulated. Because Lag=1 has the highest correlation value as seen on *Figure 21*, out of stock days are recalculated as the mean of the next day and the previous day.



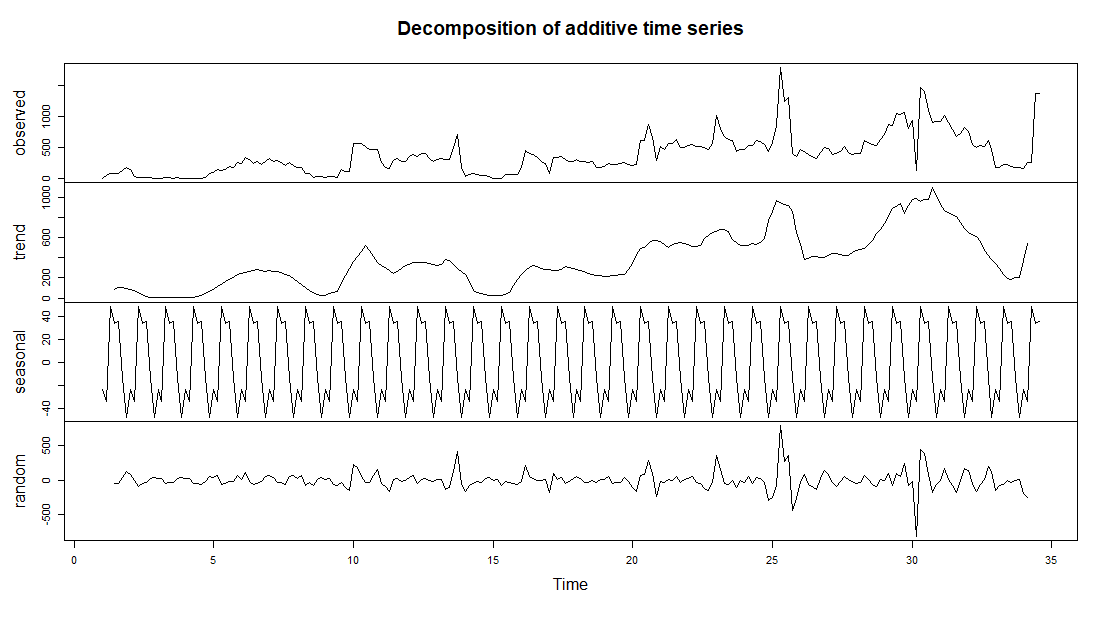
*Figure 21*

ty\_visit values might be important but there is no information before 23rd January of 2020 which is almost half of data. To handle this problem, Google Trends data for “Trendyol” search might be beneficial. As seen in *Figure 22,* they are highly correlated. Therefore, missing parts of ty\_visit can be predicted by using Google Trends data for “Trendyol”. After calculating mean ratio of GoogleData/ty\_visit, with using this value and GoogleData, older ty\_visit values are predicted and assigned accordingly.

*Figure 22*

Online marketplaces like “Trendyol” give discount for specific days to encourage consumers to purchase more. Although there is no discount for some products, consumers are more prone to purchase for this period of time and data of these days are outliers. With the same method used for “out of stock” days is used these days. Trendyol gave discount for following dates: “9-11 November (Big Discount Days)”, “25-29 November (Legendary Days)”. Assigning nearest days' values minimize errors. The outliers and how they are manipulated can be seen in *Figure 23.*

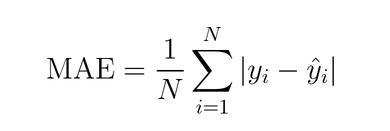
*Figure 23*

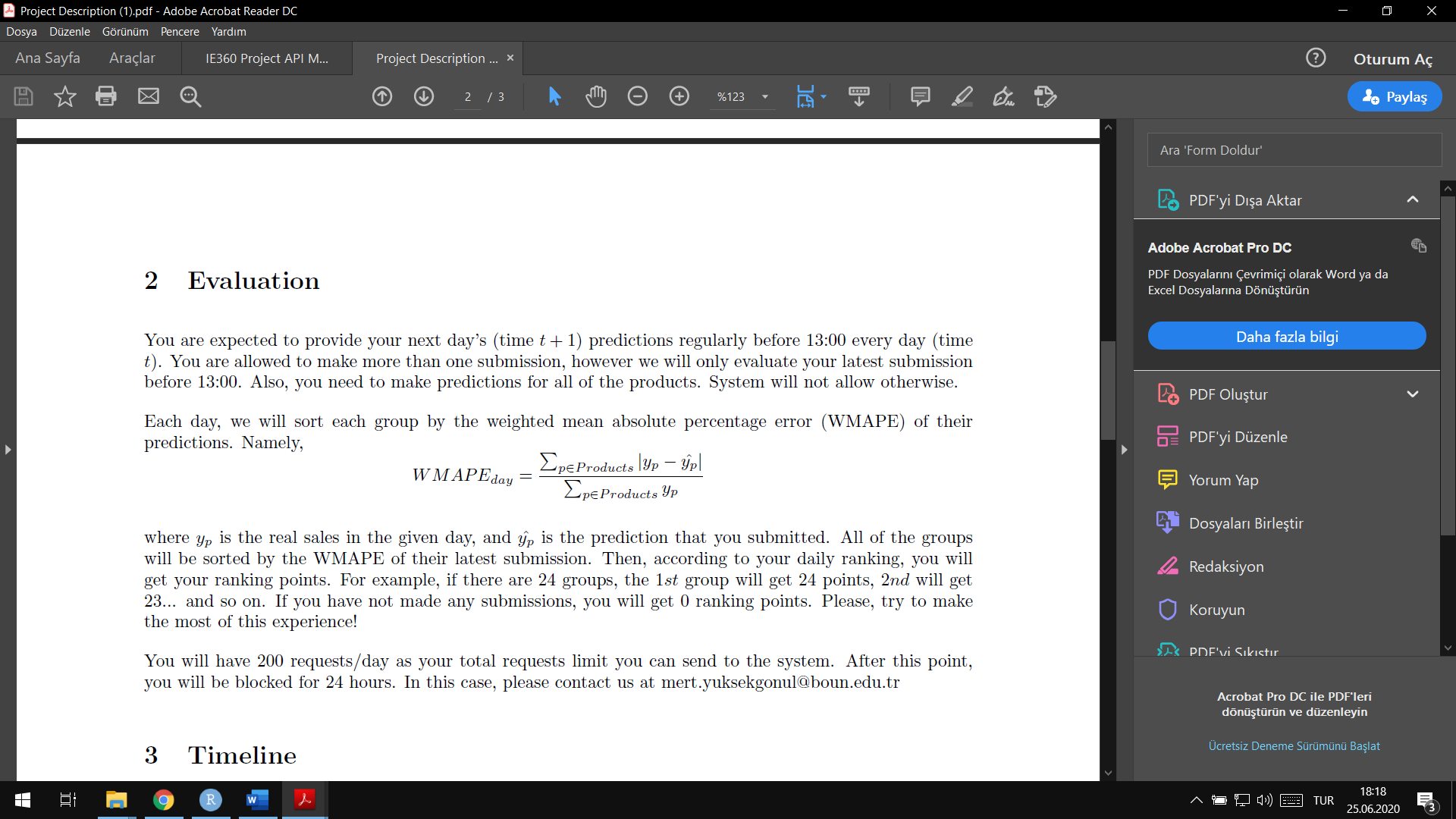
Lastly, in order to see if there is a trend or seasonality for the product 31515569 decomposing is applied. Event date and sold count columns are used while generating time series and named as “ts\_tayt”. Decompose() function is used for ts\_tayt and following results are obtained as in *Figure 24.* As you can see in *Figure 24* there isn’t any significant trend.

