time series forecast peak shaving

Gabrielle Nyirjesy

2022-12-03

```
library(dplyr)
library(xts)
library(lubridate)
library(forecast)
library(ggplot2)
library(Metrics)
library(stringr)
```

Data pre-processing

```
df <- read.csv('./data/demand_unpivoted.csv')
head(df)
```

```
##   X       Date period demand year month day           timestamp
## 1 0 2018-01-01      0 734.40 2018     1   1 2018-01-01 00:00:00
## 2 1 2018-01-01     15 731.52 2018     1   1 2018-01-01 00:15:00
## 3 2 2018-01-01     30 727.20 2018     1   1 2018-01-01 00:30:00
## 4 3 2018-01-01     45 735.84 2018     1   1 2018-01-01 00:45:00
## 5 4 2018-01-01     60 730.08 2018     1   1 2018-01-01 01:00:00
## 6 5 2018-01-01     75 728.64 2018     1   1 2018-01-01 01:15:00
```

```
df$timestamp <- strptime(df$timestamp, format="%Y-%m-%d %H:%M:%S")
head(df)
```

```
##   X       Date period demand year month day           timestamp
## 1 0 2018-01-01      0 734.40 2018     1   1 2018-01-01 00:00:00
## 2 1 2018-01-01     15 731.52 2018     1   1 2018-01-01 00:15:00
## 3 2 2018-01-01     30 727.20 2018     1   1 2018-01-01 00:30:00
## 4 3 2018-01-01     45 735.84 2018     1   1 2018-01-01 00:45:00
## 5 4 2018-01-01     60 730.08 2018     1   1 2018-01-01 01:00:00
## 6 5 2018-01-01     75 728.64 2018     1   1 2018-01-01 01:15:00
```

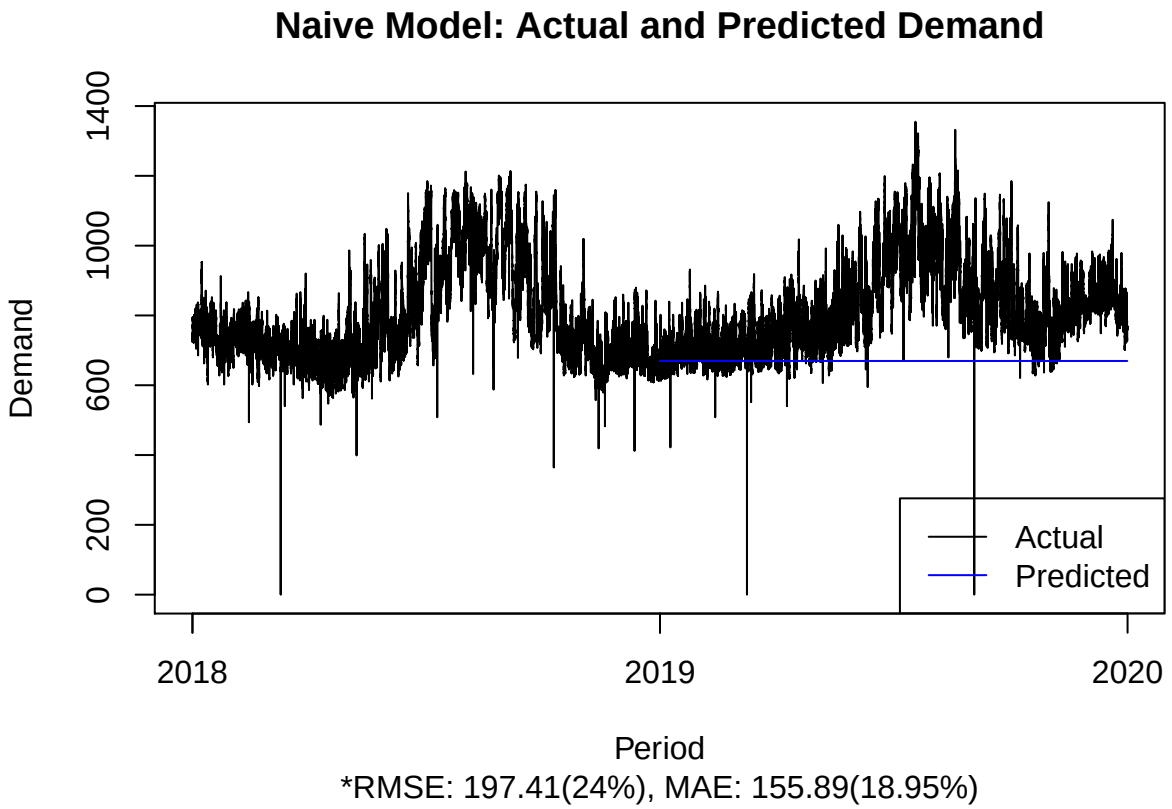
```
df_2018 <- subset(df, df$Date < as.POSIXct('2019-01-01 00:00:00') )
df_2019 <- subset(df, df$Date >= as.POSIXct('2019-01-01 00:00:00') )
df_ts <- data.frame(interval = seq(ymd_hms('2018-01-01 00:00:00'),
                                   by = '15 min',length.out=(length(df$demand))),
                    data = df$demand)
full_ts <- xts(df_ts$data, order.by=df_ts$interval)

df_ts_train <- data.frame(interval = seq(ymd_hms('2018-01-01 00:00:00'),
                                   by = '15 min',length.out=(length(df_2018$demand))),
                    data = df_2018$demand)
ts_2018 <- xts(df_ts_train$data, order.by=df_ts_train$interval)
```

```
df_ts_test <- data.frame(interval = seq(ymd_hms('2019-01-01 00:00:00'),
                                        by = '15 min',length.out=(length(df_2019$demand))),
                         data = df_2019$demand)
ts_2019 <- xts(df_ts_test$data, order.by=df_ts_test$interval)
```

Modeling

Naive Model

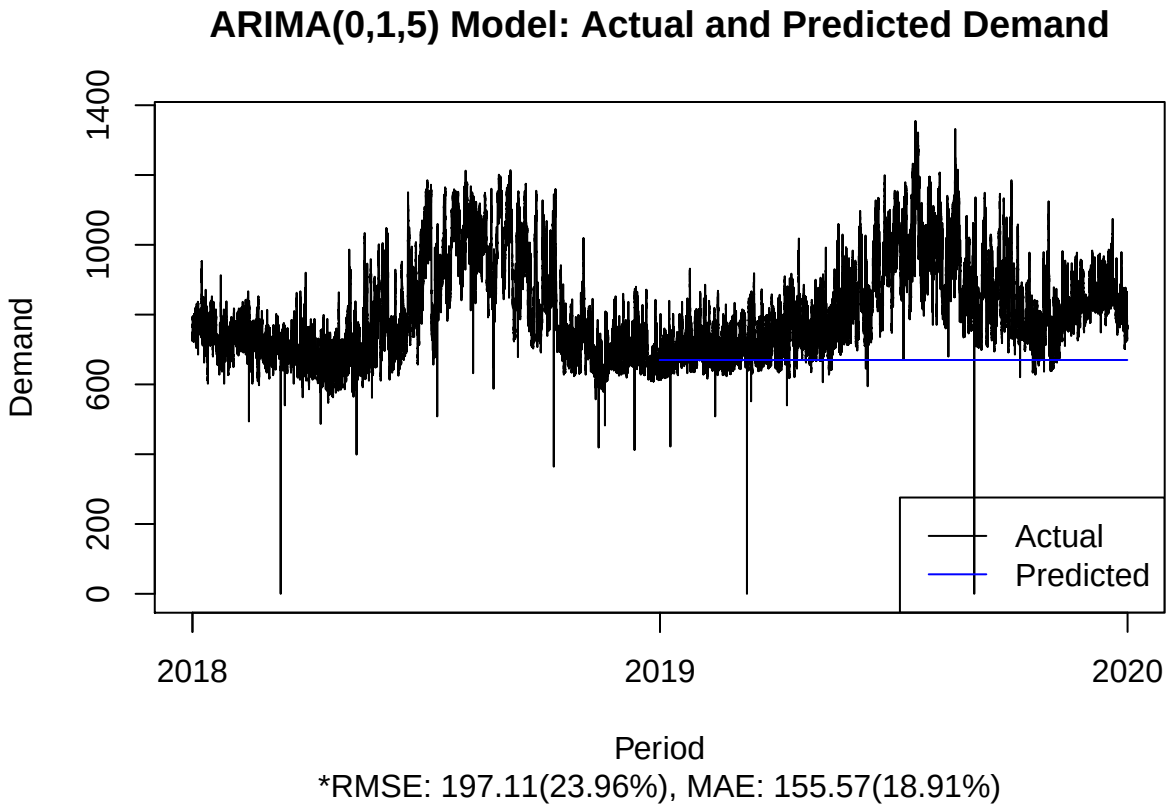
```
model <- naive(ts(df_ts_train$data, frequency = 96), level=c(80,95), h=length(df_ts_test$data))
rmse_error <- sqrt( mean( (df_ts_test$data-model$mean)^2 , na.rm = TRUE ) )
rmse_pct <- round(rmse_error/mean(df_ts_test$data)*100,2)
mean_abs_error <- sum(abs(df_ts_test$data-model$mean), na.rm = TRUE)/length(df_ts_test$data)
mae_pct <- round(mean_abs_error/mean(df_ts_test$data)*100,2)
plot(df_ts$interval, df_ts$data, xlab='Period',ylab='Demand', type='l', main = 'Naive Model: Actual and
     sub = paste0('*RMSE: ',round(rmse_error,2), '(',rmse_pct,'%)', ', MAE: ', round(mean_abs_error,2),
lines(df_ts_test$interval, model$mean, col='blue')
legend(x='bottomright', legend=c('Actual','Predicted'), col=c('black','blue'),lty=1)
```



