Appendix B

Source code

Excel code

```
A1 = 01/01/2017  00:00:00 ## combine data and time starting at 01/01/2017 00:00
=A2+TIME(0,30,0) ##thirty minute interval

=INDEX(originaldata,1+INT((ROW(B2)-1)/COLUMNS(originaldata)),MOD(ROW(B2)-
1+COLUMNS(originaldata),COLUMNS(originaldata))+1) ##KWh data from rows to column
```

```
## packages requried
library(forecasting)
library(tseries)
library(TTR)
library(forecasting)
library(lubridate)
library(imputeTS)
library(SMA)
library(reshape2)
```

```
original = read.csv("original.csv") ## read the csv file
test = read.csv("2018_month.csv") ## read the csv file. This is the test dataset for testing forecasting model
```

```
original $datetime <-as.POSIXct(original $datetime,format="%Y-%m-%d %H:%M") ## format dataset date + time
```

```
original.ts<- ts(original $KWh, start = decimal_date(as.Date("2016-01-01")), frequency =48) ## create time
series using original thirty minute interval data

plotNA.distribution(original.ts) ##plot missing data
```

```
original.ts <- na_seadec(original.ts, algorithm = "interpolation")## impute missing data
original_df <- data.frame(date=( original$datetime),KWh=as.matrix(original.ts)) ## convert time series to df

daily  <- transform(original_df, datetime = format(original_df$date, format = "%Y-%m-%d"))
daily <- ddply(daily, "datetime", summarize, KWh = sum(KWh)) ## original data to daily data

month  <- transform(original_df, datetime = format(original_df$date, format = "%Y-%m"))
month  <- ddply(month  , "datetime", summarize, KWh = sum(KWh)) ## original data to month data

weekly  <- transform(original_df, datetime = format(original_df$date, format = "%Y,%W"))
weekly  <- ddply(weekly  , "datetime", summarize, KWh = sum(KWh)) ## original data to weekly data

month$datetime <- as.POSIXct(month $datetime, format = "%d/%m/%Y") ## format dataset date
daily $datetime <- as.POSIXct(daily $datetime, format = "%d/%m/%Y") ## format dataset date
weekly $datetime <- as.POSIXct(weekly $datetime, format = "%d/%m/%Y") ## format dataset date

daily.ts<- ts(daily$KWh, start = decimal_date(as.Date("2016-01-01")), frequency =365.25/7) ## create time series using daily data, with daily frequency

weekly.ts<- ts(weekly$KWh, start = decimal_date(as.Date("2016-01-01")), frequency =52) ## create time series using weekly data, with weekly frequency
month.ts<- ts(month$KWh, start = decimal_date(as.Date("2016-01-01")), frequency =12) ## create time series using monthly data, with monthly frequency


test $datetime <-as.POSIXct(test $datetime,format="%Y-%m-%d %H:%M") ## format dataset date + time


test.ts <- na_seadec(test.ts, algorithm = "interpolation")## impute missing data
test_df <- data.frame(date=( test$datetime),KWh=as.matrix(test.ts)) ## convert time series to df

daily_test <- transform(test_df, datetime = format(test_df$date, format = "%Y-%m-%d"))
daily_test <- ddply(daily_test, "datetime", summarize, KWh = sum(KWh)) ## original data to daily data

month_test  <- transform(test _df, datetime = format(test _df$date, format = "%Y-%m"))
month_test  <- ddply(month_test  , "datetime", summarize, KWh = sum(KWh)) ## original data to month data

weekly_test  <- transform(test_df, datetime = format(test_df$date, format = "%Y,%W"))