*Figure 24*

**4.2.1 Linear Regression Model**

Data is updated at the end of day and tomorrow is predicted by using yesterday data, so all columns but price are converted to lag 2 type. Price column is converted to lag 1 because price value can be added manually, everybody can reach price information online. Also lag 2-7 sold\_count columns are generated to use if there is any kind of effect. There are some unreliable parameters such as category\_sold which does not exist for half of data so they are omitted or category\_sold which is between 0-100 for a long time but increases to range of 1000-3000. To increase accuracy “sample.split ( ) ” function is used which split data to train and test better than random splitting. With these parameters linear regression model is generated by “lm ( )” function using train data set and, forecasted values for test data set are generated. To evaluate quality of model MAPE and MAE values are calculated. As a result MAPE value equal to 127.2735 and MAE value equal to 168.115 are obtained.





The Summary of the linear regression model is shown below.

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 7.264e+02 5.716e+02 1.271 0.20574

lag2\_sold\_count 6.174e-01 1.310e-01 4.713 5.54e-06 \*\*\*

lag3\_sold\_count -1.294e-01 1.308e-01 -0.990 0.32401

lag4\_sold\_count 1.630e-01 1.016e-01 1.605 0.11067

lag7\_sold\_count 2.540e-01 7.779e-02 3.265 0.00135 \*\*

lag2\_ty\_visits 2.515e-07 4.284e-07 0.587 0.55811

lag1\_price -1.658e+01 1.447e+01 -1.146 0.25365

lag2\_visit\_count -1.748e-03 3.313e-03 -0.528 0.59859

lag2\_favored\_count -2.975e-02 5.844e-02 -0.509 0.61151

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

**4.2.2 Naive Model**

Naïve Approach is the most basic approach to apply so it should be tried. As usual Naïve approach is also used to generate a model. Most often, naïve models used are random walk (current value as a forecast of the next period) but current data is yesterday’s data so the naïve model generated will use the yesterday’s data to predict tomorrow. Then, Lag 2 sold\_count value is used as forecast directly. MAPE and MAE values of Naïve approach is also calculated. As a result MAPE value equal to 99.67039 and MAE value equal to 128.2778 are obtained.

**4.3 Results**

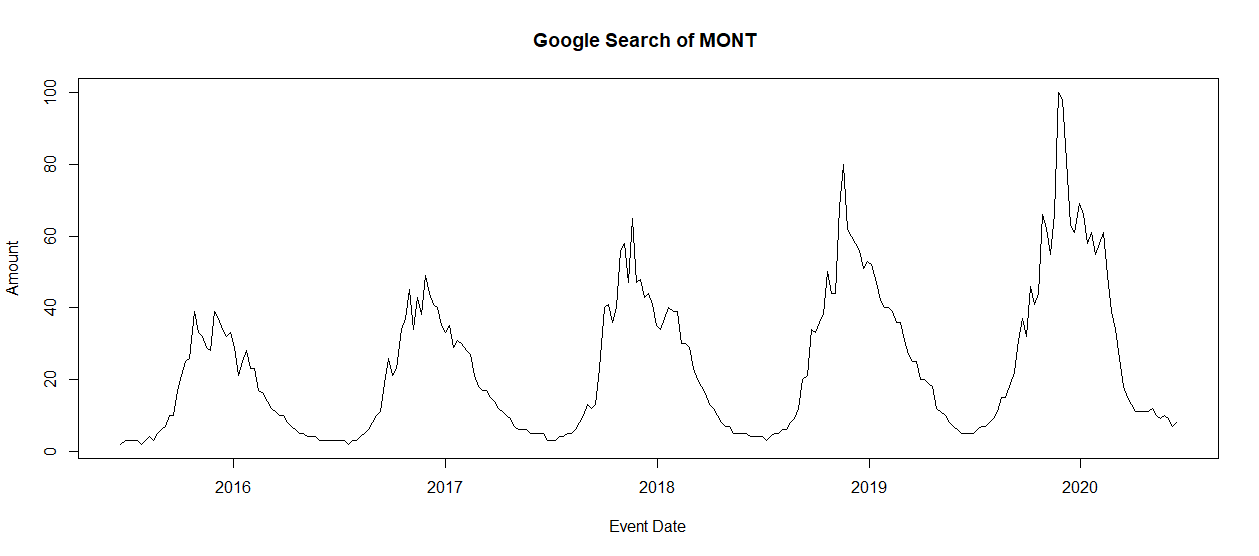
Although according to Google Trends data “tayt” is a seasonal product, there is no seasonality detected for product 31515569. One of the two generated models (Linear Regression Model and Naïve Model) should be chosen. Because MAE and MAPE values (quality measures) are better for Naïve Approach, Naïve Model is the best for now.*(Table 1)*



*Table 4*

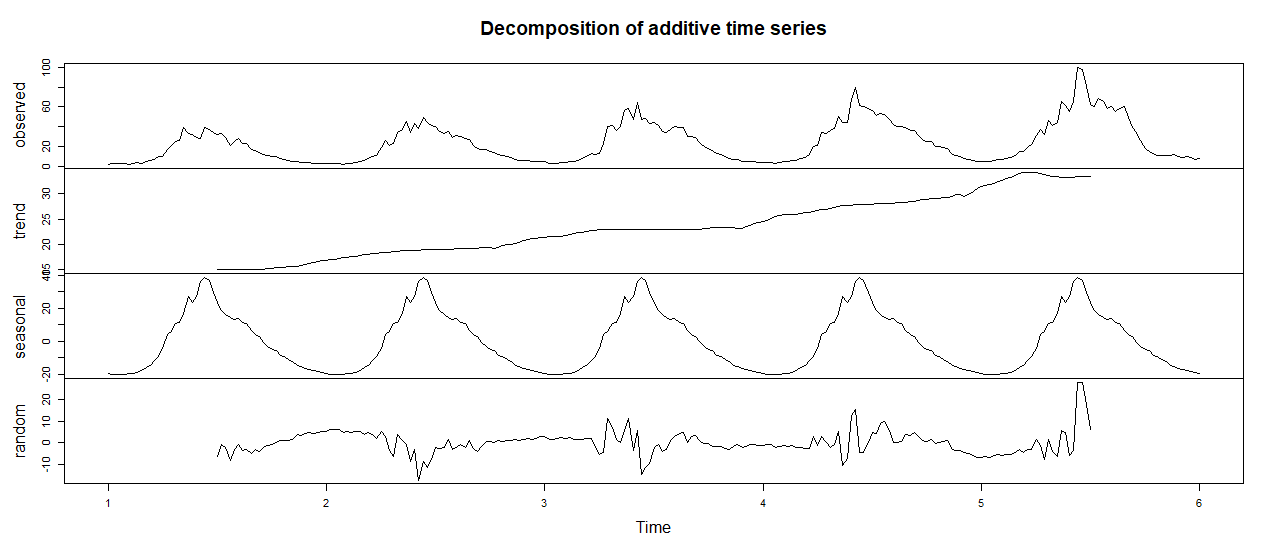
**5. Product ID = 3904356**

**5.1 Product Based Analysis**

This product is a coat of Trendyolmilla as the previous product so same conditions are valid for this data set also. “Coats” are highly seasonal products. They are only used in winter. However, consumers generally bought them just before winter. There is a peak point before winter. In order to make interpretation of data and seasonality, Google Trends is used again. Google search data of “Mont” in Turkey from 2015 to 2020 for shopping category is obtained. Trends are shown at *Figure* *25*.