ARIMA Model

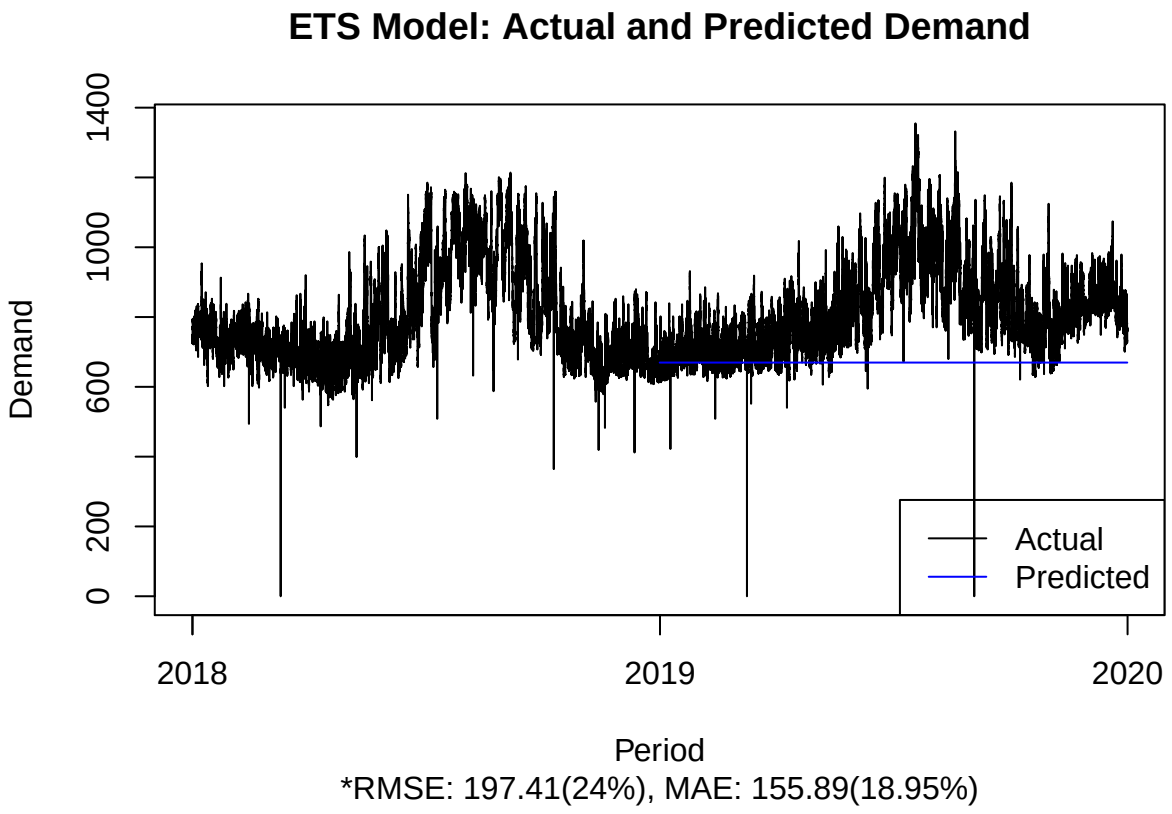
```
model <- auto.arima(ts(df_ts_train$data, frequency = 96),
                            stepwise = FALSE,
                            approximation = FALSE,
                            seasonal = FALSE,
                            ic = 'aicc')
```

```
model_preds <- forecast(model, level=c(80,95), h=length(df_ts_test$data))
rmse_error <- sqrt( mean( (df_ts_test$data-model_preds$mean)^2 , na.rm = TRUE ) )
rmse_pct <- round(rmse_error/mean(df_ts_test$data)*100,2)
mean_abs_error <- sum(abs(df_ts_test$data-model_preds$mean), na.rm = TRUE)/length(df_ts_test$data)
mae_pct <- round(mean_abs_error/mean(df_ts_test$data)*100,2)
plot(df_ts$interval, df_ts$data, xlab='Period',ylab='Demand', type='l', main = paste0(model_preds$metho
     sub = paste0('*RMSE: ',round(rmse_error,2), '(',rmse_pct,'%)', ', MAE: ', round(mean_abs_error,2),
lines(df_ts_test$interval, model_preds$mean, col='blue')
legend(x='bottomright', legend=c('Actual','Predicted'), col=c('black','blue'),lty=1)
```



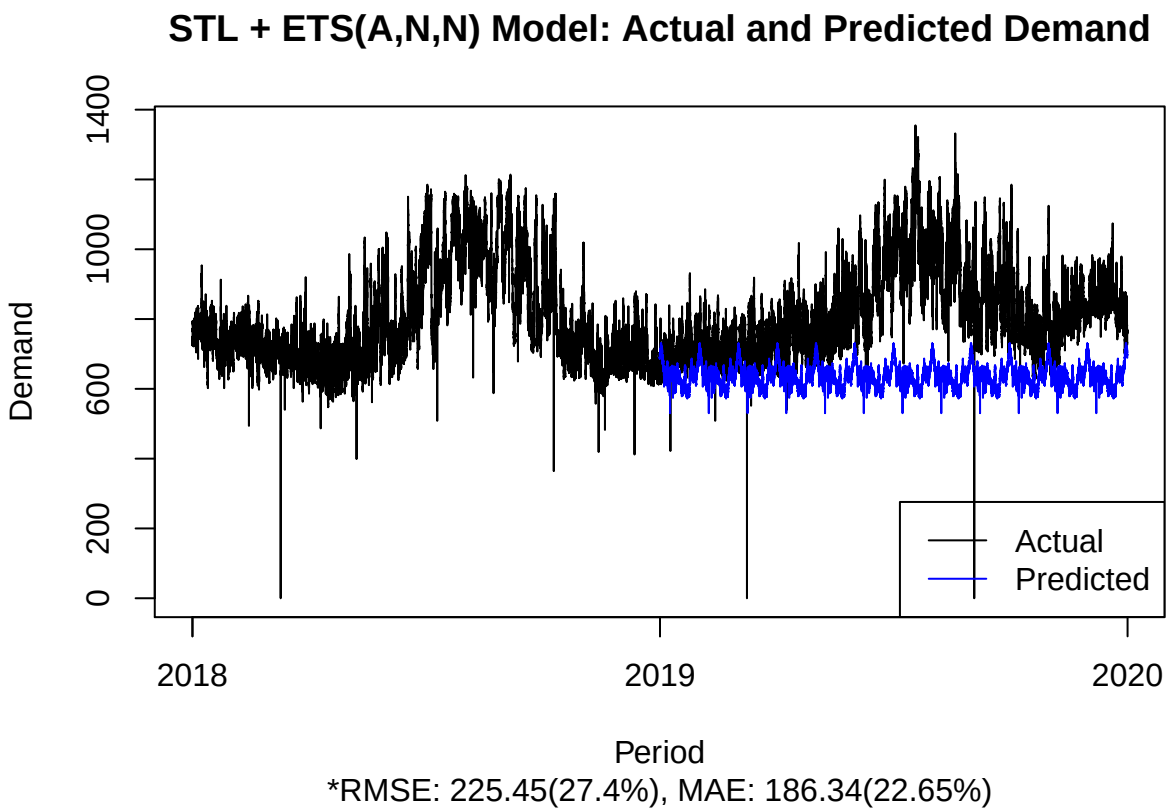
ETS Model

```
model <- ets(ts(df_ts_train$data, frequency = 96))
model_preds <- forecast(model, level=c(80,95), h=length(df_ts_test$data))
rmse_error <- sqrt( mean( (df_ts_test$data-model_preds$mean)^2 , na.rm = TRUE ) )
rmse_pct <- round(rmse_error/mean(df_ts_test$data)*100,2)
mean_abs_error <- sum(abs(df_ts_test$data-model_preds$mean), na.rm = TRUE)/length(df_ts_test$data)
mae_pct <- round(mean_abs_error/mean(df_ts_test$data)*100,2)
plot(df_ts$interval, df_ts$data, xlab='Period',ylab='Demand', type='l', main = 'ETS Model: Actual and Pr
     sub = paste0('*RMSE: ',round(rmse_error,2), '(',rmse_pct,'%)', ', MAE: ', round(mean_abs_error,2),
lines(df_ts_test$interval, model_preds$mean, col='blue')
legend(x='bottomright', legend=c('Actual','Predicted'), col=c('black','blue'),lty=1)
```



