Assignment 1：

7. (i) Explain how one can determine how to improve the predicted popularity of an article with either of the regression models considered here.

For GLM model, after comparing the positive estimate values, the largest one is n_tokens_title, which means that this variable has the most significant effect on the predicted popularity. If n_tokens_title is improved, the predicted popularity of an article will also improved. Besides, some other significant variables such as “num_imgs” , “global_subjectivity”, “kw_min_max” and so on are also important positive effect on the predicted popular. Improving them can also improve the predicted popularity. Additionally, “data_channel_is_socmed” is also a positive effect. In other words, if we want to improve shares, improving the data come from the channel socmed is a good choice. In terms of negative effect, the variables “n_tokens_content” , “LDA_02” , “data_channel_is_entertainment” are significant variables to have negative effect on the predicted popularity. In order to improve shares, reducing the value of these variables and avoiding the data coming from the channel entertainment is a good choice.

For logistic regression model, after comparing the positive estimate values, the largest is n_non_stop_words, which means that n_non_stop_words will increase the probability of the articles which will have more than 1000 shares. Therefore, it could also be explained that increase the values of the variables which have positive effect mentioned above can improve the probability of the articles which will have more than 1000 shares. Moreover, to decrease the values of the variables which have mentioned in GLM model can reduce the probability of the articles which will have more than 1000 shares.

(ii) Using your two fitted regression models, identify the articles with the highest predicted popularity and predicted probability of being popular.
I use command lines which are listed below to predict the articles with the highest predicted popularity and predicted probability of being popular by using GLM and logistic model.

```
#GLM
linear_predict_result <- lm6%>%predict(ds)
ONP[rownames(ONP) == which(linear_predict_result %in% max(linear_predict_result)),]

#logistic regression
logistic_predict_result <- predict(lo.model.log,newdata=logit.news.log,
                                   type="response")
ONP[rownames(ONP) == which(logistic_predict_result %in% max(logistic_predict_result)),]
```

The prediction result of using GLM model is:
http://mashable.com/2013/05/20/dove-ad-most-watche/

The prediction result of using logistic model is:
http://mashable.com/2013/08/30/3d-fax-machine/

(iii) For each of your two fitted regression models, list the attributes of two hypothetical articles (fake news?) which would give the highest possible predicted popularity and predicted probability of being popular, respectively. You should give values for every attribute, but for each variable, keep them within the range seen in the dataset.

For GLM:

```
> fake_high
                                        url timedelta n_tokens_title n_tokens_content
2766 http://mashable.com/2013/02/24/oscar-comic/       683              8               21
     n_unique_tokens n_non_stop_words n_non_stop_unique_tokens num_hrefs num_self_hrefs
2766       0.9523809        0.9999999              0.9999999         1               1
     num_imgs num_videos average_token_length num_keywords data_channel_is_lifestyle
2766        0          0              4.095238            8                        0
     data_channel_is_entertainment data_channel_is_bus data_channel_is_socmed
2766                             0                   0                      0
     data_channel_is_tech data_channel_is_world kw_min_min kw_max_min kw_avg_min kw_min_max
2766                    0                     0         217       697    374.125       8300
     kw_max_max kw_avg_max kw_min_avg kw_max_avg kw_avg_avg self_reference_min_shares
2766      80400      46475   1361.269   4401.559   3078.135                         0
     self_reference_max_shares self_reference_avg_sharess weekday_is_monday weekday_is_tuesday
2766                         0                           0                0                   0
     weekday_is_wednesday weekday_is_thursday weekday_is_friday weekday_is_saturday
2766                    0                   0                 0                   0
     weekday_is_sunday is_weekend    LDA_00    LDA_01     LDA_02  LDA_03     LDA_04
2766                 1          1 0.0250579 0.7746682 0.02501979 0.15025 0.02500413
     global_subjectivity global_sentiment_polarity global_rate_positive_words
2766                 0.3                     0.125                 0.04761905
     global_rate_negative_words rate_positive_words rate_negative_words avg_positive_polarity
2766                          0                   1                   1                  0.25
     min_positive_polarity max_positive_polarity avg_negative_polarity min_negative_polarity
2766                  0.25                  0.25                     0                     0
     max_negative_polarity title_subjectivity title_sentiment_polarity abs_title_subjectivity
2766                     0                0.7                     -0.6                     0.2
     abs_title_sentiment_polarity shares
2766                          0.6   3000
```

