Case1 Report

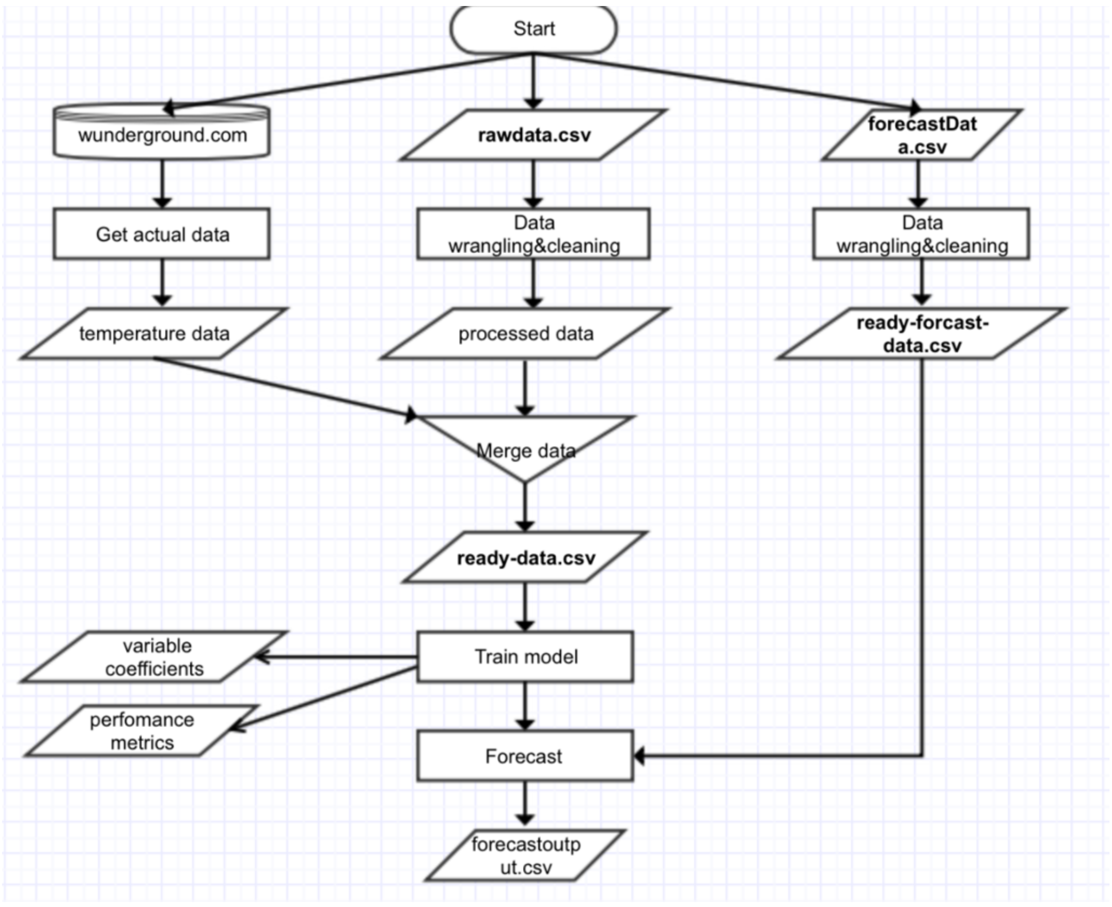
Team 4

Jianxing Lu

Wenjin Cao

Qiaomin Ling

1 Flow Chart



2 Report Description

Weather forecasting is the application of science and technology to predict the state of the atmosphere for a given location, and are made by collecting quantitative data about the current state of the atmosphere at a given place and using scientific understanding of atmospheric processes to project how the atmosphere will change.

In this report, we will focus on the process of generating a KWH (kilowatt hour) forecast model by using Multiple Linear Regression method, and discuss how to prepare the raw data and how to use feature selection and feature transformation to get ideal model. At last, the predicted result will be compared to the real data and analyze the accuracy of forecast model by RMS error, MAPE and MAE.