STLF Model

```
freq = 96*30.25
model <- stlf(ts(df_ts_train$data, frequency = freq), h=length(df_ts_test$data))
rmse_error <- sqrt( mean( (df_ts_test$data-model$mean)^2 , na.rm = TRUE ) )
rmse_pct <- round(rmse_error/mean(df_ts_test$data)*100,2)
mean_abs_error <- sum(abs(df_ts_test$data-model$mean), na.rm = TRUE)/length(df_ts_test$data)
mae_pct <- round(mean_abs_error/mean(df_ts_test$data)*100,2)
plot(df_ts$interval, df_ts$data, xlab='Period',ylab='Demand', type='l', main = 'STL + ETS(A,N,N) Model:
     sub = paste0('*RMSE: ',round(rmse_error,2), '(',rmse_pct,'%)', ', MAE: ', round(mean_abs_error,2),
lines(df_ts_test$interval, model$mean, col='blue')
legend(x='bottomright', legend=c('Actual','Predicted'), col=c('black','blue'),lty=1)
```



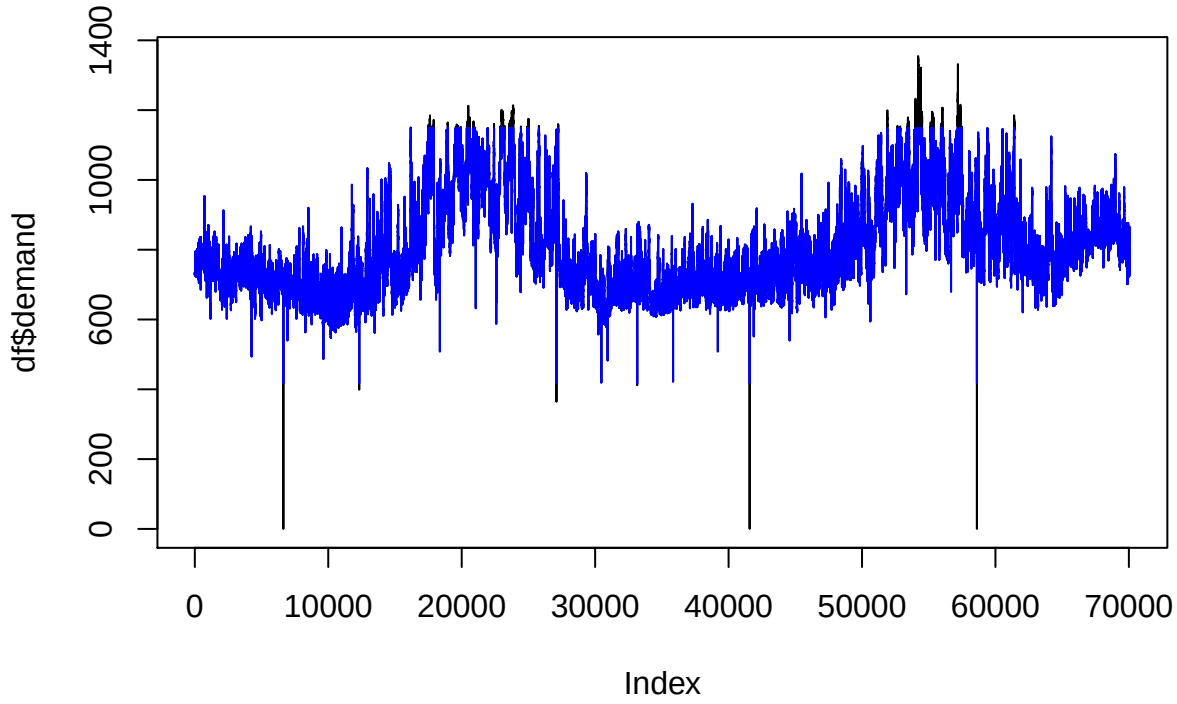

```
#Save the predicted demand
write.csv(model$fitted, './data/stlf_predicted_2019_demand.csv')
```

STLF model captures the trend the best, so use this moving forward

Pipeline for Optimization Model

Train model on 2018 only data, create forecast, update each month and re-train the forecast Try data smoothing

```
lower_quantile <- quantile(subset(df, df$Date < as.POSIXct('2019-01-01 00:00:00'))$demand, .25)
upper_quantile <- quantile(subset(df, df$Date < as.POSIXct('2019-01-01 00:00:00'))$demand, .75)
inter_quartile_range <- IQR(subset(df, df$Date < as.POSIXct('2019-01-01 00:00:00'))$demand)
lower_outlier_cutoff <- lower_quantile - 1.5*inter_quartile_range
upper_outlier_cutoff <- upper_quantile + 1.5*inter_quartile_range
#Replace outliers in demand
df$smoothed <- ifelse(df$demand < lower_outlier_cutoff, lower_outlier_cutoff,
                      ifelse(df$demand > upper_outlier_cutoff, upper_outlier_cutoff, df$demand))
plot(df$demand, type='l')
lines(df$smoothed, type='l', col='blue')
```



Create empty column to fill with forecasts

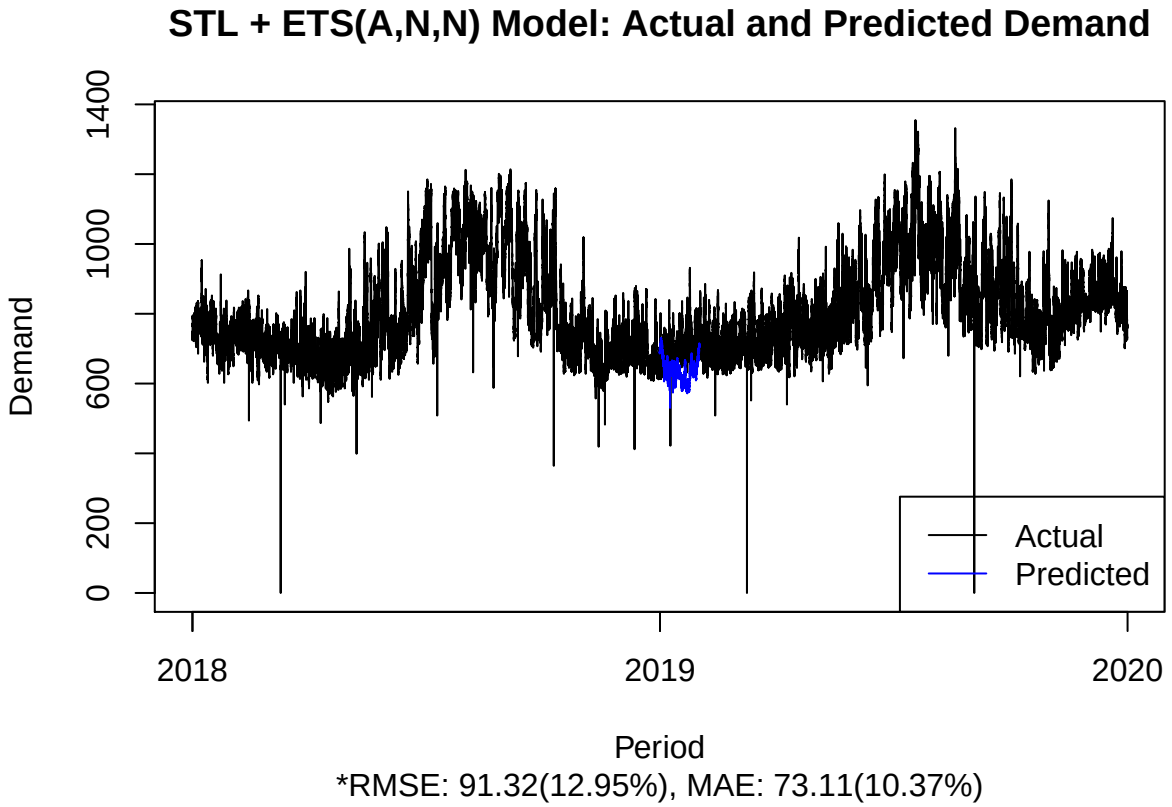
```
df['forecast'] <- NA
start_index = 1
head(df)
```