weekly_test  <- ddply(weekly_test  , "datetime", summarize, KWh = sum(KWh)) ## original data to weekly data


month_test  $datetime <- as.POSIXct(month_test  $datetime, format = "%d/%m/%Y") ## format dataset date
daily_test $datetime <- as.POSIXct(daily_test $datetime, format = "%d/%m/%Y") ## format dataset date
weekly_test  $datetime <- as.POSIXct(weekly_test  $datetime, format = "%d/%m/%Y") ## format dataset date
```

```
week_test<- ts(week_test $KWh, start = decimal_date(as.Date("2018-01-01")), frequency =52) ## create time series using testing weekly data which will be used testing foresting weekly forecasting models
month_test <- ts(month_test $KWh, start = decimal_date(as.Date("2018-01-01")), frequency =12) ) ## create time series using testing monthly data which will be used testing foresting monthly forecasting models



plot(original.ts) ## plot original time series
plot(daily.ts) ## plot daily time series
plot(weekly.ts) ## plot weekly time series
plot(month.ts) ## plot monthly time series


smooth_daily <- SMA(daily_ts, n=7) ## smooth daily time series plot
smooth_monthly <- SMA(month_ts, n=12) ## smooth monthly time series plot
smooth_weekly <- SMA(weekly_ts, n=7) ## smooth weekly time series plot

adf.test(daily.ts) ## seasonality test for daily time series
adf.test(weekly.ts) ## seasonality test for weekly time series
adf.test(monthly.ts) ## seasonality test for monthly time series
weekly_diff <- diff(weekly.ts) ## differencing weekly data as data is not stationary
adf.test(weekly_diff) ## seasonality test for weekly time series


acf (weekly_diff) ## acf for weekly time series
pacf(weekly_diff) ## pacf for weekly time series
acf (daily.ts) ## acf for daily time series
pacf (daily.ts) ## pacf for daily time series
acf (month.ts) ## acf for monthly time series
pacf (month.ts) ## pacf for monthly time series


month_model_1 <- Arima(month.ts, order=c(0,0,3)) ## arima(0,0,3) model for monthly forecasting

checkresiduals(month_model_1) ## checking arima(0,0,3) residuals

tsdiag(month_model_1) ## arima(0,0,3) ljung-Box-Pierce plot

pacf (month_model_1 $residuals) ## arima(0,0,3) residuals pacf

month_model_1 _forecast <- month_model1 %>% forecast(h=3) +
```

```
autoplot() + autolayer(month_test) +   labs(y="KWh", title = "Monthly time series forecast from Arima (0,0,3)")
## arima(0,0,3) forecasting compared with test dataset

accuracy(month_model_1 _forecast, x= month_test)## compare model performance with test data


month_model_2 <- Arima(month.ts, order=c(0,0,2)) ## arima(0,0,2) model for monthly forecasting

checkresiduals(month_model_2) ## checking arima(0,0,2) residuals


tsdiag(month_model_2) ## arima(0,0,2) ljung-Box-Pierce plot

pacf (month_model_2 $residuals) ## arima(0,0,2) residuals pacf


month_model_2 _forecast <- month_model2 %>% forecast(h=3) +
autoplot() + autolayer(month_test) +   labs(y="KWh", title = "Monthly time series forecast from Arima (0,0,2)")
## arima(0,0,2) forecasting compared with test dataset

accuracy(month_model_2 _forecast, x= month_test)## compare model performance with test data


month_model_3 <- auto.arima(month.ts) ## auto arima model for monthly forecasting

checkresiduals(month_model_3) ## checking auto arima residuals