3 Data wrangling and cleansing

Manipulation environment: R Studio

Package Using: weatherData (used to get weather data from 2014/01/01 to 2014/12/31), devtools (used to install weatherData package by using github_pull() function), plyr (Tools for Splitting, Applying and Combining Data)

In order to enhance the accuracy of forecast model and reduce the calculation, data wrangling and cleansing is the better solution and they have to be completed in high quality before generating model. Data wrangling and cleansing is loosely the process of manually converting or mapping data from one "raw" form into another format that allows for more convenient consumption of the data with the help of semi-automated tools such as R Studio. In this assignment, we used those effective methods to get ideal format data shown below.

1. For the rawdata.csv file, we noticed that there are three rows information for every day, only the first row is the key value for this assignment, so the other two rows need to be remove firstly. Next, the third and fourth columns of rawdata.csv are not mentioned in sample data format, we need to remove them too. And result will be stored into DatawithEnergy.csv file for back up.
2. To calculate day of week, we use for loop and add a new column by using cbind(). Same method is used in calculating year column to fill 2014 for all rows.
3. There are 12 months in one year, so we need to use 12 for loop to add the day for every row and add this column by cbing().
4. Same method to fill 12 months
5. To determine whether one day is weekday or not, we need to use one for loop to calculate.
6. Use for loop to add hour column and peak hour column
7. Kwh is aggregated hourly. Sum of 12 observations (5 min intervals rolled up to hourly)
8. Use getWeatherForDate API to get weather information in details, and store them into rawdata-weather.csv file.
9. Sort the weather data by DateUTC and remove all the irrelative data. There is a problem that the weather data in one day are more than 24, the record of 54 minute, however, is shown in every hour, that means we have to reduce the redundant data and keep the record of the 54th minute hourly.
10. Delete repeat data by for loop.
11. After delete repeat data, there are still some missing weather data, our solution is find the missing data out and using the average temperature of the day to replace the missing data.
12. Tore the final data into readydata.csv file, and using MissingweatherData.csv for testing.

After generating the forecast model, we can use real data to predict the weather forecast. Also, those real data need to input by specific format, the method to modify data is shown below.

Read forecastData.csv and keep all relative data, using gsub() to split data as proper value for every column. Using for loop to get peak hour, weekday and day of week. Finally, we store them as ready-forecast-data.csv file

4 Model & evaluation

1) Before building a linear regression model, we need to observe and analyze the data.

   Multiple linear regression analysis makes several key assumptions: linear relationship, multivariate normality, no or little multicollinearity, no auto-correlation, homoscedasticity.

- Multiple linear regression needs the relationship between the independent and dependent variables to be linear. It is also important to check for outliers since multiple linear regression is sensitive to outlier effects.

  We calculated the min, max, median of the data to see if there is any abnormal data.

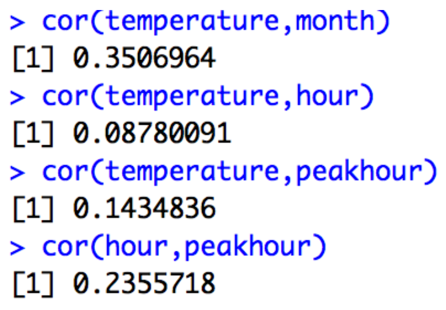
  ```
  readydata<-read.csv("ready-data.csv",header=T)
  Mean <- sapply(readydata,mean)
  Median <- sapply(readydata,median)
  Min <- sapply(readydata,min)
  Max <- sapply(readydata,max)
  SD <- sapply(readydata,sd)
  cbind(Mean, Median, Min, Max, SD)
  ```

  We found NA in the data. So we use an average to replace it.
  All data is in a normal range. No outlier.