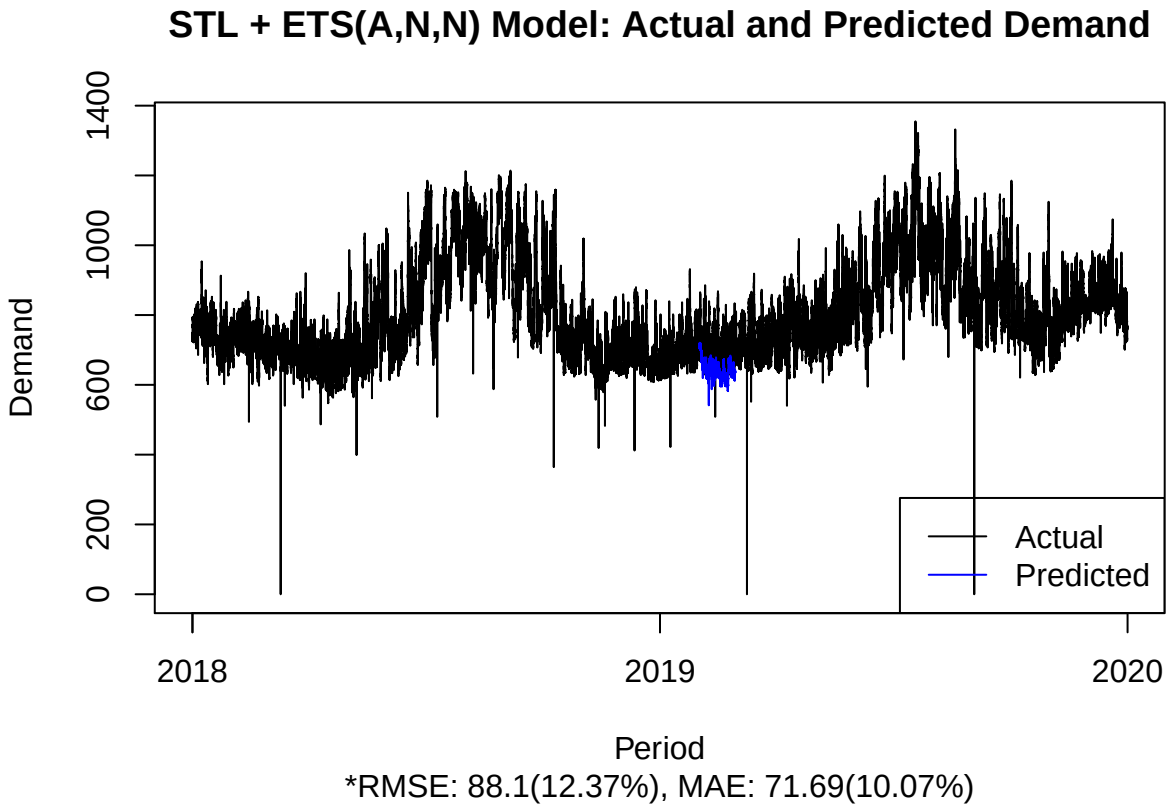
```
##   X       Date period demand year month day           timestamp smoothed
## 1 0 2018-01-01      0 734.40 2018     1   1 2018-01-01 00:00:00   734.40
## 2 1 2018-01-01     15 731.52 2018     1   1 2018-01-01 00:15:00   731.52
## 3 2 2018-01-01     30 727.20 2018     1   1 2018-01-01 00:30:00   727.20
## 4 3 2018-01-01     45 735.84 2018     1   1 2018-01-01 00:45:00   735.84
## 5 4 2018-01-01     60 730.08 2018     1   1 2018-01-01 01:00:00   730.08
## 6 5 2018-01-01     75 728.64 2018     1   1 2018-01-01 01:15:00   728.64
##   forecast
## 1       NA
## 2       NA
## 3       NA
## 4       NA
## 5       NA
## 6       NA
```

Forecast for each month and then update the train data with that month's data for the next month's forecast.

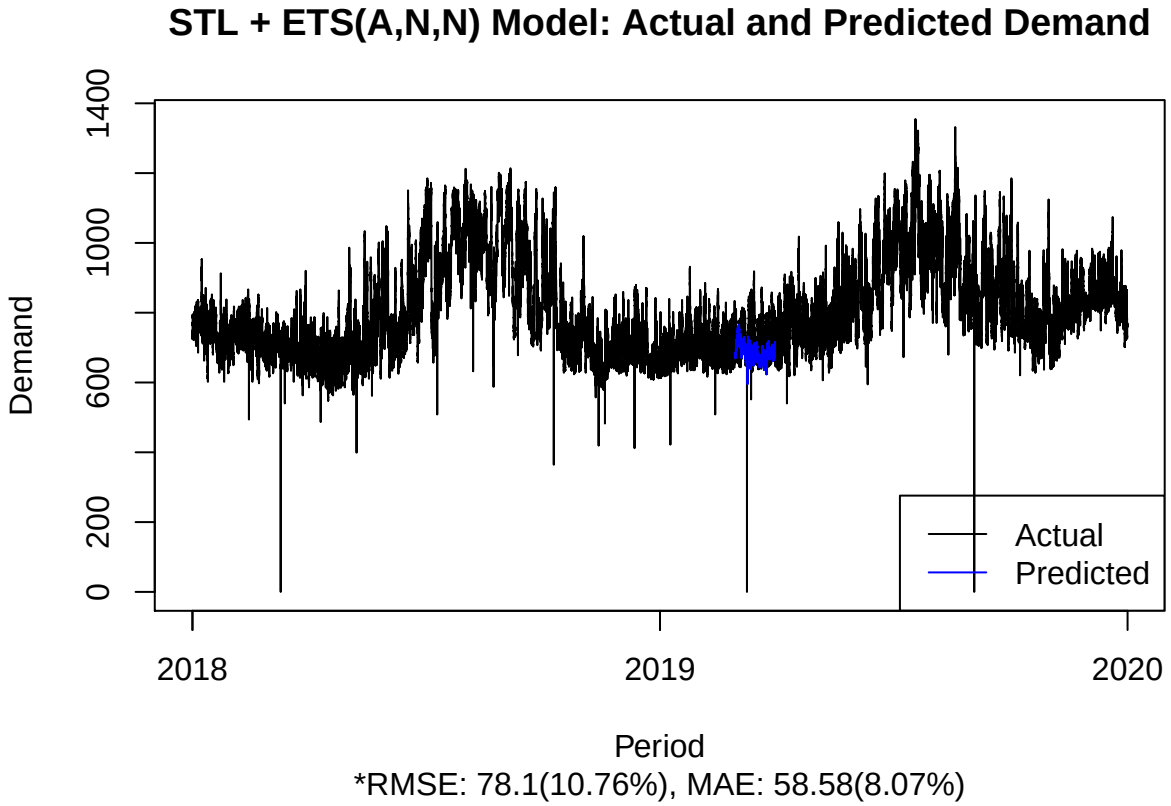
```
res <- c()
for(i in 1:12){
  if(i == 12){
    start_test_period = '2019-12-01 00:00:00'
    end_test_period = '2020-01-01 00:00:00'
  }
  else{
    start_test_period = paste0('2019-',str_pad(i, 2, pad = "0"),'-01 00:00:00')
    end_test_period = paste0('2019-',str_pad(i+1, 2, pad = "0"),'-01 00:00:00')
  }
```

```r
  # df$log_demand <- log(df$demand)
  df_train <- subset(df, df$Date < as.POSIXct(start_test_period) )
  df_test <- subset(df, (df$Date >= as.POSIXct(start_test_period) )& (df$Date < as.POSIXct(end_test_peri
  df_ts <- data.frame(interval = seq(ymd_hms('2018-01-01 00:00:00'),
                                     by = '15 min',length.out=(length(df$demand))),
                      data = df$demand)
  full_ts <- xts(df_ts$data, order.by=df_ts$interval)

  df_ts_train <- data.frame(interval = seq(ymd_hms('2018-01-01 00:00:00'),
                                     by = '15 min',length.out=(length(df_train$demand))),
                      data = df_train$demand)
  df_ts_test <- data.frame(interval = seq(ymd_hms(start_test_period),
                                     by = '15 min',length.out=(length(df_test$demand))),
                      data = df_test$demand)

  freq = 96*30.25
  model <- stlf(ts(df_ts_train$data, frequency = freq), h=length(df_ts_test$data))
  rmse_error <- sqrt( mean( (df_ts_test$data-model$mean)^2 , na.rm = TRUE ) )
  rmse_pct <- round(rmse_error/mean(df_ts_test$data)*100,2)
  mean_abs_error <- sum(abs(df_ts_test$data-model$mean), na.rm = TRUE)/length(df_ts_test$data)
  mae_pct <- round(mean_abs_error/mean(df_ts_test$data)*100,2)
  plot(df_ts$interval, df_ts$data, xlab='Period',ylab='Demand', type='l', main = 'STL + ETS(A,N,N) Model
       sub = paste0('*RMSE: ',round(rmse_error,2), '(',rmse_pct,'%)', ', MAE: ', round(mean_abs_error,2
  lines(df_ts_test$interval, model$mean, col='blue')
  legend(x='bottomright', legend=c('Actual','Predicted'), col=c('black','blue'),lty=1)
  res <- c(res, model$mean)
}
```

