Oncor Electicity Delivery Project

Ives

4/3/2020

Recent studies showed that Electric Vehicles(EV) charging has a large impact on electricity delivery especially when many users charge EV simultaneously. Therefore, EV charging monitoring has become a more important and urgent missing piece in electricity delivery.This project is aimed to identify which electricity users have Electric Vehicles (EV) based on electricity meter data of 6 million records.

For this project, I used Non-Intrusive Load Monitoring(NILM) model to transfer the long electricity signal to several peaks.Then we will have several useful variables including: the duration of peaks, the hight of the peaks and the frequency of peaks. Finally, we identify which kinds of peaks are more likely to be EV users pattern based on some practical assumptions. The accuracy rate of this combined model is **87.5%** in a test dataset.

1. Introduction to data and data preprocessing

```
#load
data <- read.csv("C:/Study/Capstone/Data/EV_Team_Sample_5.csv")
#change the colom name
colnames(data)[1]="id"
#process null values
data[which(is.na(data$consumption)),3] <- 0
#change the type of time
data$read_date <- as_datetime(data$read_date)
#slice the data
d1 <- data
#extract all the years and months
d1 <- d1 %>%
  mutate(year=year(d1$read_date),month=month(d1$read_date))
#get the time metrix
y_m=d1%>%
  select(year,month)%>%
  distinct()
#show the data
str(data)
```

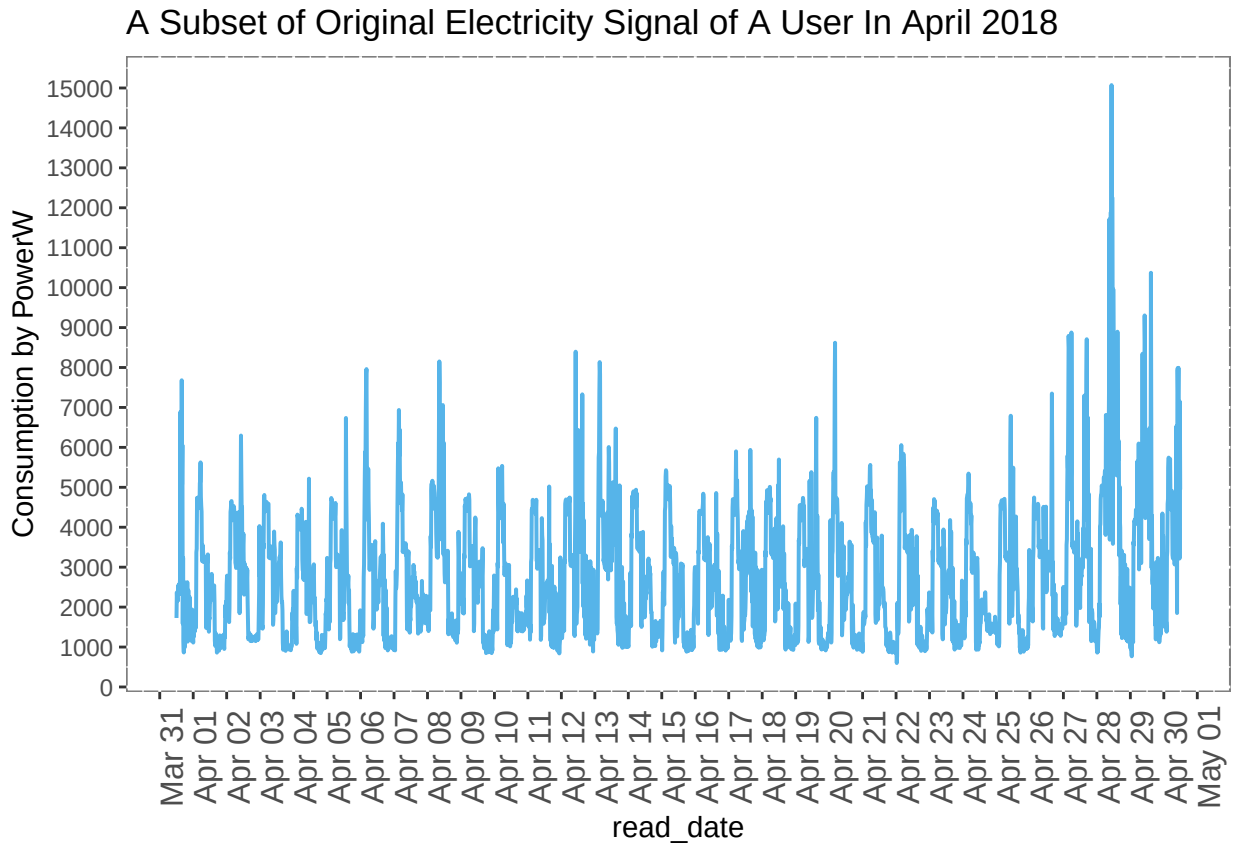
```
## 'data.frame':    6313866 obs. of  3 variables:
##  $ id         : Factor w/ 100 levels "RDM0077172","RDM0112455",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ read_date  : POSIXct, format: "2018-04-01 00:00:00" "2018-04-01 00:15:00" ...
##  $ consumption: num  0.942 0.538 0.588 0.542 1.254 ...
```