- The multiple linear regression analysis requires all variables to be normal. This assumption can best be checked with a histogram and a fitted normal curve or a Q-Q-Plot.
  (Discussed later)

- Multiple linear regression assumes that there is little or no multicollinearity in the data.

  i) Correlation matrix.
     We use correlation analysis to see if the predictors are independent from each other. (Correlation among all independent variables the correlation coefficients need to be smaller than .08)
     These are not:



```
> cor(temperature,month)
[1] 0.3506964
> cor(temperature,hour)
[1] 0.08780091
> cor(temperature,peakhour)
[1] 0.1434836
> cor(hour,peakhour)
[1] 0.2355718
```

ii) T or VIF.

Tolerance – the tolerance measures the influence of one independent variable on all other independent variables. $T = 1 - R^2$. With $T < 0.2$ there might be multicollinearity in the data and with $T < 0.01$ there certainly is.

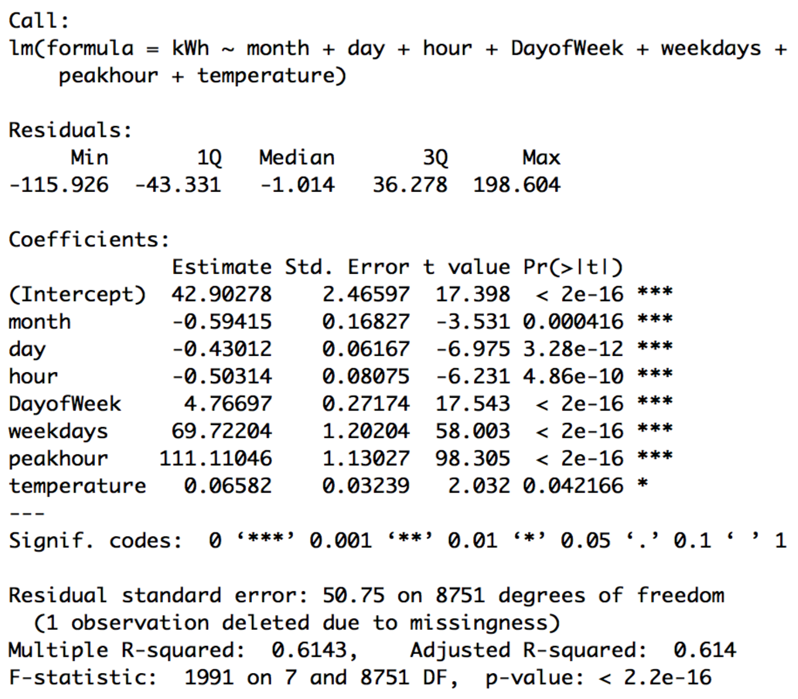
Variance Inflation Factor. $VIF = 1/T$.

(Discussed later)

2) Fit a model.