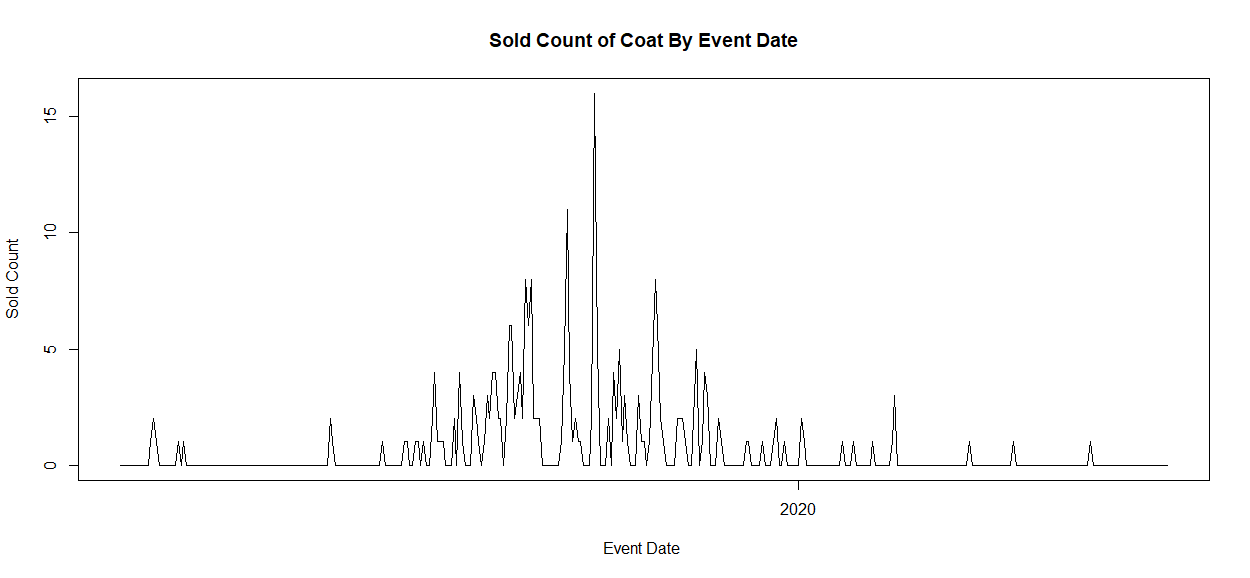
*Figure 25*

To understand better seasonality of this data set decomposing is used again. Firstly data set converted to time series and “decompose ( )” function is applied. As a result trend and seasonality are obtained clearly. Trend may stem from the increasing use of online marketplaces and internet. However, seasonality directly affects quantity sold. Decomposed data is shown in *Figure 26* .



*Figure 26*

**5.2 Approach**

To improve quality of model, manipulation of discount days as before is preferred. Same method used for product 31515569 is used for coat data. Coat data must be a seasonal data because of product’s nature. Sold\_count by event\_date plot is shown at *Figure 27.*

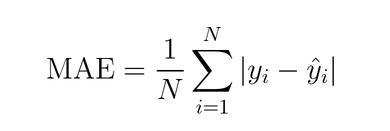
*Figure 27*

This data should be added to main data to generate better model. Seasonal data can be added alone, however trend is omitted in this way so google data is converted to lag 2 version and combined with original data.

Specifically, for this product sold counts are quite low even in winters. There are “-1” values for price column and sold\_count values are “0” for these days. Although Trendyol is able to sell the product, none were bought thus making data unclear. To minimize negative influence of this problem, -1 price values are assigned as the nearest price value.

**5.2.1 Rounded Linear Regression Model**

Data is updated at the end of day and tomorrow is predicted by using yesterday data, so all columns but price are converted to lag 2 type. Price column is converted to lag 1 because price value can be added manually, everybody can reach price information online. Also lag 2-7 sold\_count columns are generated to use if there is any kind of effect. There are some unreliable parameters such as category\_visit and ty\_visit which is not exist for half of data so they are omitted To increase accuracy as before “sample.split ( ) ” function is used which split data to train and test better than random splitting. With these parameters linear regression model is generated by “lm ( )” function using train data set and also forecasted values for test data set are generated. To evaluate quality of model MAPE value cannot be calculated because real sold\_counts 0 most of the time so only MAE value is calculated as 0.7288984. Some of the forecast values were negative and most of them were quite close to 0, since most of the sold\_count values close to 0. Then it is decided to round them and assign to 0 if it is negative. Then MAE value decreased to 0.6160714.



The Summary of the linear regression model is shown below.

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 0.7459790 0.4385992 1.701 0.090220 .

lag2\_sold\_count -0.2898724 0.1107622 -2.617 0.009409 \*\*

lag3\_sold\_count -0.1412884 0.0915347 -1.544 0.123962

lag4\_sold\_count -0.0046802 0.0671520 -0.070 0.944492

lag7\_sold\_count 0.0941432 0.0622195 1.513 0.131522

lag1\_price -0.0076880 0.0037663 -2.041 0.042276 \*

lag2\_visit\_count -0.0003731 0.0013448 -0.277 0.781661

lag2\_favored\_count 0.1745972 0.0458613 3.807 0.000177 \*\*\*

lag2\_basket\_count -0.1491106 0.1035188 -1.440 0.151000

lag2\_category\_sold -0.0004370 0.0003673 -1.190 0.235325

lag2\_category\_brand\_sold 0.0096540 0.0032357 2.984 0.003131 \*\*

lag2\_amount 0.0076941 0.0087685 0.877 0.381074

**5.2.2 Naive Model**

As usual Naïve approach is also used to generate model Then, Lag 2 sold\_count value is used as forecast directly. As before MAPE value can’t be calculated because of sold\_count values being 0 most of the time. MAE values of Naïve approach is also calculated as 0.7142857

**5.3 Results**

In order to evaluate which model is better, another method is generated. For each test data, difference between forecast and sold\_count is calculated. For each day, result of the model closer to sold\_count value is assigned to a new column. Then linear regression model made better predictions for 81.25% of test data. For both measurements (MAE and the new method) linear regression is better model for coat data set. (Table 2)

*Table 5*

**6. Product ID= 4066298**

**6.1 Product Based Analysis**

This product is a wet towel for newborn children, and it is called Sleepy Natural Yenidoğan Islak Pamuklu Havlu. The product is a special kind of wet towel which is used for the sensitive skins of newborn children when changing their diapers. Lately the sales of wet wipes performed a steep increase due to the COVID-19 outbreak. However, the same increase is not observed for Sleepy Yenidoğan Islak Havlu product. The reason for that lies behind the usage of the product. This product is not used for disinfection in daily life since it is an expensive type of wet towel which designed for babies and for the processes in changing their diapers.