- We have 3 coloums but 6 million observations.
  - The id is the only key to identify each user anonymously.In this training dataset, we have 100 users.
  - The read_date means when the data was collected. This dataset contains 2 years data collected by every 15 minute
  - The consumption is the electricity usage of every 15 mintues by the unit of KWh. We will transfer it to average electricity power for furthur analysis.

2. Non-Intrusive Load Monitoring(NILM) model

In this model, we cut the long signal into many subsets by id, year and month. Then we use NILM model to disaggregate the signal to multiple variables dataset. Therefore, to build a NILM model, we test it with a sample subset of the first user's data of 2018 April.

```
#select a dataset
power_data=subset(d1,d1$id=='RDM2377306'&d1$year==2018&d1$month==4)
#show it in a graph
ggplot(power_data[1:3000,],aes(x=read_date,y=consumption*4000,group=1)) %>%
  +geom_line(color="#56B4E9",size=0.7) %>%
  +scale_x_datetime(date_labels = "%b %d",breaks=("24 hours"))%>%
  +scale_y_continuous(breaks=seq(0,18000,1000)) %>%
  +theme(axis.text.x = element_text(angle=90, size=12)) %>%
  +ylab("Consumption by PowerW")%>%
  +theme(panel.background = element_rect(fill = "white", colour = "grey50")) %>%
  +ggtitle("A Subset of Original Electricity Signal of A User In April 2018")
```



We can also look at three days data to better understand a usr's pattern.