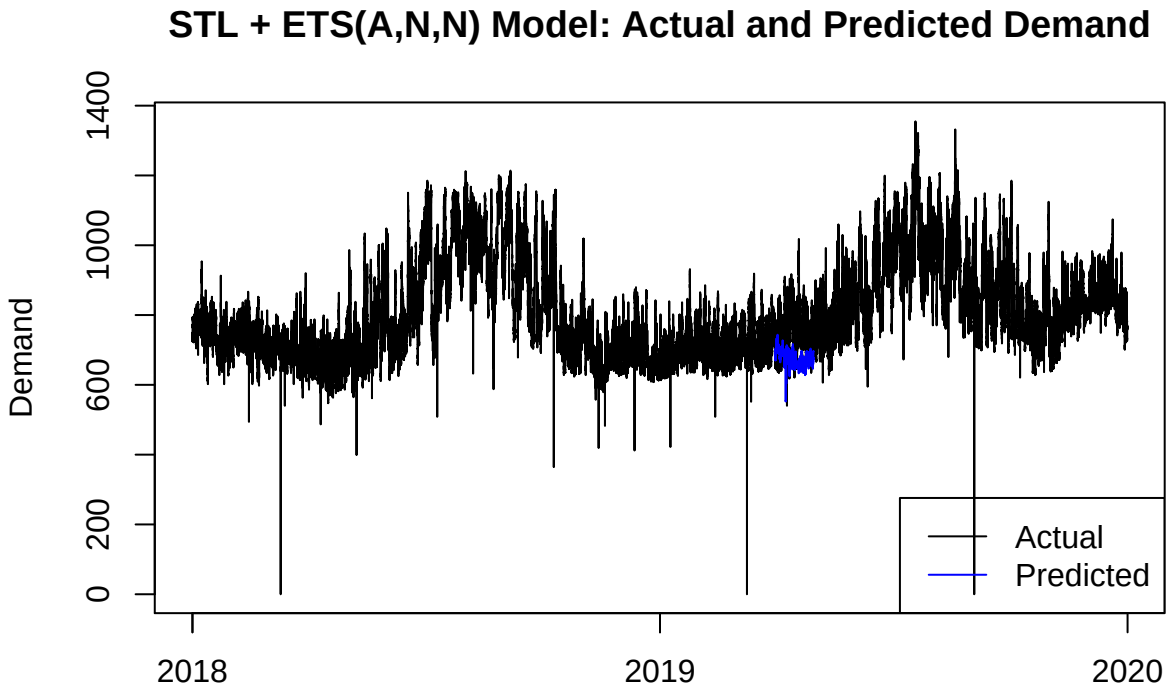




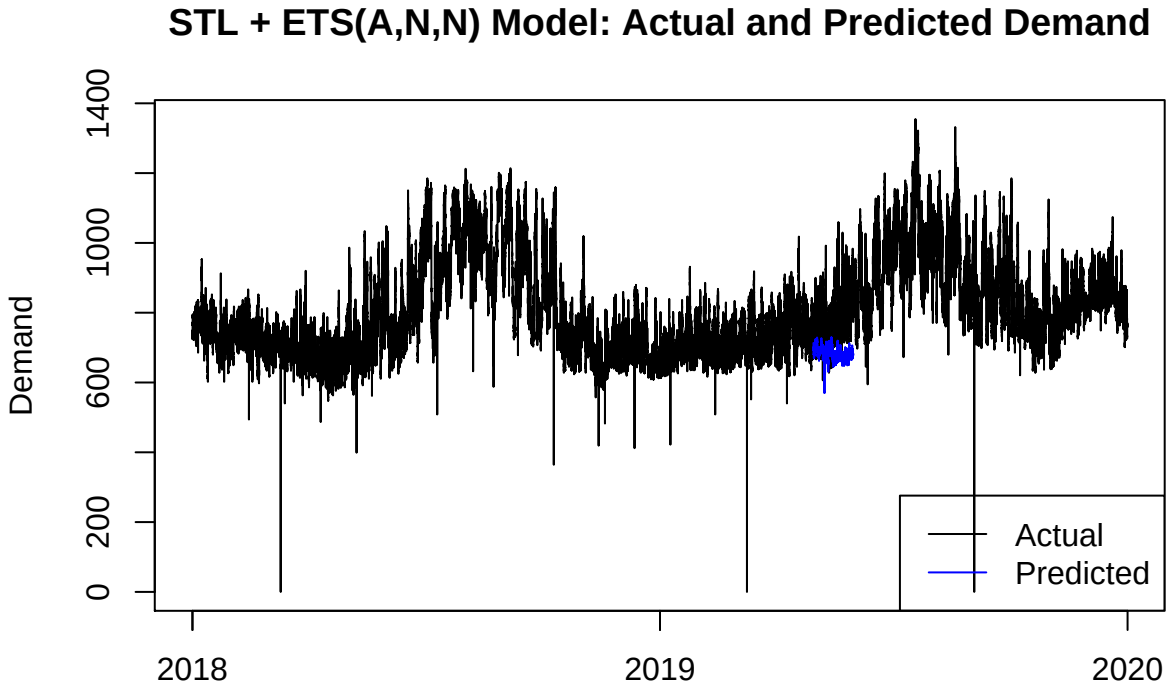
*RMSE: 88.1(12.37%), MAE: 71.69(10.07%)