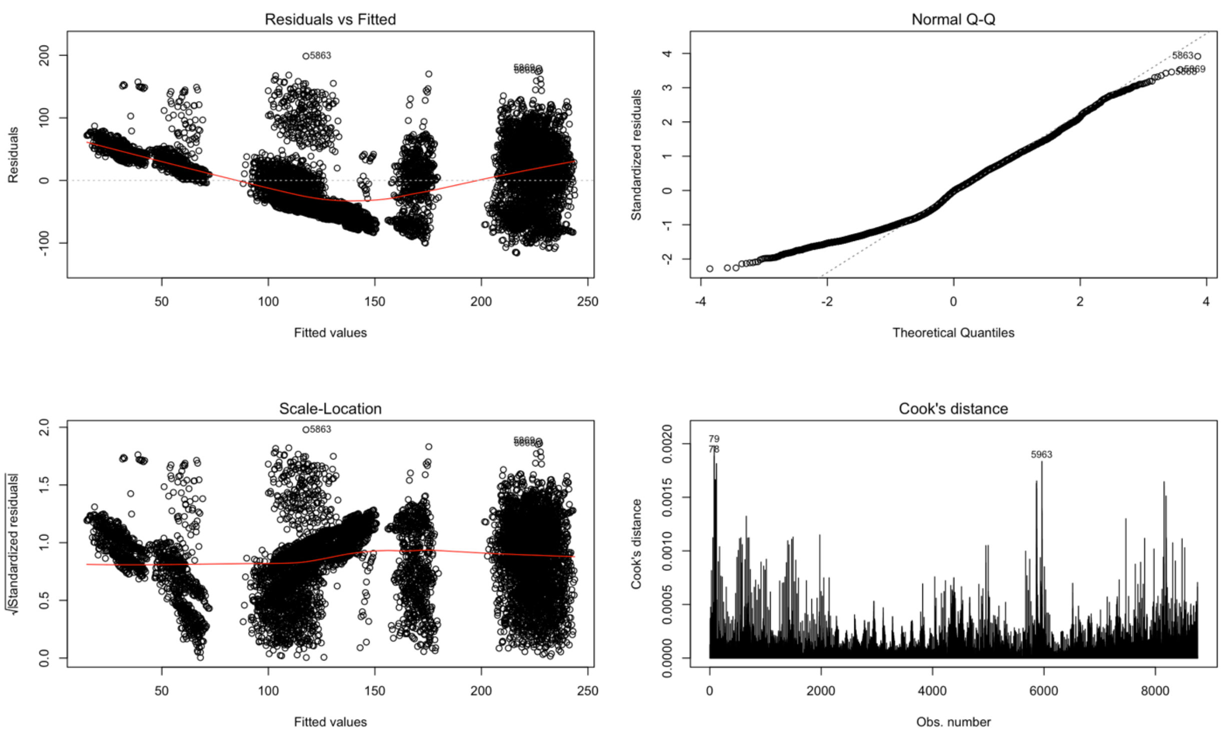
- Try all the possible predictors (except Date and year)

  model1 <- lm(kWh ~ month + day + hour + DayofWeek + weekdays + peakhour + temperature)
  summary(model1)



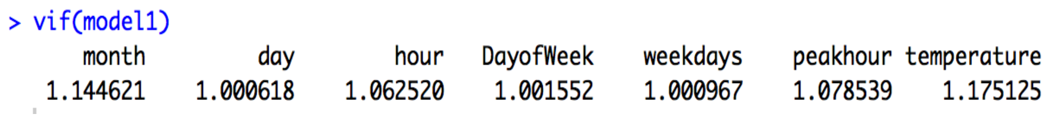
```
Call:
lm(formula = kWh ~ month + day + hour + DayofWeek + weekdays +
    peakhour + temperature)

Residuals:
     Min       1Q   Median       3Q      Max
-115.926  -43.331   -1.014   36.278  198.604

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  42.90278    2.46597  17.398  < 2e-16 ***
month        -0.59415    0.16827  -3.531 0.000416 ***
day          -0.43012    0.06167  -6.975 3.28e-12 ***
hour         -0.50314    0.08075  -6.231 4.86e-10 ***
DayofWeek     4.76697    0.27174  17.543  < 2e-16 ***
weekdays     69.72204    1.20204  58.003  < 2e-16 ***
peakhour    111.11046    1.13027  98.305  < 2e-16 ***
temperature   0.06582    0.03239   2.032 0.042166 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 50.75 on 8751 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.6143,    Adjusted R-squared:  0.614
F-statistic:  1991 on 7 and 8751 DF,  p-value: < 2.2e-16
```

- par(mfrow = c(2,2))
  plot(model1,which=c(1:4))



- Multicollinearity checking

  install.packages("car")
  library(car)
  vif(model1)