tsdiag(month_model_3) ## auto arima ljung-Box-Pierce plot

pacf (month_model_3 $residuals) ## auto arima residuals pacf

month_model_3 _forecast <- month_model3 %>% forecast(h=3) +
autoplot() + autolayer(month_test) +   labs(y="KWh", title = "Monthly time series forecast from Arima (1,0,0)")
## auto arima forecasting compared with test dataset

accuracy(month_model_3 _forecast, x= month_test)## compare model performance with test data




week_model_1 <- Arima(weekly.ts, order=c(0,1,1)) ## arima(0,1,1) model for weekly forecasting
```

```
checkresiduals(week _model_1) ## arima(0,1,1) residuals

tsdiag(week _model_1) ## arima(0,1,1) ljung-Box-Pierce plot

pacf (week _model_1 $residuals) ## arima(0,1,1) residuals pacf

week _model_1 _forecast <- week _model1 %>% forecast(h=12) +
autoplot() + autolayer(week_test) +   labs(y="KWh", title = "Weekly time series forecast from Arima (0,1,1)")
## arima(0,1,1) forecasting compared with test dataset

accuracy(week_model_1 _forecast, x= week_test)## compare model performance with test data



week_model_2 <- Arima(weekly.ts, order=c(0,1,2)) ## arima(0,1,2) model for weekly forecasting

checkresiduals(week _model_2) ## arima(0,1,2) residuals

tsdiag(week _model_2) ## arima(0,1,2) ljung-Box-Pierce plot

pacf (week _model_2 $residuals) ## arima(0,1,2) residuals pacf

week _model_2 _forecast <- week _model2 %>% forecast(h=12) +
autoplot() + autolayer(week_test) +   labs(y="KWh", title = "Weekly time series forecast from Arima (0,1,2)") ##
arima(0,1,2) forecasting compared with test dataset

accuracy(week_model_2 _forecast, x= week_test)## compare model performance with test data



week_model_3 <- auto.arima(weekly.ts) ## auto arima model for weekly forecasting

checkresiduals(week_model_3) ## checking auto arima residuals

tsdiag(week_model_3) ## auto arima ljung-Box-Pierce plot

pacf (week_model_3 $residuals) ## auto arima residuals pacf

month_model_3 _forecast <- month_model3 %>% forecast(h=12) +
autoplot() + autolayer(week_test) +   labs(y="KWh", title = "Monthly time series forecast from Arima (1,1,0)") ##
auto arima forecasting compared with test dataset
accuracy(week_model_3 _forecast, x= week_test)## compare model performance with test data
```

```
daily_model_1 <- Arima(daily.ts, order=c(2,0,2)) ## arima(2,0,2) model for daily forecasting

checkresiduals(daily _model_1) ## arima(2,0,2) residuals

tsdiag(daily _model_1) ## arima(2,0,2) ljung-Box-Pierce plot

pacf (daily _model_1 $residuals) ## arima(2,0,2) residuals pacf

daily _model_1 _forecast <- daily _model1 %>% forecast() +
autoplot() +   labs(y="KWh", title = "Daily time series forecast from Arima (2,0,2)") )") ## arima(2,0,2) forecasting
compared with test dataset

daily_model_1 <- Arima(daily.ts, order=c(2,0,2))  %>% summary () ## model performance


daily_model_2 <- auto.arima(daily.ts) ## auto arima model for daily forecasting

checkresiduals(daily_model_2) ## auto arima residuals

tsdiag(daily_model_2) ## auto arima ljung-Box-Pierce plot

pacf (daily_model_2$residuals) ## auto arima residuals pacf

daily _model_2 _forecast <- daily _model2 %>% forecast() +
autoplot() +   labs(y="KWh", title = "Daily time series forecast from Arima (3,0,3)") )") ## auto arima forecasting