**6.2 Approach**

A screenshot of a cell phone

Description automatically generatedFirst, to have a general understanding of the data which will be predicted, a sold count and date plot is visualized (Figure 28). The immediate action for this data is taken by eliminating the part of the data which sold count was 0. The reason for the 0 sold count is most probably the product was not available in that period. The other outlier values in the data were due to special events. Since there are not same special events for the prediction interval (16/05/2020 – 13/06/2020) of this study, the sold count for the past special event dates are manipulated by taking average of the closest possible normal dates’ sold count. Also, there is one day in the middle of the data which has sold count 0 because of the stock out. That point of the data also smoothed by taking the average of the nearest two data points. There are also two special events in the forecasting period. However, for the both events there aren’t any past data to evaluate the change in the sold count in that period. So, the two special events are treated as normal dates in the prediction of the models. The last shape of the sold count data by date which will be used by the A close up of a black background

Description automatically generatedpredictive models can be seen in the Figure 29.

*Figure 28*

*Figure 29*

**6.2.1 Naïve Model by Lag 2 Sold Count**

Before constructing any model, the autocorrelation coefficients for sold count data is observed (Figure 30). The correlation coefficients of lag 2 and lag 3 values are significantly high than any other lag values. Therefore, using the lag 2 value of the sold count to predict the sold count for two days later is the base model for this product. Any model which performs better than Naïve Model by Lag 2 Sold Count can be seen as a useful model.

*Figure 30*

**6.2.2 Naïve Model by Lag 3 Sold Count**

The autocorrelation coefficient of lag 3 is close to the lag 2’s autocorrelation coefficient (see Figure 3). Therefore, this Naïve model is also used and observed in the evaluation part of the models.

**6.2.3 Linear Regression Model**

To build a regression model, 4 explanatory variables (independent variables) are used. The independent variables are lag 2 sold count which is the latest (yesterday’s) sold count data that is available, lag 3 sold count which is the sold count data for the day before yesterday, lag 2 price of the product which is the latest (yesterday’s) price data that is available, and lastly lag 4 Google search popularity for “Sleepy Islak Mendil”. The last independent variable is obtained from Google Trends. However, the latest data available for the prediction day in Google Trends is the popularity of Google search of 4 days before the prediction day. Therefore, the latest data available is used in linear regression model.

**6.2.4 Model by Trend and Seasonality**

In this model, seasonality of the sold count data was the main factor. To be able to observe a seasonality pattern, first the trend (obtained by simple linear regression) should be removed (Figure 31). After subtracting the trend, the averages for each day of the weak is calculated. From that calculation the weekly seasonality is determined. According to the averages for each day of the week, “Monday, Saturday and Sunday” has negative effects in the sales of Sleepy Wet Towel (see Figure 32). Lastly, the effect of the days of the week is subtracted to check the residuals. Even though the residuals don’t look like white noise, the model still might be useful. The usefulness of the model can be observed in the evaluation part.

*Figure 31*

*Figure 32*

**6.2.5 Ensembled Model**

The Ensembled Model is actually obtained after the evaluation of the models. After making predictions to evaluate models, the predictions of linear regression are usually under the sold count and the predictions of the model by trend and seasonality is usually over the sold count (see Figure 33). Therefore, an ensembled model which is a mix of the both models mentioned is created by taking the average of the two prediction values of those models.

*Figure 33*

**6.3 Results**

A close up of a logo

Description automatically generatedTo decide which model performs the best among the five models created for Sleepy Wet Towel product, test date interval is first determined (2020-03-01 to 2020-05-15). In other words, the test interval is 2.5 months. Then starting from the first day of the test date the models make predictions with the past data (n data points). Then the date updates and the train data for the models also updates (n+1 data points). To make a healthy evaluation, there are 4 error parameters calculated for each model (Table 6). The parameters are Mean Squared Error (MSE), Mean

*Table 6*

Absolute Deviation (MAD), Mean Absolute Percentage Error (MAPE), and Absolute Percentage error in the 90% quantile (APE QUANTILE 90). The most important parameter for this study is MAPE since the leaderboard for the models are evaluated by MAPE. The ensembled model which is named in Table 1 performs with the lowest MAPE. Also, the other parameters for ensembled model is also performs best. Therefore, the predictions will be made by using Ensembled Model for Sleepy Wet Wipes.

**7. Product ID= 6676673**

**7.1 Product Based Analysis**

A screenshot of a social media post

Description automatically generatedThe product “Xiomi Redmi Airdots Tws Bluetooth Kulaklık” has become very popular in Turkey due to its price & performance. It is an earpods with a Bluetooth connection. The product is also preferred since the other well-known brands’ Bluetooth earpods are too expensive when compared with Xiomi’s. The general category for this product is “Bluetooth Kulaklık” in Turkish and the google search of this category according to Google Trends (Figure 34) can be interpreted as the category itself has a high popularity for the last months.

Figure 34 – Google Trends for “Bluetooth Kulaklık” term

**7.2 Approach**

At first, to have a general idea about how the product sales perform in Trendyol, the sold count versus date is observed in Figure 35. The left tail of the data has quite low sold counts until 9th of the September in 2019 (most of them are below 10). After 9th of the September the sales jumps suddenly to one thousand per day for 3 days. That jump could lead to a bad performance in the predictive models. To be able to develop healthy models, the left tail of the data until 13-09-2019 is removed. Other than left tail of the sold count data, as it can be seen obviously, for some dates the sales are skyrocketed. When the dates are analyzed in detail, it is seen that they are the special event dates in Trendyol. However, for this study, there are not any of the special events, that has occurred, in the prediction interval. Therefore, the sold count data in the past A picture containing sitting, large, table, water

Description automatically generatedspecial events are smoothed by taking average of the closest sold count data points. There is also one day which the product was not available to be sold, therefore that day has also been smoothed by the closest two data points. After all the manipulations on the sold count data, the sold count by date can be seen in the Figure A close up of text on a white background

Description automatically generated36 With the data in Figure 36, the models are built.

Figure 35

Figure 36

**7.2.1 Naïve Model by Lag 2 Sold Count**

Before constructing any complicated model, the first approach to predict the sold count data is a naïve model which uses the two days’ before sold count data as a prediction. To be able to decide on any other models, this naïve model will be compared with it since the autocorrelation coefficient is significantly (0.6) high as it can be seen in the Figure 37.

Figure 37

**7.2.2 Naïve Model by Lag 3 Sold Count**

The autocorrelation coefficient of lag 3 is close to the lag 2’s autocorrelation coefficient (see Figure 4). Therefore, this Naïve model is also used and observed in the evaluation part of the models.