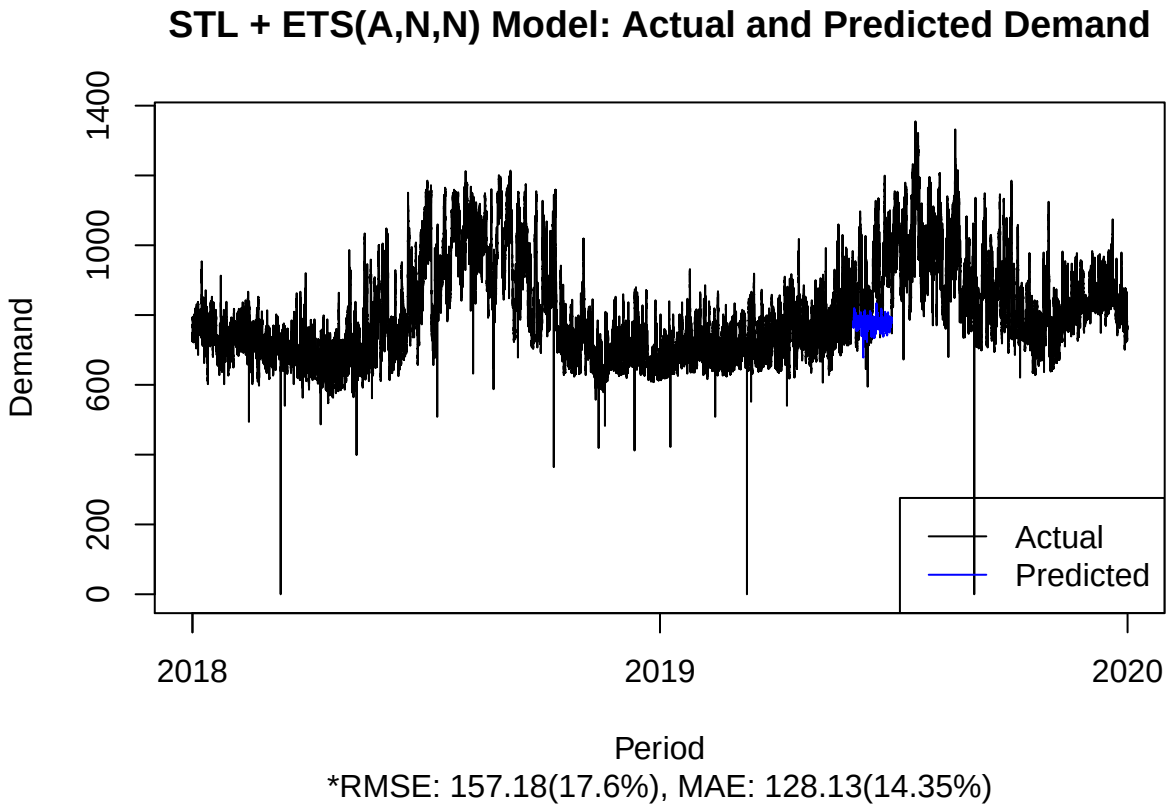
*RMSE: 78.1(10.76%), MAE: 58.58(8.07%)



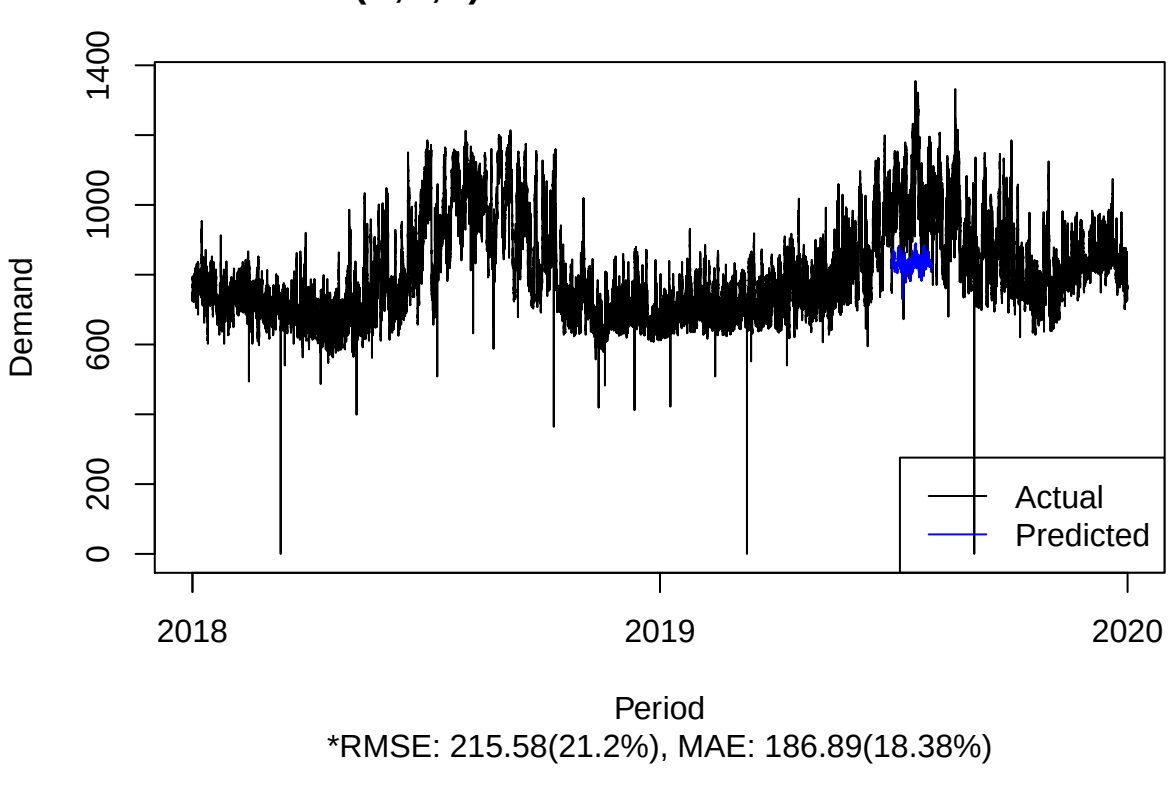
*RMSE: 106.81(14.24%), MAE: 86.14(11.48%)



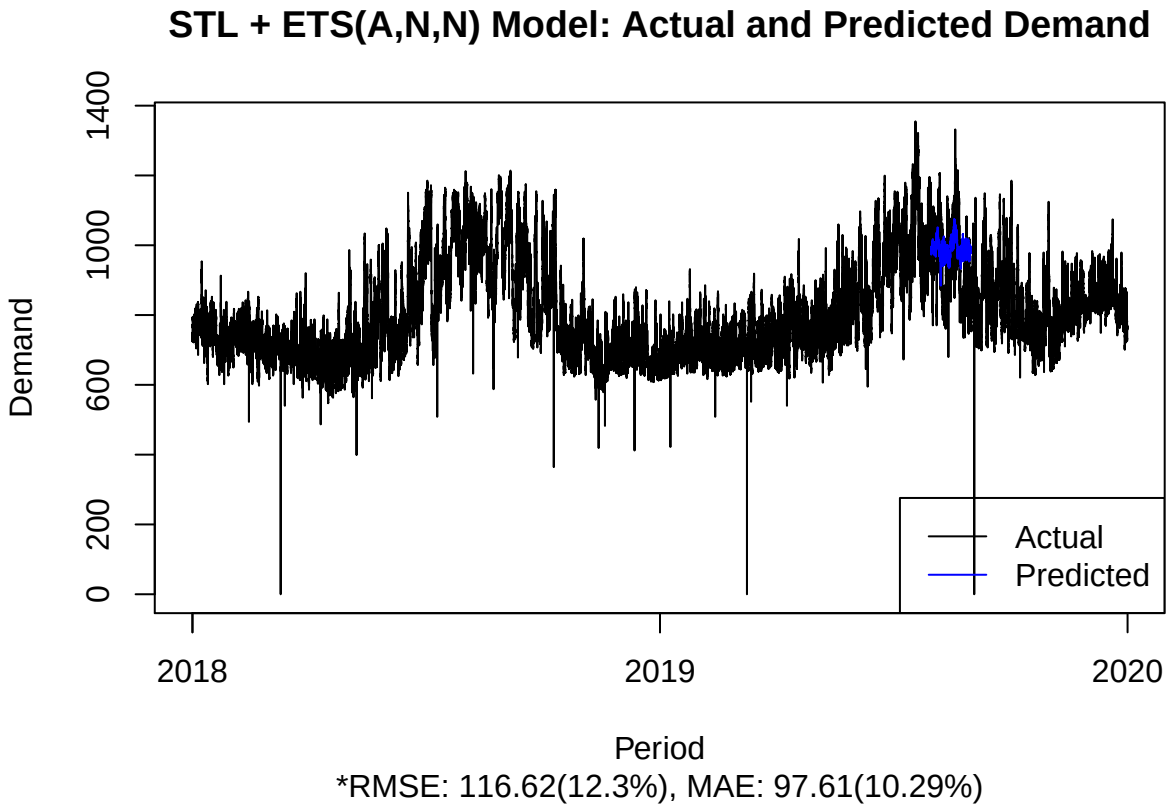
*RMSE: 136.97(17.31%), MAE: 111.99(14.15%)