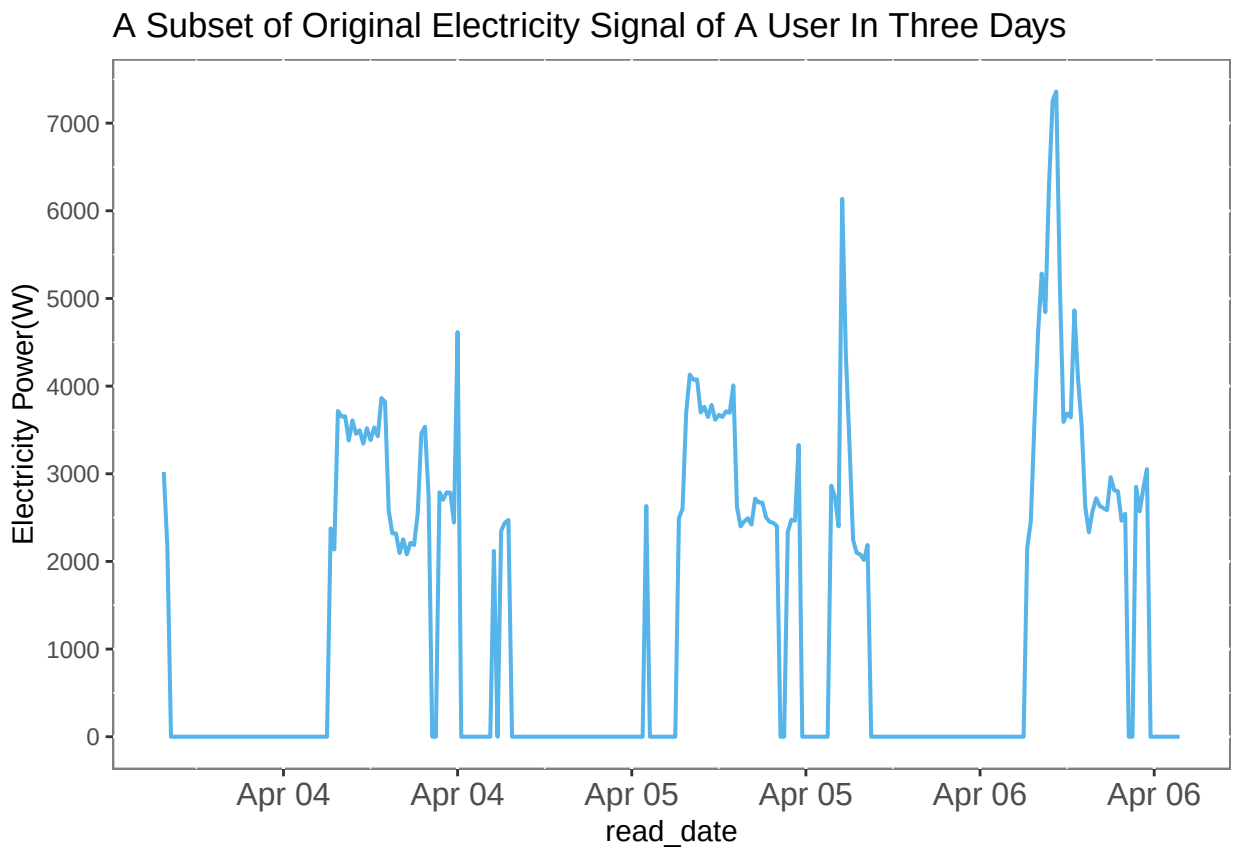
We can see there are many peaks which indicate tense electricity usage probably caused by AC,evener and EV.This NILM model will identify these peaks to filter all the other useless information for furthur detecting.

STEP 1 Get the residual and remove it

```
# the first column should be time and the second column should be the power data
org_time=power_data$read_date
orgAgg=power_data$consumption*4000
```

```r
# the first column should be time and the second column should be the power data
org_time=power_data$read_date
orgAgg=power_data$consumption*4000
res=min(orgAgg)
ts=orgAgg-res
#set threshold value
Threshold=Threshold_value
EVsignal = ts;
EVsignal[EVsignal<Threshold]=0

#draw a graphic
power_data$consumption=EVsignal
#show it in a graph
ggplot(power_data[300:580,],aes(x=read_date,y=consumption,group=1)) %>%
  +geom_line(color="#56B4E9",size=0.71) %>%
  +scale_x_datetime(date_labels = "%b %d",breaks=("12 hours"))%>%
  +scale_y_continuous(breaks=seq(0,18000,1000)) %>%
  +theme(axis.text.x = element_text(angle=0, size=12)) %>%
  +ylab("Electricity Power(W)")%>%
  +theme(panel.background = element_rect(fill = "white", colour = "grey50")) %>%
    +ggtitle("A Subset of Original Electricity Signal of A User In Three Days")
```



STEP 2 Get segments

From last step, we have the peaks data and right now we need to know the start point and end point of these peaks to analyze. So I define a function getSegment() to finish this task.

```r
# Define getSegment function, return segment
## the first row is N/A, if there is no segment, there will only be one row
getSegment <- function(EVsignal){
  idx=as.numeric(EVsignal>0)
  idx2=c(0,idx,0)
  prePt = c("")
  postPt = c("")

  #get the segment starts and ends
  for (i in 2:length(idx)){
    if (idx2[i-1] == 0 & idx2[i] == 1 & idx2[i+1] == 1){
      prePt=c(prePt,i-1)
    }
    if (idx2[i-1] == 1 & idx2[i] == 1 & idx2[i+1] == 0){
      postPt=c(postPt,i-1)
    }
  }
  #if the segment extend this month signal, we set the last index as the postPt
  if(length(prePt)==length(postPt)+1){
    postPt=c(postPt,length(idx))
  }
  segment=tibble(prePt,postPt)
  return (segment)
}

segment=getSegment(EVsignal)
segment <- as.data.frame(lapply(segment, function(x) as.numeric(as.character(x))))
```

In this sample dataset, our peaks are showed below:

Table 1: Table with peaks

	prePt	postPt
10	239	242
11	250	277
12	280	283
13	291	301
14	346	373
15	376	381

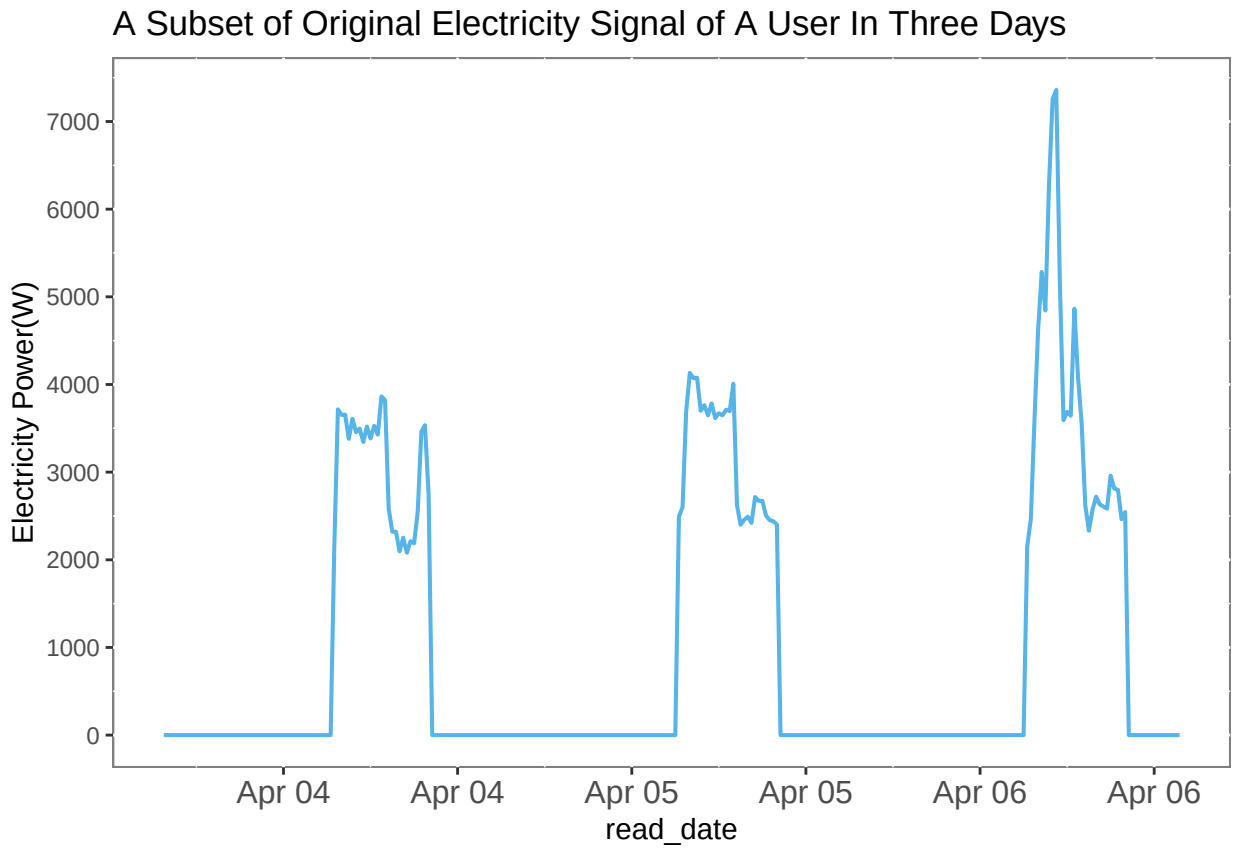
Step 3 Filter segments based on charging duration

```r
#set duration range
min_Duration = min_duration
max_Duration = max_duration

#calculate the duraton
segment= segment %>% mutate(duration=interval*(postPt-prePt))
#filter segments
Dur_segment = filter(segment,duration>min_duration&duration<max_duration)

#draw a graphic
```