**7.2.3 Linear Regression Model**

To build a linear regression model, a variety of combinations of 6 different independent variables (different lag values of sold count, lag 2 price, lag 4 google search) used in the model and evaluated by the evaluation process mentioned in the 2.3 Result part of the report. In the end, the 4 different variables are all used. The first independent variable used is lag 2 sold count which is the latest (yesterday’s) sold count data that is available have. The second independent variable used is lag 3 sold count. The third one is the lag 2 price which is the latest price available. At last, the fourth independent variable used is the lag 4 (4 days ago) “Bluetooth kulaklık” search popularity provided by Google Trends. Lag 4 search popularity is used because the latest data available for the prediction day in Google Trends is the popularity of Google search of 4 days before the prediction day. Therefore, the latest data available is used in linear regression model.

**7.2.4 Model by Trend and Seasonality**

The seasonality for sold count data cannot be noticed by looking at the Figure 3 at first glance. However, a basic approach is used to determine if there are seasonality for the days of the month and the days of the week. To be able to see a seasonality, first the trend (obtained by simple linear regression) should be removed (Figure 38). After the trend is subtracted from sold count data, the averages for each days of the month are calculated. By observing the averages (see Figure 39), the sold count for the second half of the months are mostly lower than the first part of the months. Then, the day of the month effects are also subtracted from the sold count data. The day of the week averages are also calculated. However, as it can be seen in the performance evaluation in 2.3 Result part, the prediction performance of this model is quite low when compared with other models in this study. Therefore, it is possible to say that the detected seasonality of the data is not important since the sold count values are much higher than the seasonality effect counts.

Figure 38

Figure 39

**7.2.5 Ensembled Model**

After evaluating models (see 7.3 Results part of the report), the performance of some mixture of models are observed. A combination of lag2 sold count, lag3 sold count, and linear regression model performed the best. The ensembled model is created by taking the 40% of the lag2 sold count, 40% of the lag3 sold count and 20% of the linear regression’s prediction.

**7.3 Results**

A close up of a logo

Description automatically generatedTo be able to evaluate the performances of the 5 different models, first the test date interval is determined (2020-03-01 to 2020-05-15). In other words, the test interval is 2.5 months. Then starting from the first day of the test date the models make predictions with the past data (n data points). Then the date updates and the train data for the models also updates (n+1 data points). To make a reliable evaluation, there are 4 error parameters calculated for each model (Table 1). The parameters are Mean Squared Error (MSE), Mean Absolute Deviation (MAD), Mean Absolute Percentage Error (MAPE), and Absolute Percentage error in the 90% quantile (APE QUANTILE 90). The most important parameter for this study is MAPE since the leaderboard for the models are evaluated by MAPE. The evaluation of the models in Table 7 demonstrates that the ensembled model performs slightly better than all the other models. Therefore, the ensembled model will be used to make predictions for “Xiomi Redmi Airdots Tws Bluetooth Kulaklık”.

*Table 7*

**8. Product ID=7061886**

**8.1 Product Based Analysis**

This product is Fakir Lucky vacuum cleaner. According to the customer reviews on the site it is a durable machine that gives a good performance considering its price. The naming policy of company is giving different model names for different colors of vacuum cleaners (Lucky for beige, Darky for purple, Starky for grey etc.). Which makes it hard to use the Google trends data.

**8.2 Approach**

The product has started to be sold at 26-07-2019. From that date it has been sold on Trendyol in a consistent manner with little exceptions. These exceptions are mostly the same ones that have been mentioned above, a huge increase in November due to Trendyol’s Black Friday-ish discount campaign and some other smaller marketing works. Besides that there are few days that no sales happened.

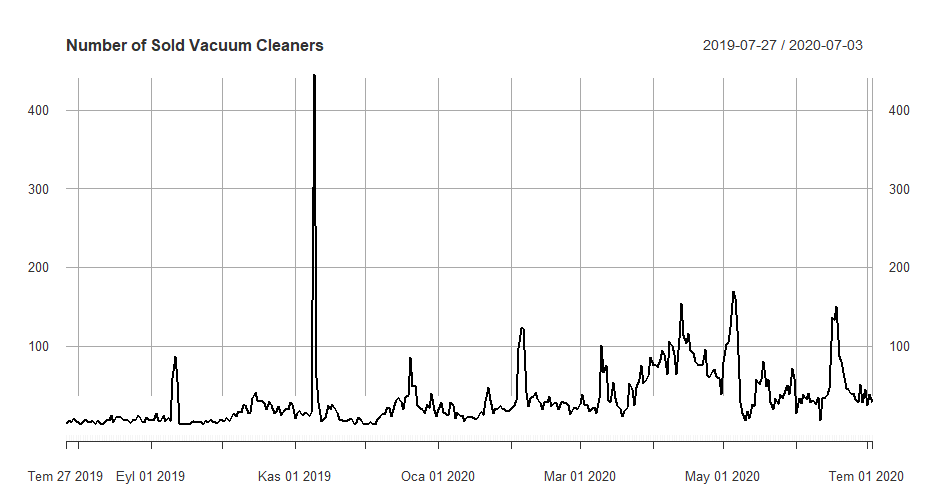


Figure 40

As it can be seen at figure above the number of sales at 09-11-2019 is incredibly higher than the rest of the data. This is mostly due to marketing success. The price (179 liras) at this date is smaller than the average price (208 liras). Yet there are many days that are cheaper but reached nowhere near of sale numbers of 09-11-2019. Which makes it necessary to smooth it.

In addition to smoothing data for sales that exceeded predictable range, there are few days that no sales have occurred. This is mostly due to unavailability of product. And the prices of these dates are -1 in the original data as it was mentioned in previous products which will corrupt the relationship between the sales and price.

After the issues above are solved, the modeling can be done. The two main approaches are linear regression modeling and arima. The prepared models will be compared to naïve method. Since the forecast horizon is 2, the naïve method will use the sold count from two days ago.

The first model is done by linear regression method, with the only smoothed data are the ones mentioned above. The data is divided into two: train and test. Two additional columns are added for each row. First one is forecast, it is basically lag=(-2) since the main logic is trying to predict the number of sales of two days later with the present information. That column helps to build a relationship. The second column is pricel1. It represents the today’s price to forecast tomorrow. The information about today, isn’t in the data. Yet the price can be put in manually by checking the site.

**8.2.1 Naïve Method**

To measure models RMSE method has been used. Since the threshold of the models is about being better that naïve method. It is useful to calculate it first.

> RMSE(vacuumnet[336:342]$sold\_count,arimavactest$sold\_count)

[1] 16.93264

**8.2.2 Linear Model**

After some trial and error process, the best result for specified test data comes from the model below.

Call:

lm(formula = forecast ~ price + sold\_count + visit\_count + pricel1,

data = lmtrain)

Coefficients:

(Intercept) price sold\_count visit\_count pricel1

-17.850530 1.154346 0.692712 -0.003463 -0.982314

According to the model while the price of yesterday having a positive effect, today’s price has negative effect on tomorrow’s sold count. Even though this doesn’t make sense in economical way, it shouldn’t be forgotten that Corona Pandemic has changed the shopping behavior drastically. People got more likely to buy products from internet instead of going to a shopping mall or a technology store. When this idea mixes with the increase of prices in Turkey, this positive correlation between price and sold counts makes a bit more sense. Apparently, the sales are autocorrelated, as the sold count has a significantly big coefficient.

The RMSE test result of this model is:

> RMSE(lmpredfinal,arimavactest$sold\_count)

[1] 9.348883

Which is a huge improvement in comparison with naïve method.