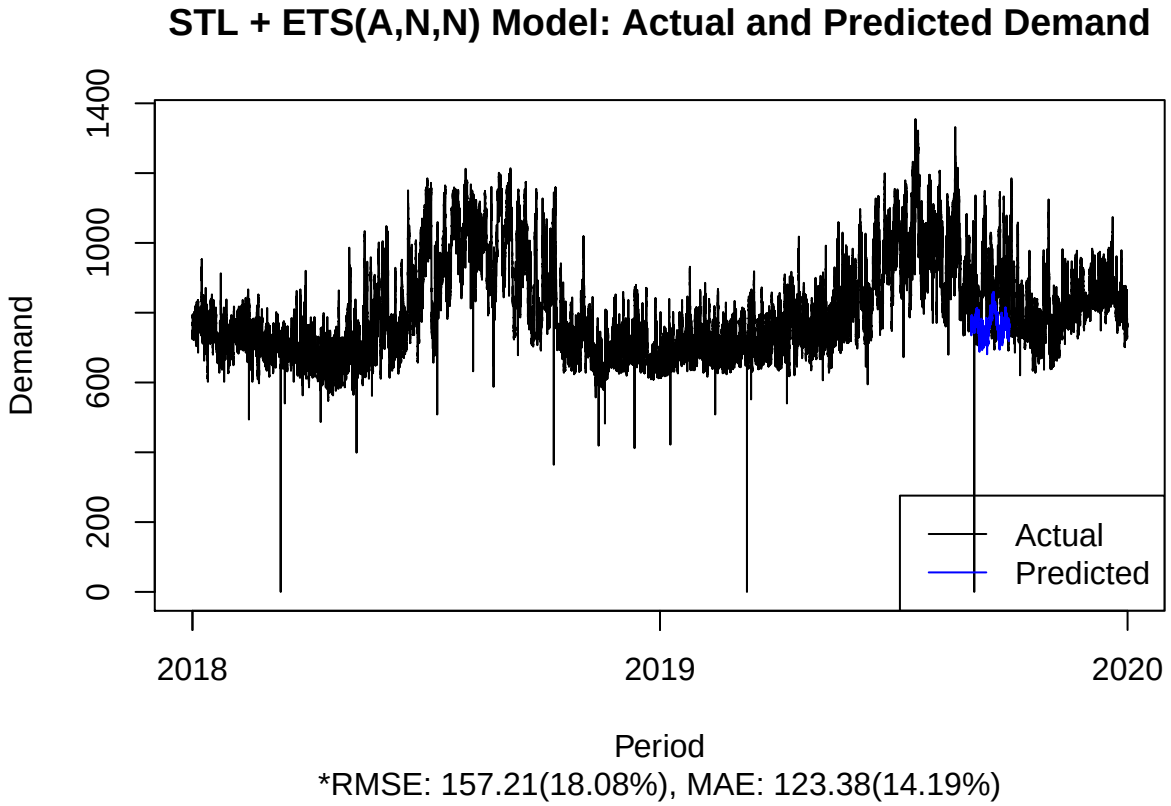
*RMSE: 157.18(17.6%), MAE: 128.13(14.35%)



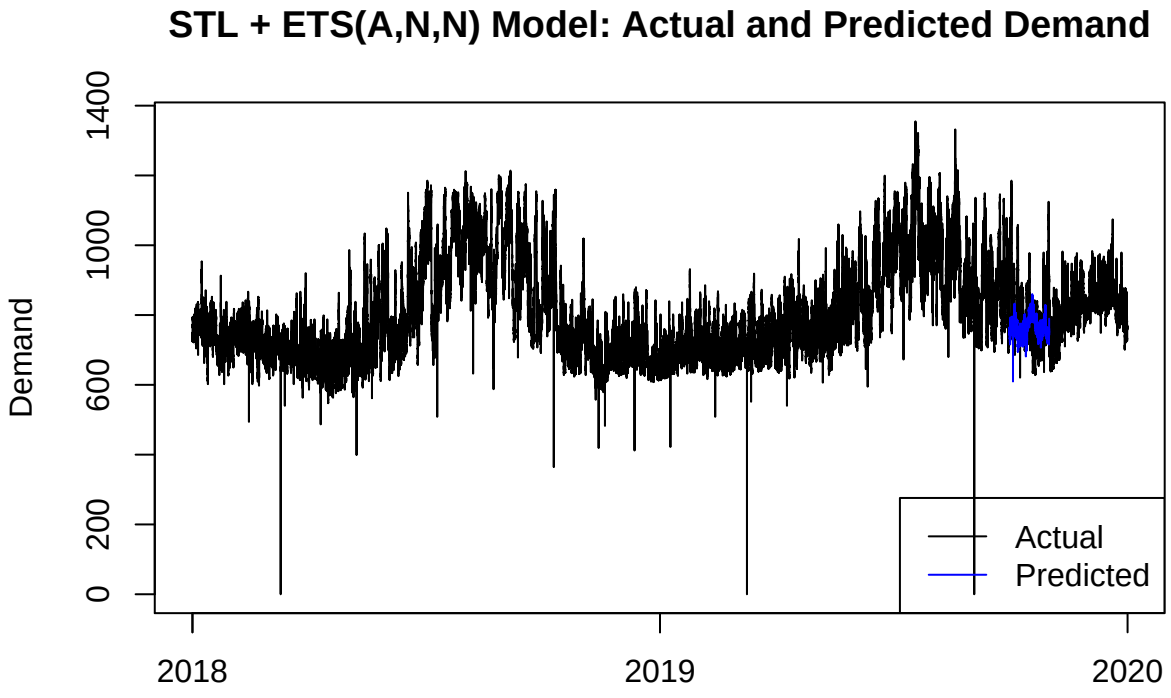
*RMSE: 215.58(21.2%), MAE: 186.89(18.38%)



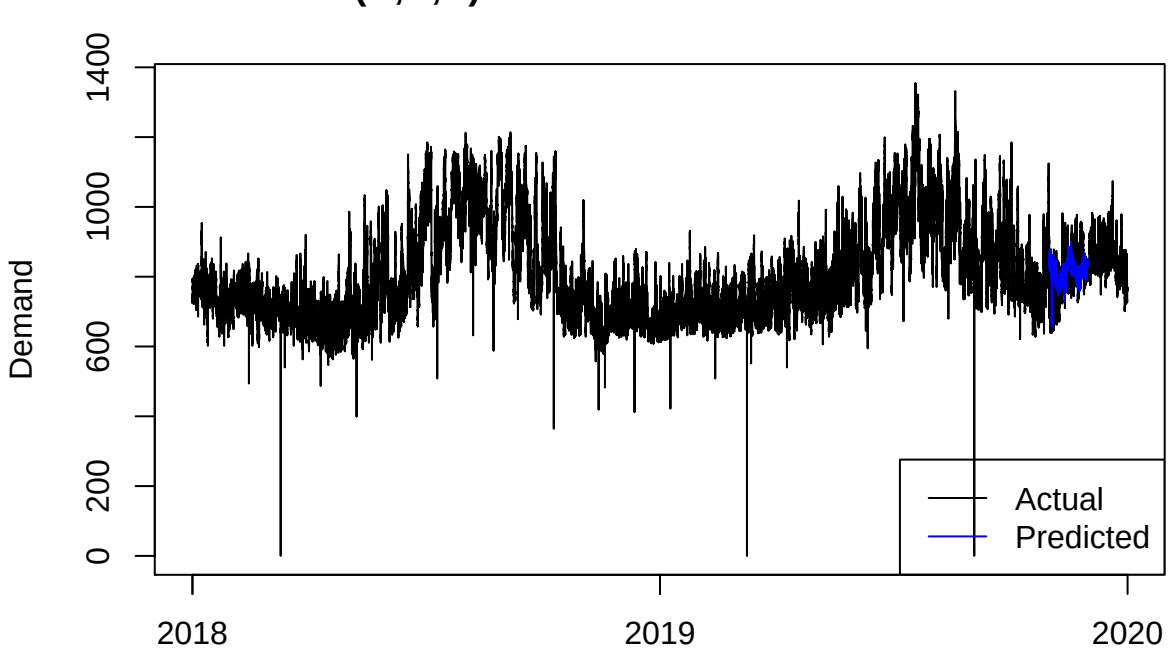
*RMSE: 116.62(12.3%), MAE: 97.61(10.29%)