daily_model_2 <- auto.arima(daily.ts) ## model performance


## arima (0,0,3) monthly forecasting model outliers
fit <- Arima(month.ts, order=c(0,0,3))
upper <- fitted(fit) + 1.96*sqrt(fit$sigma2)
lower <- fitted(fit) - 1.96*sqrt(fit$sigma2)
plot(month.ts, type="n", ylim=range(lower,upper)) + title("Arima (0,0,3) outliers")
polygon(c(time(month.ts),rev(time(month.ts))), c(upper,rev(lower)),
col=rgb(0,0,0.6,0.2), border=FALSE)
lines(month.ts)
lines(fitted(fit),col='red')
out <- (month.ts < lower | month.ts > upper)
points(time(month.ts)[out], month.ts [out], pch=19)
```

```
## arima (0,0,2) monthly forecasting model outliers
fit1 <- Arima(month.ts, order=c(0,0,2))
upper <- fitted(fit1) + 1.96*sqrt(fit1$sigma2)
lower <- fitted(fit1) - 1.96*sqrt(fit1$sigma2)
plot(month.ts, type="n", ylim=range(lower,upper)) + title("Arima (0,0,2) outliers")
polygon(c(time(month.ts),rev(time(month.ts))), c(upper,rev(lower)),
col=rgb(0,0,0.6,0.2), border=FALSE)
lines(month.ts)
lines(fitted(fit1),col='red')
out <- (month.ts < lower | month.ts > upper)
points(time(month.ts)[out], month.ts [out], pch=19)

## arima (1,0,0) monthly forecasting model outliers
fit2 <- Arima(month.ts, order=c(1,0,0))
upper <- fitted(fit2) + 1.96*sqrt(fit2$sigma2)
lower <- fitted(fit2) - 1.96*sqrt(fit2$sigma2)
plot(month.ts, type="n", ylim=range(lower,upper)) + title("Arima (1,0,0) outliers")
polygon(c(time(month.ts),rev(time(month.ts))), c(upper,rev(lower)),
col=rgb(0,0,0.6,0.2), border=FALSE)
lines(month.ts)
lines(fitted(fit2),col='red')
out <- (month.ts < lower | month.ts > upper)
points(time(month.ts)[out], month.ts [out], pch=19)

## arima (0,1,1) weekly forecasting model outliers
fit3<- Arima(weekly.ts, order=c(0,1,1))
upper <- fitted(fit3) + 1.96*sqrt(fit3$sigma2)
lower <- fitted(fit3) - 1.96*sqrt(fit3$sigma2)
plot(weekly.ts, type="n", ylim=range(lower,upper)) + title("Arima (0,1,1) outliers")
polygon(c(time(weekly.ts),rev(time(weekly.ts))), c(upper,rev(lower)),
col=rgb(0,0,0.6,0.2), border=FALSE)
lines(weekly.ts)
lines(fitted(fit3),col='red')
out <- (weekly.ts < lower | weekly.ts > upper)
points(time(weekly.ts)[out], weekly.ts [out], pch=19)



## arima (0,1,2) weekly forecasting model outliers
fit4<- Arima(weekly.ts, order=c(0,1,2))
upper <- fitted(fit4) + 1.96*sqrt(fit4$sigma2)
lower <- fitted(fit4) - 1.96*sqrt(fit4$sigma2)
plot(weekly.ts, type="n", ylim=range(lower,upper)) + title("Arima (0,1,2) outliers")
```

```
polygon(c(time(weekly.ts),rev(time(weekly.ts))), c(upper,rev(lower)),
col=rgb(0,0,0.6,0.2), border=FALSE)
lines(weekly.ts)
lines(fitted(fit4),col='red')
out <- (weekly.ts < lower | weekly.ts > upper)
points(time(weekly.ts)[out], weekly.ts [out], pch=19)

## arima (1,1,0) weekly forecasting model outliers
fit5<- Arima(weekly.ts, order=c(1,1,0))
upper <- fitted(fit5) + 1.96*sqrt(fit5$sigma2)
lower <- fitted(fit5) - 1.96*sqrt(fit5$sigma2)
plot(weekly.ts, type="n", ylim=range(lower,upper)) + title("Arima (1,1,0) outliers")
polygon(c(time(weekly.ts),rev(time(weekly.ts))), c(upper,rev(lower)),
col=rgb(0,0,0.6,0.2), border=FALSE)
lines(weekly.ts)
lines(fitted(fit5),col='red')
out <- (weekly.ts < lower | weekly.ts > upper)
points(time(weekly.ts)[out], weekly.ts [out], pch=19)
```