**8.2.3 ARIMA Model**

Since arima models heavily relies on stationarity of data, decompose function has been used at the first step of the arima model.

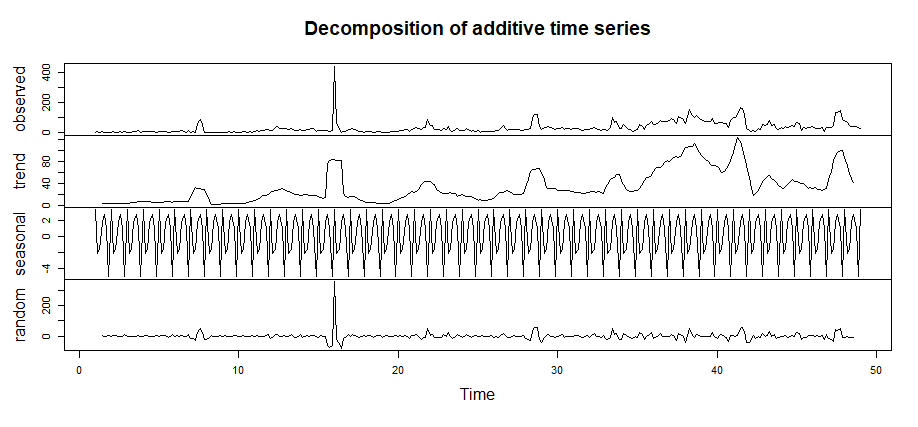


Figure 41

As it was mentioned in the above, the sales of 09-11-2019 is too big to be decomposed by this function when frequency=7. Apart from that, the rest of the random plot looks stationary enough to give successful results with arima. After smoothing the 09-11-2019 numbers the plot of decompose function becomes:

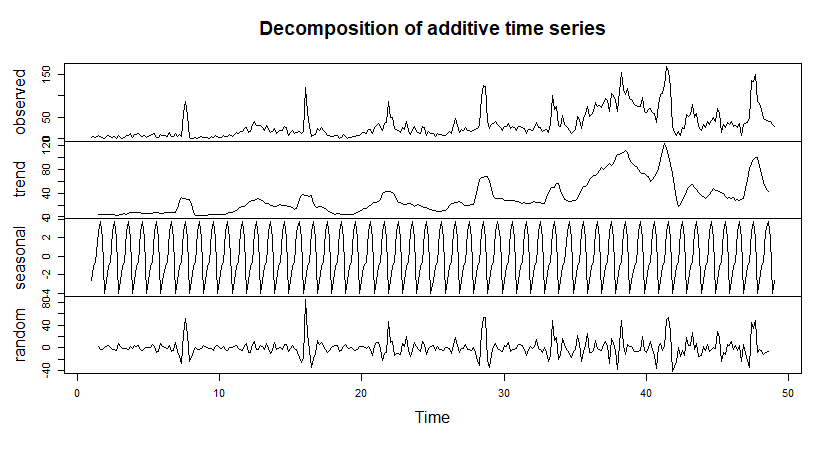


Figure 42

From the graph, it can also be said that “non-smoothed outlier” corrupts seasonality in addition to random results.

Since the random results are good to go, arima models can be applied. However at this point, it shouldn’t be forgotten that to get the length of random same as the length of original data, trend part needs to be extended. Shifting had been applied for first and last N/A values. This makes random as long as data.

At this point, auto.arima function gives (0,0,1) as the best result yet manually some other better results can be found. (5,0,2) gives a better result than the other tries. The rest of the arima results and predictions are based on it.

As they are subtracted in the first place, season and trend had to be added back to the forecasted “random”.

The RMSE of this model is:

> RMSE(valuesarimavac,arimavactest$sold\_count)

[1] 14.94042

This result shows that arima model is better than naïve approach. Yet it is worse than linear model.

**8.3 Results**

From the initial results, it is safe to say that linear model is the one that gives the best results. However, it is useful to check if there is any room for improvement. To do such a thing, ensembling strategy is a good way. All three models are equally weighted for ensemble model.

> RMSE(naive\_vacuum,real\_values\_vacuum)

[1] 16.93264

> RMSE(linear\_values\_vacuum,real\_values\_vacuum)

[1] 9.348883

> RMSE(arima\_values\_vacuum,real\_values\_vacuum)

[1] 14.94042

> RMSE(ensembled\_model,real\_values\_vacuum)

[1] 11.95331

> MAPE(naive\_vacuum,real\_values\_vacuum)

[1] 0.462547

> MAPE(linear\_values\_vacuum,real\_values\_vacuum)

[1] 0.2630857

> MAPE(arima\_values\_vacuum,real\_values\_vacuum)

[1] 0.4075678

> MAPE(ensembled\_model,real\_values\_vacuum)

[1] 0.3355711

Theses MAPE results with ensembled method doesn’t change the winner. Linear model still gives the best results.

**9 Product ID=5926527**

**9.1 Product Based Analysis**

This product is a Trendyolmilla brand bikini top. The reviews are good. Yet it is nothing more than an ordinary black bikini top. Which means it is hard to reach even by googling “siyah bikini üstü”. The results coming from Trendyol site when asked for bikini top, doesn’t even show the bikini in the first 10 pages.

**9.2 Approach**

The data of bikini sales is pretty problematic. This is probably due to company’s selling policy. Yet it is worth mentioning. Unlike the most other products in this study, bikini top starts getting sold after few days of data starting at 2019-05-12. And keeps getting sold for the rest of the summer.

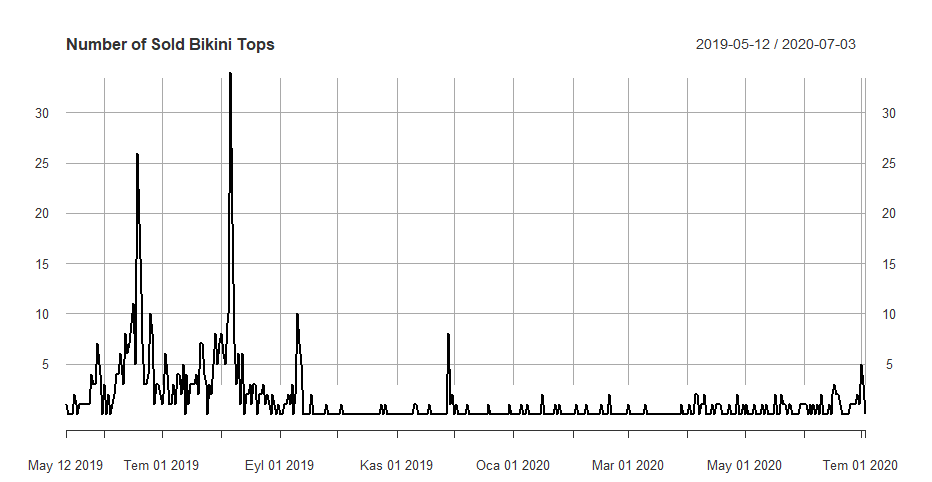


Figure 43

The problem begins with the end of the September 2019. The product stops getting sold. But, it doesn’t. Every week, few bikini tops got sold and for the rest of the week the product didn’t even go for sale. To be more precise, there are 180 days that this bikini top is sold any number and 239 days that didn’t even make any sale. This issue makes it harder to build a model. The empty spots can be filled with nearest values, but it isn’t chosen as there are more empty spots than the full ones. If the empties are filled with the rest, this would increase the weight of “neighbors” significantly. Since the proper data is from summer where the problem above didn’t occur, doing such an operation would only decrease the weight of it. So the models are based on data without any manipulation.