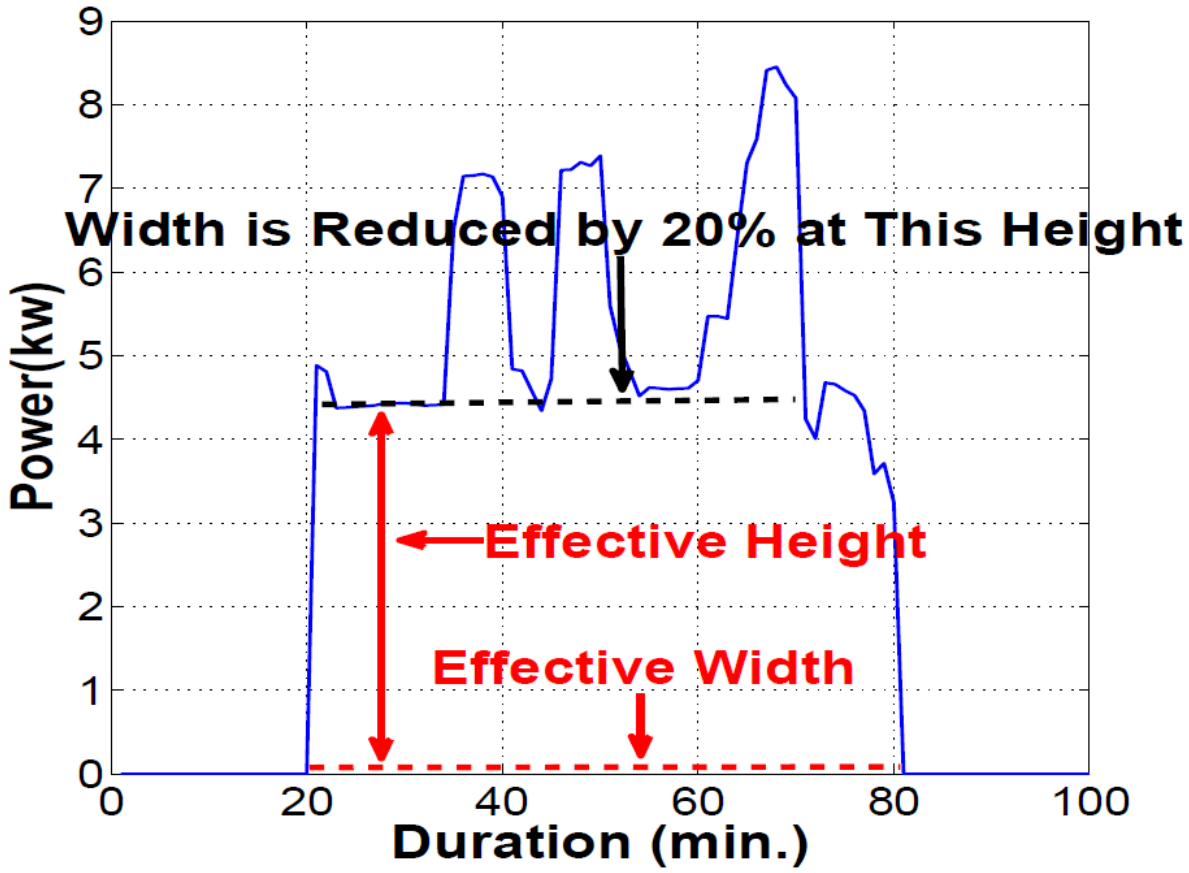
```
EVsignal[300:346]=0
EVsignal[374:441]=0
EVsignal[470:537]=0
EVsignal[566:580]=0
power_data$consumption=EVsignal
#show it in a graph
ggplot(power_data[300:580,],aes(x=read_date,y=consumption,group=1)) %>%
  +geom_line(color="#56B4E9",size=0.71) %>%
  +scale_x_datetime(date_labels = "%b %d",breaks=("12 hours"))%>%
  +scale_y_continuous(breaks=seq(0,18000,1000)) %>%
  +theme(axis.text.x = element_text(angle=0, size=12)) %>%
  +ylab("Electricity Power(W)")%>%
  +theme(panel.background = element_rect(fill = "white", colour = "grey50")) %>%
    +ggtitle("A Subset of Original Electricity Signal of A User In Three Days")
```



We have three peaks left after filtering by duration.

STEP 4 Calculate the height and the width of each peaks

The height means the electric power of a peak and the width means duration time. These two features are very useful for EV signals to be distinguished from AC's and evener's.