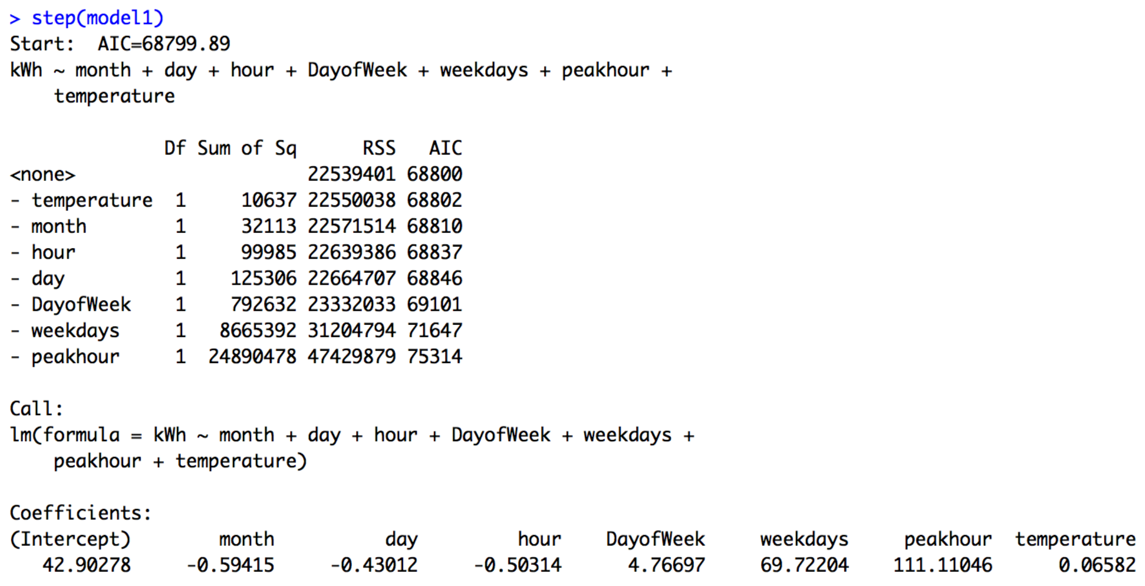


```
> vif(model1)
      month         day        hour   DayofWeek    weekdays   peakhour temperature
   1.144621    1.000618    1.062520    1.001552    1.000967    1.078539    1.175125
```

With VIF < 10, there is no indication for multicollinearity to be present.

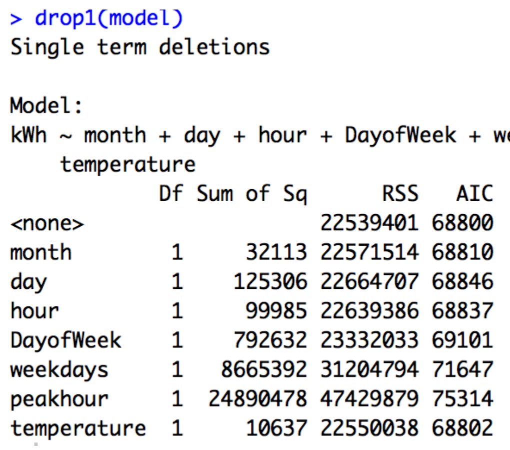
- Step regression

  step(lm(kWh ~ month + day + hour + DayofWeek + weekdays + peakhour + temperature,data = readydata),direction = "backward")
  or step(model1)



```
> step(model1)
Start:  AIC=68799.89
kWh ~ month + day + hour + DayofWeek + weekdays + peakhour +
    temperature

              Df Sum of Sq      RSS   AIC
<none>                     22539401 68800
- temperature  1     10637 22550038 68802
- month        1     32113 22571514 68810
- hour         1     99985 22639386 68837
- day          1    125306 22664707 68846
- DayofWeek    1    792632 23332033 69101
- weekdays     1   8665392 31204794 71647
- peakhour     1  24890478 47429879 75314

Call:
lm(formula = kWh ~ month + day + hour + DayofWeek + weekdays +
    peakhour + temperature)

Coefficients:
(Intercept)        month          day         hour    DayofWeek     weekdays     peakhour  temperature
   42.90278     -0.59415     -0.43012     -0.50314      4.76697     69.72204    111.11046      0.06582
```