**9.2.1 Naïve Method**

The same method as the previous products. It is basically using the last available number as prediction. Since the data is full of small numbers between 0 and 5. It performs well.

> rmse(bikiniarima\_test[1:7]$sold\_count,realvalues)

[1] 2.360387

**9.2.2 Linear Regression Model**

While building the linear model; another problematic issue is deciding whether using price as an independent variable, or not. The bikini top had sold well for 2019 Summer, but at that time the price of the product was at its peak. Which means if it is used in the model it will show a big positive correlation. Besides, most of the data shows price as (-1) and there is no substitute for them. Because of them price isn’t included in the model. Also using data from outside such as google trends isn’t so feasible according to results caused by the reason given in **9.1**.

The model is:

Call:

lm(formula = forecast ~ sold\_count + visit\_count + favored\_count,

data = bikiniarima\_train)

Coefficients:

(Intercept) sold\_count visit\_count favored\_count

0.357245 0.312259 -0.004735 0.221298

As it was in vacuum cleaner data, this model also tries to predict “forecast” column that have been added later. The model is simple. Using only the “uncorrupted” pieces of the data. The performance of it, is:

> rmse(haspredictions,realvalues)

[1] 2.004977

**9.2.3 ARIMA Model**

To make the data stationary, first step is decomposing.

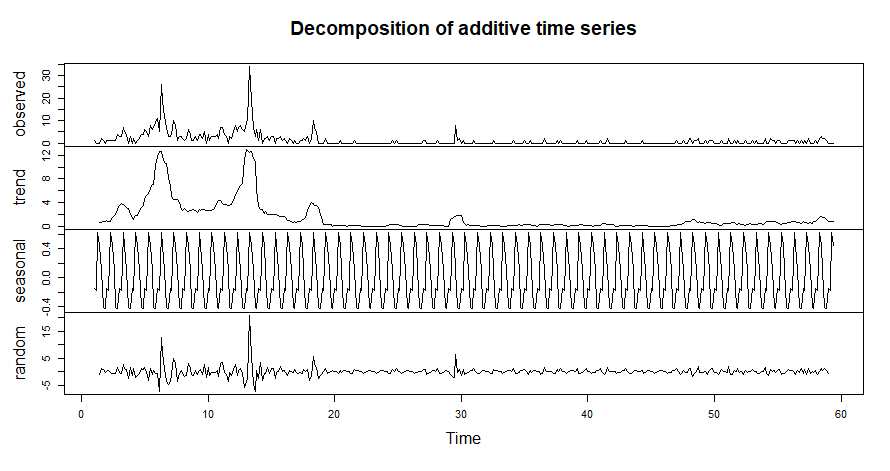


Figure 44

After some point the random is stationary since it mostly consists of zeroes and ones in the observed part. Before constructing the model, standard procedure is applied. Extending the trend and getting the “long” random. The model used for forecasting with arima is, (1,0,1). The RMSE of the model is:

> rmse(testbikini,realvalues)

[1] 1.992786

The result is little better than linear one’s. Yet it might be improved by ensembling.

**9.3 Results**

The test data is the last week of the data.

> rmse(lmbikini\_pred,realvalues)

[1] 2.004977

> rmse(arimabikini\_pred,realvalues)

[1] 1.992786

> rmse(naivebikini\_pred,realvalues)

[1] 2.360387

> rmse(ensembledbikini,realvalues)

[1] 1.78202

These results show that the best prediction method is the mean of arima and linear model. The MAPE test didn’t applied as there are some zeroes in the real values causing all of them to result infinite.

**Appendix**

1. **Code for Figure 2**

plot\_ly(x=posay$event\_date,y=posay$sold\_count, mode="lines")

1. **Code for Figure 3**

plot\_ly(x=posay$event\_date,y=posay$sold\_count, mode="lines")

1. **Code for Figure 4**

acf(posay$sold\_count,main="Autocorrelation of Sold Count")

1. **Code for Figure 5**

**#**Without data manipulation

checkresiduals(arima\_model\_product8)

1. **Code for Figure 6**

**#**With data manipulation

checkresiduals(arima\_model\_product8)

1. **Code for Figure 7**

forecast\_product8 <- forecast(arima\_model\_product8, h=nrow(product8\_data\_test))

plot(forecast\_product8)

1. **Code for Figure 8**

plot(test\_numeric,type='l',ylab="sold\_count ",ylim=c(0,100))

par(new=TRUE)

plot(forecast\_numeric,col='Red',type = 'l',ylab=" ",ylim=c(0,100),main="Actual vs Predicted Sold Counts")

legend("bottomleft", legend=c("Actual Sold Count", "Predicted Sold Count"),

col=c("black", "red"), lty=1:1, cex=0.5,text.width = 15, box.lty=1)

1. **Code for Table 2**

ppp<- train

ppp$event\_date<- as.numeric(ppp$event\_date)

cor(ppp)

1. **Code for Table 3**

vif(lm\_model)

1. **Code for Figure 9**

plot(ppt$sold\_count,type='l',ylab=" ",ylim=c(0,300))

par(new=TRUE)

plot(ppp\_prediction\_posay,col='Red',ylim=c(0,300),type ='l',main="Actual vs Predicted Sold Counts")

legend("topright", legend=c("Actual Sold Count", "Predicted Sold Count"), col=c("black", "red"), lty=1:1, cex=0.5,text.width = 15,box.lty=1)

1. **Code for Figure 10**

plot(last67\_naive\_posay,type='l',ylab=" ")

par(new=TRUE)

plot(last67\_lagged\_posay,col='Red',type = 'l',main="Actual vs Predicted Sold Counts")

legend("bottomleft", legend=c("Actual Sold Count", "Predicted Sold Count"),

col=c("black", "red"), lty=1:1, cex=0.5,text.width = 15,box.lty=1)

1. **Code for Figure 11**

plot(test$sold\_count,type='l',ylim=c(0,185),ylab=" ")

par(new=TRUE)

plot(double\_mean\_forecast,col='Red',type = 'l',ylim=c(0,185),main="Actual vs Predicted Sold Counts")

legend("bottomleft", legend=c("Actual Sold Count", "Predicted Sold Count"),

col=c("black", "red"), lty=1:1, cex=0.5,text.width = 15,box.lty=1)

1. **Code for Figure 12**

#When

#ppp\_prediction\_posay[8]<-80

#ppp\_prediction\_posay[9]<-80

plot(test$sold\_count,type='l',ylim=c(0,185),ylab=" ")

par(new=TRUE)

plot(double\_mean\_forecast,col='Red',type = 'l',ylim=c(0,185),main="Actual vs

Predicted Sold Counts")

legend("bottomleft", legend=c("Actual Sold Count", "Predicted Sold Count"),

col=c("black", "red"), lty=1:1, cex=0.5,text.width = 15,box.lty=1)

1. **Code for Figure 14**

plot(oralb$event\_date,oralb$sold\_count,type='l',main="Sold Count in Time")