In this NILM model, the effective width is defined as the width of a segment at bottom. The effective height is defined as the height at which the segment's width becomes only 80% of the bottom width.

```
#Define getHeight function
calHeight = function(signal,width_percent){
  for (i in 1:300){
    height=i/300*max(signal)
    if (sum(as.numeric(signal>=height))/length(signal)<width_percent){
      return(height)
    }
  }
}
getHeight = function(segment,signal){
  height=c()
  for (i in 1:nrow(segment)){
    height=c(height,calHeight(signal[segment[i,1]:segment[i,2]],width_percent))
  }
  return(mutate(segment,estPower=height))
}

Height_segment=getHeight(Dur_segment,EVsignal)

#Filter segment which height is below EVAMP
EV_segment=Height_segment %>% filter(estPower>3000)
names(EV_segment)=c("Start Point","End Point","Charging Time(min)","Electricity Power")
```

Table 2: Table with peaks

Start Point	End Point	Charging Time(min)	Electricity Power
1152	1165	195	3431.280
1978	2005	405	3157.867
2074	2101	405	3018.987
2650	2708	870	4245.120
2746	2800	810	3517.920

Therefore, this results shows that this user might charge his/her EV 5 times a month. So, this is the process we did to one subdataset. In my real project, I combined all these steps into a function file and the next step is to apply this model to all the 2400 subdatasets.

5.Apply NILM Model to all the subdatasets

```
#Group the data by year and month and then get the result
result=group_by(d1,id,year,month)%>% do(as.data.frame(estEV(.)))

#remove na in variance
result$Duration.Variance[is.na(result$Duration.Variance)]=0
result$estPower.Variance[is.na(result$estPower.Variance)]=0

#make the last month for every user to be 0
for(i in 2:nrow(result)){
  if(result[i,1]!=result[i-1,1]){
    result[i,4]=0
  }
}
```

The results show below:

```
#change the name
names(result)=c("id","Year","Month","EV_user_flag","Singals_Num","Duration.Variance","mean.estPower",
#select coloums
result1=result[c("id","Year","Month","Singals_Num","mean.estPower","estPower.Variance")]
#show the result
kable(result1[1:50,],caption = "EV Charging Signals Statistics Group By Users Every Month")
```

Table 3: EV Charging Signals Statistics Group By Users Every Month

id	Year	Month	Singals_Num	mean.estPower	estPower.Variance
RDM0077172	2018	4	2	3640.493	187.5247
RDM0077172	2018	5	5	3939.792	156.3307
RDM0077172	2018	6	11	4253.756	547.6242
RDM0077172	2018	7	11	4040.558	323.6309
RDM0077172	2018	8	12	3984.537	350.8866
RDM0077172	2018	9	1	3508.800	0.0000
RDM0077172	2018	10	6	4051.276	585.0311
RDM0077172	2018	11	7	4418.730	884.9131
RDM0077172	2018	12	3	4018.347	288.2629
RDM0077172	2019	1	6	4511.607	1385.9332
RDM0077172	2019	2	3	3811.409	126.0245

id	Year	Month	Singals_Num	mean.estPower	estPower.Variance
RDM0077172	2019	3	0	0.000	0.0000
RDM0077172	2019	4	0	0.000	0.0000
RDM0077172	2019	5	1	3579.440	0.0000
RDM0077172	2019	6	5	3969.112	124.4939
RDM0077172	2019	7	9	3897.273	246.4836
RDM0077172	2019	8	17	3924.510	196.5373
RDM0077172	2019	9	2	4130.293	225.8593
RDM0077172	2019	10	1	3870.613	0.0000
RDM0077172	2019	11	6	4447.167	1506.5483
RDM0077172	2019	12	5	4355.971	817.4145
RDM0077172	2020	1	7	3906.196	393.0146
RDM0077172	2020	2	9	4335.074	666.0438
RDM0077172	2020	3	3	3864.129	306.3388
RDM0112455	2018	4	0	0.000	0.0000
RDM0112455	2018	5	0	0.000	0.0000
RDM0112455	2018	6	0	0.000	0.0000
RDM0112455	2018	7	0	0.000	0.0000
RDM0112455	2018	8	0	0.000	0.0000
RDM0112455	2018	9	0	0.000	0.0000
RDM0112455	2018	10	0	0.000	0.0000
RDM0112455	2018	11	0	0.000	0.0000
RDM0112455	2018	12	0	0.000	0.0000
RDM0112455	2019	1	0	0.000	0.0000
RDM0112455	2019	2	0	0.000	0.0000
RDM0112455	2019	3	0	0.000	0.0000
RDM0112455	2019	4	0	0.000	0.0000
RDM0112455	2019	5	0	0.000	0.0000
RDM0112455	2019	6	0	0.000	0.0000
RDM0112455	2019	7	0	0.000	0.0000
RDM0112455	2019	8	0	0.000	0.0000
RDM0112455	2019	10	0	0.000	0.0000
RDM0112455	2019	11	0	0.000	0.0000
RDM0112455	2019	12	0	0.000	0.0000
RDM0112455	2020	1	0	0.000	0.0000
RDM0112455	2020	2	0	0.000	0.0000
RDM0112455	2020	3	0	0.000	0.0000
RDM0145005	2018	4	0	0.000	0.0000
RDM0145005	2018	5	0	0.000	0.0000
RDM0145005	2018	6	0	0.000	0.0000