Use AIC value to see if omitting any variable would make a better model.



```
> drop1(model)
Single term deletions

Model:
kWh ~ month + day + hour + DayofWeek + weekdays + peakhour +
    temperature
            Df Sum of Sq      RSS   AIC
<none>                   22539401 68800
month        1     32113 22571514 68810
day          1    125306 22664707 68846
hour         1     99985 22639386 68837
DayofWeek    1    792632 23332033 69101
weekdays     1   8665392 31204794 71647
peakhour     1  24890478 47429879 75314
temperature  1     10637 22550038 68802
```

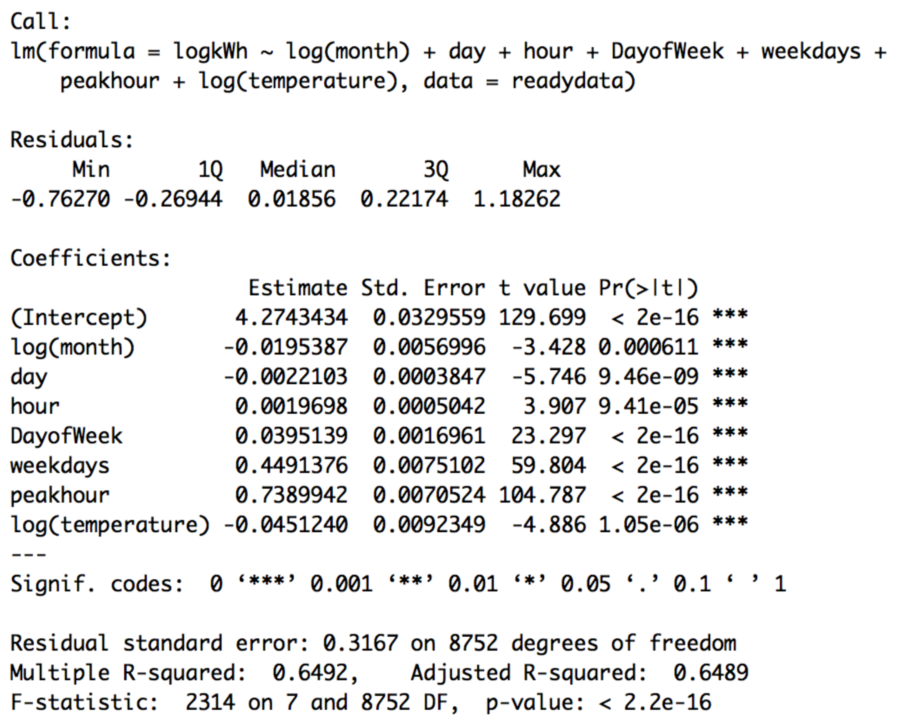
We decide to use all seven predictors: month, day, hour, DayofWeek, weekdays, peakhour, temperature.

- Data transformation
  We can transform both the predictors (independent variables) and the response (dependent variable).
  There are many ways to transform variables to achieve linearity for regression analysis. We need to test all the methods on data.
  Testing the effect of a transformation method involves looking at residual plots and correlation coefficients.
  i) Construct a residual plot. A residual plot reveals departures from linearity. If the plot pattern is random, do not transform data. If the plot pattern is not random, transform.
     We would expect the residuals to be randomly scattered without showing any systematic patterns.