1. **Code for Figure 15**

acf(product2\_data$sold\_count, lag.max = 60)

1. **Code for Figure 16**

checkresiduals(arima\_model\_product2)

1. **Code for Figure 17**

plot(forecast\_product2)

1. **Code for Figure 18**

plot(test\_oralb$sold\_count,type='l',ylab="sold\_count",ylim=c(0,500),main="Actual vs Predicted Sold Counts")

par(new=TRUE)

plot(prediction\_oralb,col='Red',type = 'l',ylab=" ",ylim=c(0,500))

legend("topright", legend=c("Actual Sold Count", "Predicted Sold Count"), col=c("black",

"red"), lty=1:1, cex=0.5,box.lty=1)

1. **Code for Figure 19**

plot(last67\_naive\_oralb,type='l',ylab=" ")

par(new=TRUE)

plot(last67\_lagged\_oralb,col='Red',type = 'l',main="Actual vs Predicted Sold Counts")

legend("topright", legend=c("Actual Sold Count", "Predicted Sold Count"),

col=c("black", "red"), lty=1:1, cex=0.5,text.width = 15 ,box.lty=1)

1. **Code for Figure 21**

acf(tayt\_data$sold\_count, main="Correlation of Sold Count")

1. **Code for Figure 22**

plot(TrendyolGraph$event\_date,TrendyolGraph$amount, type="l",xlab=" ",ylab=" ")

par(new=TRUE)

plot(temp$event\_date,temp$ty\_visits, type="l", col="blue",yaxt = "n",main = "Google vs ty\_visit Comparison",xlab="Event Date",ylab="Values")

legend("topleft", legend=c("Google Data", "ty\_visit"), col=c("black", "blue"), lty=1:1, cex=0.8)

1. **Code for Figure 23**

plot(tayt\_data\_temp$event\_date,tayt\_data\_temp$sold\_count,type="l",main = "sold\_count by event\_date",xlab="event\_date",ylab="sold\_count", ylim=c(0, 1900))

par(new=TRUE)

plot(tayt\_data$event\_date,tayt\_data$sold\_count,type="l",col = "blue" ,main = "",xlab="",ylab="",ylim=c(0, 1900))

legend("topleft", legend=c("Original Data", "After Manipulation"), col=c("black", "blue"), lty=1:1, cex=0.8)

1. **Code for Figure 24**

plot(decompose\_tayt)

1. **Code for Figure 25**

plot(mont\_google$event\_date,mont\_google$amount, type="l",main = "Google Search of MONT ",xlab="Event Date",ylab="Amount")

1. **Code for Figure 26**

plot(decompose\_mont)

1. **Code for Figure 27**

plot(mont\_data$event\_date,mont\_data$sold\_count, type="l",main = "Sold Count of Coat By Event Date",xlab="Event Date ",ylab="Sold Count")

1. **Code for Figure 28**

fig <- plot\_ly(x=product4\_data$event\_date,y=product4\_data$sold\_count, mode="lines")

fig <- fig %>% layout(title = 'Sold Count by Date',

xaxis = list(title = 'Date'),

yaxis = list(title = 'Sold Count'))

fig

1. **Code for Figure 29**

fig2 <- plot\_ly(x=product4\_data$event\_date,y=product4\_data$sold\_count, mode="lines")

fig2 <- fig2 %>% layout(title = 'Sold Count by Date',

xaxis = list(title = 'Date'),

yaxis = list(title = 'Sold Count'))

fig2

1. **Code for figure 30**

acf(product4\_data$sold\_count, lag.max = 60, main = "Autocorrelation for Sold Count")

1. **Code for figure 31**

matplot(filtered\_product4[,list(sold\_count,lr\_trend)], type="l",

xlab = "Time Index", ylab = "Sold Count", main = "The Trend in Sold Count")

legend("topright" , legend = c("Sold Count", "Trend"), col = c("black", "red"), cex=0.5, lty=1:1)

1. **Code for table 6**

summary\_result[,list(mse=mean(se),mad=mean(ad),mape=mean(ape),ape\_quantile90=quantile(ape,0.9)),by=list(variable)]

1. **Code for figure 33**

plot(merged\_pred\_datatable$event\_date, merged\_pred\_datatable$sold\_count, type = "l", col = "red", xlab = "Date", ylab = "Sold Count", main = "Sold Count and Predictions")

lines(merged\_pred\_datatable$event\_date, merged\_pred\_datatable$Model\_by\_Trend\_and\_Seasonality, type = "l", col = "blue")

lines(merged\_pred\_datatable$event\_date, merged\_pred\_datatable$Linear\_Regression\_Model, type = "l", col = "green")

legend("topright" , legend = c("Sold Count", "Linear Regression Model Prediction", "Model by Trend and Seasonality"), col = c("red", "green", "blue"), cex=0.5, lty=1:1)

1. **Code for figure 35**

fig <- plot\_ly(x=product6\_data$event\_date,y=product6\_data$sold\_count, mode="lines")

fig <- fig %>% layout(title = 'Sold Count by Date',

xaxis = list(title = 'Date'),

yaxis = list(title = 'Sold Count'))

fig

1. **Code for figure 36**

fig2 <- plot\_ly(x=product6\_data$event\_date,y=product6\_data$sold\_count, mode="lines")

fig2 <- fig2 %>% layout(title = 'Sold Count by Date',

xaxis = list(title = 'Date'),

yaxis = list(title = 'Sold Count'))

fig2

1. **Code for figure 37**

acf(product6\_data$sold\_count, lag.max = 60, main = "Autocorrelation for Sold Count")

1. **Code for figure 38**

matplot(filtered\_product6[,list(sold\_count,lr\_trend\_6)], type="l", xlab = "Time", ylab = "Sold Count", main = "Sales Trend")

1. **Code for figure 39**

matplot(filtered\_product6$day\_of\_the\_month, filtered\_product6$day\_of\_the\_month\_effect\_lr, col = "brown" ,type = "h" ,xlab = "Day of the Month", ylab = "Day of the Month Effect", main = "Day of the Month Effect")

1. **Code for table 7**

summary\_result[,list(mse=mean(se),mad=mean(ad),mape=mean(ape),quantile90=quantile(ape,0.9)),by=list(variable)]

1. **Code for figure 40**

vacuum.arima=data.table()

vacuum.arima$date=vacuumnet$event\_date

vacuum.arima$sold=vacuumnet$sold\_count

vacuum.arima=as.xts(vacuum.arima)

plot.xts(vacuum.arima$sold,main = "Number of Sold Vacuum Cleaners")

1. **Code for figure 41**

dcvac0=decompose(ts(vacuumnet$sold\_count,frequency = 7))

plot(dcvac0)

1. **Code for figure 42**

dcvac=decompose(ts(arimavactrain$sold\_count,frequency = 7))

plot(dcvac)

1. **Code for figure 43**

bikinixts=data.table()

bikinixts$date=bikini$event\_date

bikinixts$sold=bikini$sold\_count

plot.xts(as.xts(bikinixts),main = "Number of Sold Bikini Tops")

1. **Code for figure 44**

todcbikini=ts(bikini$sold\_count,frequency = 7)

dcbikini=decompose(todcbikini)

plot(dcbikini)