The mean.estPower means the mean value of all the electric power signals estimated value in a month. This value should be in the range of EV electric power. We can also use this range to identify different types of EV.For examole, Tesla model S has an eletric power range from 15Kw to 18 Kw, while the BMW electric cars mainly stay from 4 Kw to 6 Kw.

The estPower.Variance means the variance of all the electric power signals estimated value in a month.This value should be as small as possible. If the variance is small enough, it is more likely to say that this household use a certain type of EV a lot.

Next, based on the assumption that an EV user will continuously to charge his/her EV for at least six month. So we group the data by the users and calculate the longest time of continuent signal exists as the maxperiod and sort the data by it.

```r
##Calculate the longest time of continuent signal
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.6.3
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

```r
result=as.data.table(result)[, maxperiod := sequence(.N), by = rleid(EV_user_flag)][EV_user_flag == "0"

#build a new table for selecting
r1=group_by(result,id)%>%summarise(total_month=sum(EV_user_flag),
                                   data_period=n(),
                                   mean_times=mean(Singals_Num),
                                   sd_times=sd(Singals_Num),
                                   maxperiod=max(maxperiod),
                                   mean_dur_var=mean(Duration.Variance),
                                   mean_estP_mean=mean(mean.estPower),
                                   mean_estP_var=mean(estPower.Variance))
r2=r1[order(r1$maxperiod,decreasing = TRUE),]
```

The results show below:

```r
kable(r2[1:50,1:6],caption = "EV Charging Signals Statistics Group By Users")
```

Table 4: EV Charging Signals Statistics Group By Users

id	total_month	data_period	mean_times	sd_times	maxperiod
RDM2377306	23	24	9.6666667	4.5556145	23
RDM2899443	23	24	4.5000000	2.9192018	23
RDM9182391	22	24	4.8750000	2.9534431	22
RDM0749394	21	24	4.9166667	3.4378541	17
RDM9650194	21	24	5.0833333	3.3997016	16
RDM4955610	22	24	19.5833333	10.1549588	15
RDM0369914	14	15	4.8666667	1.9952324	14
RDM2704419	21	24	13.0000000	10.4714934	12
RDM0077172	22	24	5.5000000	4.3238469	11
RDM2988595	18	24	7.1250000	8.0989130	11
RDM6871043	21	24	6.1666667	6.5784012	11
RDM7850846	19	24	4.4583333	4.9341681	11
RDM7938157	19	24	9.3333333	10.4243305	11
RDM9414917	19	24	3.7916667	3.1204677	11
RDM8152672	18	24	3.3333333	3.7840244	10
RDM9024599	19	24	2.0416667	1.6010640	10
RDM3118666	15	24	5.4583333	6.1359891	9
RDM9574879	20	24	9.9583333	10.5190476	9
RDM1077107	16	24	1.2083333	1.1412871	8

id	total_month	data_period	mean_times	sd_times	maxperiod
RDM2589665	11	24	3.2500000	5.3750632	8
RDM7643923	10	22	1.5909091	2.4622694	8
RDM7720847	14	24	2.0000000	2.7186953	8
RDM7486143	19	24	5.7916667	4.3636758	7
RDM7836583	15	24	8.8750000	8.9555060	7
RDM1619286	8	24	2.1666667	3.9416027	6
RDM2782043	10	24	2.4583333	4.4816453	6
RDM3105955	11	24	3.1666667	5.1132111	6
RDM3383685	10	24	3.1666667	5.4825150	6
RDM3641122	10	24	4.3333333	7.6138699	6
RDM5546473	10	24	1.8333333	3.5833755	6
RDM6363093	12	24	1.4166667	2.3015433	6
RDM6756375	12	24	3.9166667	5.5239925	6
RDM7469421	9	24	3.2916667	5.3930645	6
RDM8167794	10	24	1.2916667	1.9219367	6
RDM8381179	18	24	2.3333333	2.0359095	6
RDM1152875	16	24	1.9583333	2.2161299	5
RDM1307489	7	24	0.9166667	1.8157922	5
RDM1354990	16	24	2.1250000	2.6426683	5
RDM1969805	10	24	1.5000000	2.1058924	5
RDM2940693	16	24	2.1250000	2.8485313	5
RDM3206777	5	24	1.0000000	3.1484986	5
RDM3532764	9	24	5.1250000	8.3734181	5
RDM3831919	8	24	1.0000000	1.7937088	5
RDM3939544	11	24	2.8333333	4.6873249	5
RDM4224697	7	24	0.4166667	0.6538625	5
RDM5411465	12	24	1.5000000	2.1467873	5
RDM6632060	10	24	6.3333333	9.4439613	5
RDM7221869	9	24	0.5416667	0.8329709	5
RDM8114691	12	24	1.0833333	1.4116493	5
RDM9874873	10	24	3.2916667	5.5987512	5

Based on the knowledge of EV usage pattern, we set several rules to filter this dataset and get our final prediction results.

```
# set the rules
final_result=r2[which(r2$total_month/r2$data_period>=0.5
                      &r2$maxperiod/r2$total_month>=0.6
                      &r2$mean_times>1
                      &r2$mean_times<15
                      &r2$sd_times<10
                      &r2$mean_estP_mean>2800),]
Score=c("Right","Wrong","Right","Right","Right","Right","Right","Right")
final_result=mutate(final_result,Score)
final_result=final_result[c("id","total_month","mean_times","maxperiod","mean_estP_mean","Score")]
kable(final_result,caption = "Prediction Results")
```

Table 5: Prediction Results

id	total_month	mean_times	maxperiod	mean_estP_mean	Score
RDM2377306	23	9.666667	23	4402.195	Right
RDM2899443	23	4.500000	23	3926.365	Wrong

id	total_month	mean_times	maxperiod	mean_estP_mean	Score
RDM9182391	22	4.875000	22	5073.680	Right
RDM0749394	21	4.916667	17	4754.033	Right
RDM9650194	21	5.083333	16	3683.389	Right
RDM0369914	14	4.866667	14	6394.385	Right
RDM2988595	18	7.125000	11	3211.822	Right
RDM3118666	15	5.458333	9	3570.606	Right

Finally, in our prediction results, there are 7 true EV users among 8 estimated EV users. My model's accuracy is around 87.5%.

This model is traning-free and very fast. However, in this period, it is really hard to collect enough training datasets. That means you can't just call the customers to ask whether they have EV. If we have a larger training dataset. We can use machine learning model to classify after we use the NILM model to dissagregate the EV signal.

Reference

[1] G. W. Hart, "Nonintrusive appliance load monitoring," Proceedings of the IEEE, vol. 80, no. 12, pp. 1870–1891, 1992.

[2] M. Zeifman and K. Roth, "Nonintrusive appliance load monitoring: Review and outlook," Consumer Electronics, IEEE Transactions on, vol. 57, no. 1, pp. 76–84, 2011.

[3] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar, "Non-intrusive load monitoring approaches for disaggregated energy sensing: a survey,"Sensors, vol. 12, no. 12, pp. 16 838–16 866, 2012.

[4] K. Clement-Nyns, E. Haesen, and J. Driesen, "The impact of charging plug-in hybrid electric vehicles on a residential distribution grid," Power Systems, IEEE Transactions on, vol. 25, no. 1, pp. 371–380, 2010.