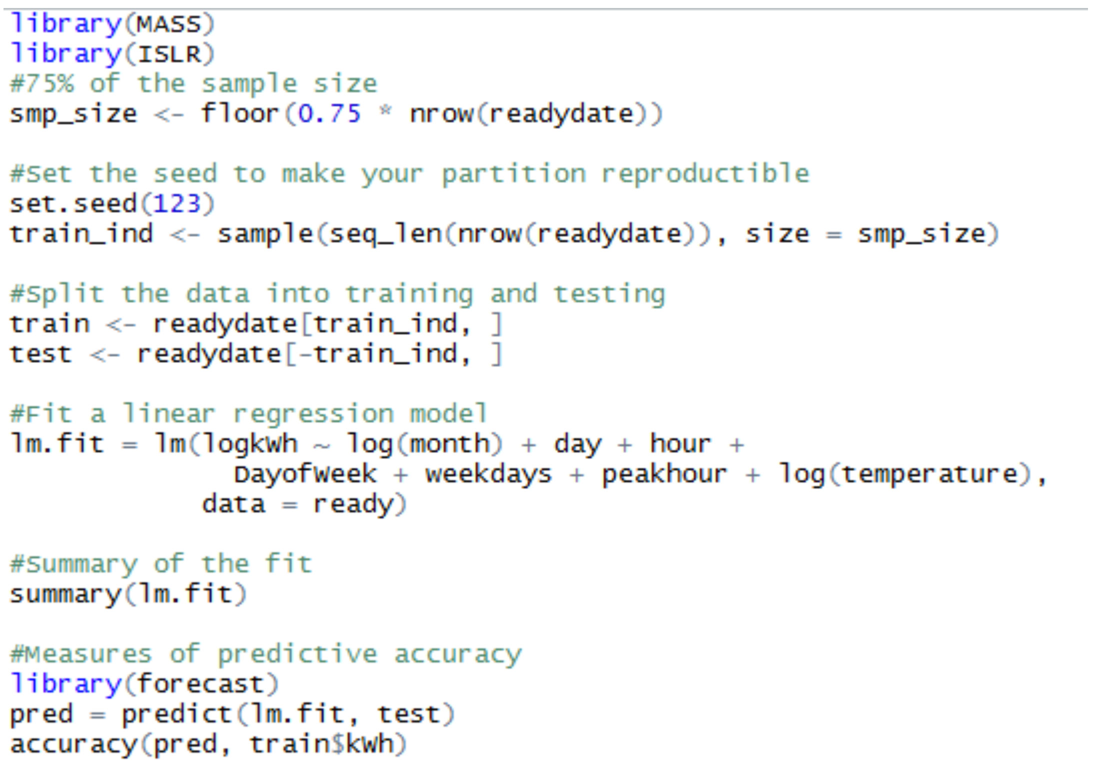
ii) Compute the coefficient of determination (R^2), based on the transformed variables. If the transformed R^2 is greater than the raw-score R^2, the transformation was successful.

The best transformation method (exponential model, quadratic model, reciprocal model, etc.) will depend on nature of the original data. The only way to determine which method is best is to try each and compare the result.

readydata$logkWh <- log(readydata$kWh)
modellog <- lm(logkWh ~ log(month) + day + hour + DayofWeek + weekdays + peakhour + log(temperature),data = readydata)
summary(modellog)