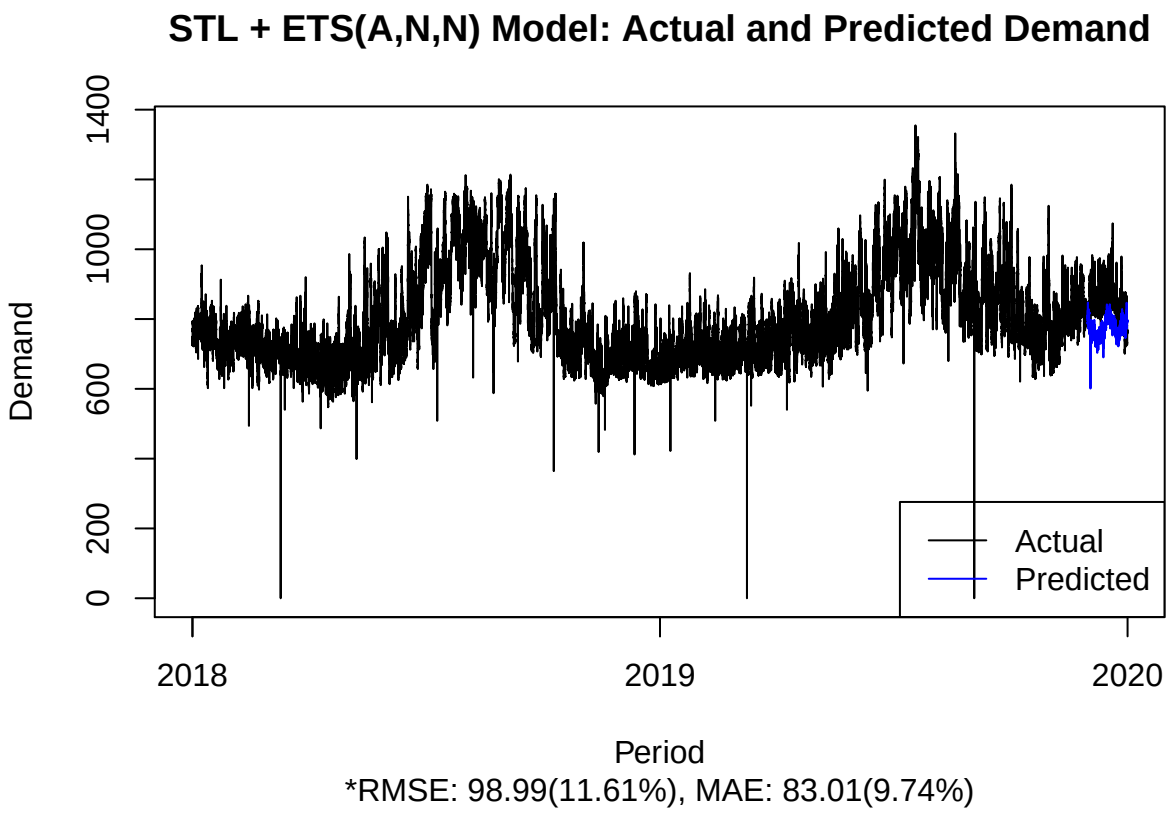
*RMSE: 157.21(18.08%), MAE: 123.38(14.19%)



*RMSE: 113.24(14.14%), MAE: 84.37(10.54%)



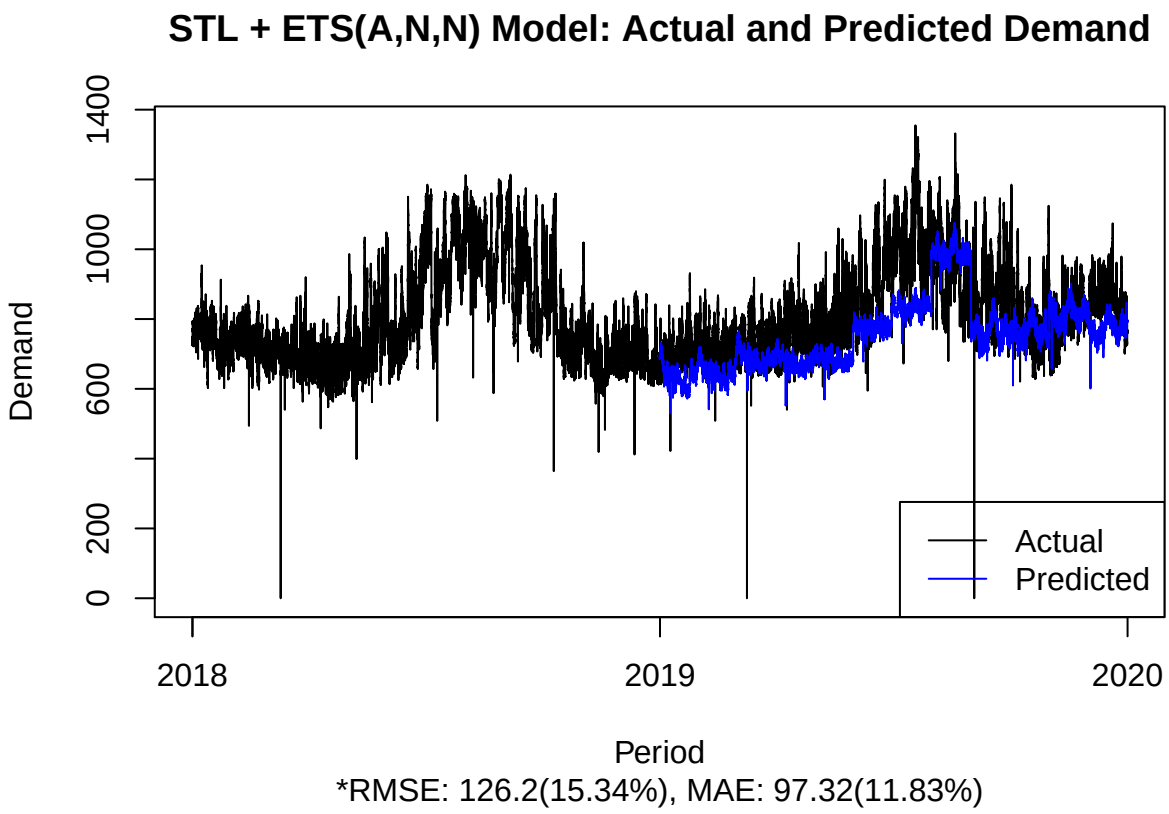
*RMSE: 77.09(9.67%), MAE: 60.8(7.63%)



*RMSE: 98.99(11.61%), MAE: 83.01(9.74%)

```
df_ts['forecast'] <- NA
df_ts[35041:length(df_ts$data),]$forecast <- res
```

```
testing <- df_ts[35041:length(df_ts$data),]
rmse_error <- sqrt( mean( (testing$data-testing$forecast)^2 , na.rm = TRUE ) )
rmse_pct <- round(rmse_error/mean(testing$data)*100,2)
mean_abs_error <- sum(abs(testing$data-testing$forecast), na.rm = TRUE)/length(testing$data)
mae_pct <- round(mean_abs_error/mean(testing$data)*100,2)
plot(df_ts$interval, df_ts$data, col='black', xlab='Period',ylab='Demand', type='l', main = 'STL + ETS(
lines(df_ts$interval, df_ts$forecast, type='l', col='blue')
legend(x='bottomright', legend=c('Actual','Predicted'), col=c('black','blue'),lty=1)
```



*RMSE: 126.2(15.34%), MAE: 97.32(11.83%)

```
# write.csv(df_ts, './data/forecasted_demand.csv')
```