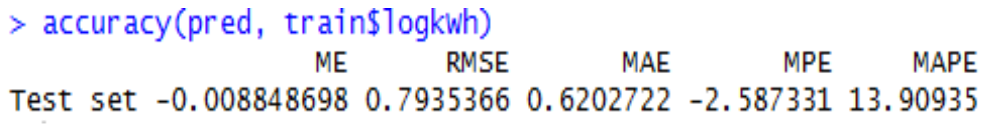


```
Call:
lm(formula = logkWh ~ log(month) + day + hour + DayofWeek + weekdays + 
    peakhour + log(temperature), data = readydata)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.76270 -0.26944  0.01856  0.22174  1.18262 

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)    
(Intercept)       4.2743434  0.0329559 129.699  < 2e-16 ***
log(month)       -0.0195387  0.0056996  -3.428 0.000611 ***
day              -0.0022103  0.0003847  -5.746 9.46e-09 ***
hour              0.0019698  0.0005042   3.907 9.41e-05 ***
DayofWeek         0.0395139  0.0016961  23.297  < 2e-16 ***
weekdays          0.4491376  0.0075102  59.804  < 2e-16 ***
peakhour          0.7389942  0.0070524 104.787  < 2e-16 ***
log(temperature) -0.0451240  0.0092349  -4.886 1.05e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3167 on 8752 degrees of freedom
Multiple R-squared:  0.6492,	Adjusted R-squared:  0.6489 
F-statistic:  2314 on 7 and 8752 DF,  p-value: < 2.2e-16
```



```r
library(MASS)
library(ISLR)
#75% of the sample size
smp_size <- floor(0.75 * nrow(readydate))

#Set the seed to make your partition reproductible
set.seed(123)
train_ind <- sample(seq_len(nrow(readydate)), size = smp_size)

#Split the data into training and testing
train <- readydate[train_ind, ]
test <- readydate[-train_ind, ]

#Fit a linear regression model
lm.fit = lm(logkwh ~ log(month) + day + hour +
                 DayofWeek + weekdays + peakhour + log(temperature),
               data = ready)

#Summary of the fit
summary(lm.fit)

#Measures of predictive accuracy
library(forecast)
pred = predict(lm.fit, test)
accuracy(pred, train$kwh)
```

Running result:



```
> accuracy(pred, train$logkwh)
                   ME      RMSE       MAE       MPE     MAPE
Test set -0.008848698 0.7935366 0.6202722 -2.587331 13.90935
```

- Model Evaluation
  We can assess the regression model by how well the model fits the data and how well is the model in predicting the outcome. We analyze the performance to decide whether the numerical results quantifying hypothesized relationships between variables, obtained from regression analysis, are acceptable as descriptions of the data.

  i) R-squared: 0.649
  R-squared describes has the useful property that its scale is intuitive: it ranges from zero to one, with zero indicating that the proposed model does not improve prediction over the mean model and one indicating perfect prediction. The R-squared of our model is not very good but still promising.

  ii) RMS error: 0.794
  Whereas R-squared is a relative measure of fit, RMSE is an absolute measure of fit. It s a good measure of how accurately the model predicts the response. As lower values of RMSE indicate better fit, the result we got is not good.

  iii) MAPE: 13.909.

It expresses accuracy as a percentage. Usually it should be lower that 15% to indicate a good model.

iv) MAE: 0.620

v) F-test : F-statistic:2314 on 7 and 8752 DF
A significant F-test indicates that the observed R-squared is reliable, and is not a spurious result of oddities in the data set. Thus, the F-test determines whether the proposed relationship between the response variable and the set of predictors is statistically reliable, and can be useful when the research objective is either prediction or explanation.

vi) p value: <2.2e-16
It helps determine the significance of the results. The p-value is a number between 0 and 1 and interpreted in the following way: a small p-value (typically ≤ 0.05) indicates strong evidence against the null hypothesis, so we can reject the null hypothesis; a large p-value (> 0.05) indicates weak evidence against the null hypothesis.

vii) T value:
T-statistic is a ratio of the departure of an estimated parameter from its notional value and its standard error. The t-value measures the size of the difference relative to the variation in your sample data. Put another way, T is simply the calculated difference represented in units of standard error. The greater the magnitude of T (it can be either positive or negative), the greater the evidence against the null hypothesis that there is no significant difference. The closer T is to 0, the more likely there isn't a significant difference.
For our model, peakhour, weekdays, DayofWeek have the greated T value. These variables contribute the most to the model.

viii) Residuals:

Min	1Q	Median	3Q	Max
-0.76270	-0.26944	0.01856	0.22174	1.18262

Along with the residual plot above, we can tell that the regression residuals are random.