For logistic model:

```
> fake_high1
                                                  url timedelta n_tokens_title
16466 http://mashable.com/2013/11/21/iran-blocks-cryptocat/       413             11
      n_tokens_content n_unique_tokens n_non_stop_words n_non_stop_unique_tokens num_hrefs
16466              190       0.6460674                1                0.6929134        17
      num_self_hrefs num_imgs num_videos average_token_length num_keywords
16466              2        0          0              4.910526            6
      data_channel_is_lifestyle data_channel_is_entertainment data_channel_is_bus
16466                         0                             0                  1
      data_channel_is_socmed data_channel_is_tech data_channel_is_world kw_min_min kw_max_min
16466                      0                    0                     0          4       1400
      kw_avg_min kw_min_max kw_max_max kw_avg_max kw_min_avg kw_max_avg kw_avg_avg
16466   416.3333       3300     843300     381600       2350      10407    4478.21
      self_reference_min_shares self_reference_max_shares self_reference_avg_sharess
16466                    652900                    652900                    652900
      weekday_is_monday weekday_is_tuesday weekday_is_wednesday weekday_is_thursday
16466                 0                  0                    0                  1
      weekday_is_friday weekday_is_saturday weekday_is_sunday is_weekend    LDA_00    LDA_01
16466                 0                   0                 0          0 0.5245371 0.3753347
         LDA_02     LDA_03     LDA_04 global_subjectivity global_sentiment_polarity
16466 0.03339501 0.03339858 0.03333455           0.4916667                 0.2354167
      global_rate_positive_words global_rate_negative_words rate_positive_words
16466                 0.02631579                          0                   1
      rate_negative_words avg_positive_polarity min_positive_polarity max_positive_polarity
16466                   1             0.3266667                   0.1                   0.5
      avg_negative_polarity min_negative_polarity max_negative_polarity title_subjectivity
16466                     0                     0                     0                  0
      title_sentiment_polarity abs_title_subjectivity abs_title_sentiment_polarity shares
16466                        0                    0.5                             0  11900
```

(iv) Based on your analysis of dependence between variables in the dataset, comment on whether or not these hypothetical articles could be produced.

I think this article can be produced because all of the values from different variables have met the requirement to be produced.

Assignment 2：

Part I Re-analysis of the UCI online news popularity dataset

1. Choose a suitable generalised linear model and suitable link function and explain these decisions. Fit the model to the data and report summary results.

We need to use a generalised linear model(glm) and a non-negative distribution to predict number of shares for a new article. Because the values of shares which is dependent variable are count variable of type and independent variables are continuous and categorical variables, there are three distributions of glm could be used to deal with dataset, such as gaussian regression, poisson regression and gamma regression and negative binomial regression. To find a suitable glm, I compare AIC and RMSE and R^2 of these three glm of different distribution.
The form and snapshots of these details are listed below.

(1) Dispersion parameter for poisson family

```
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 257399786  on 38803  degrees of freedom
Residual deviance: 231196242  on 38759  degrees of freedom
AIC: 231557948
```

(2) Dispersion parameter for gamma family

```
(Dispersion parameter for Gamma family taken to be 7.278776)

    Null deviance: 50868  on 38803  degrees of freedom
Residual deviance: 43533  on 38759  degrees of freedom
AIC: 702279

Number of Fisher Scoring iterations: 14
```

(3) Dispersion parameter for gaussian family

```
(Dispersion parameter for gaussian family taken to be 135024180)

    Null deviance: 5.3348e+12  on 38803  degrees of freedom
Residual deviance: 5.2334e+12  on 38759  degrees of freedom
AIC: 836616

Number of Fisher Scoring iterations: 2
```

(4) Dispersion parameter for negative binomial family

```
(Dispersion parameter for Negative Binomial(1.0261) family taken to be 1)

    Null deviance: 52172  on 38803  degrees of freedom
Residual deviance: 44649  on 38759  degrees of freedom
AIC: 701818
```

Deviance is a measure of goodness of fit of a model. Higher numbers always indicates bad fit. The null deviance shows how well the response variable is predicted by a model that includes only the intercept (grand mean) where as residual with inclusion of independent variables. Comparing the null deviance value of different family type, we could found that the null deviance value of gamma family is smaller than others, which means gamma regression is better fit of glm.

	AIC	RMSE	R^2
Poisson	231557948	11661.01	0.101801
Gamma	702279	94724.52	0.1441833
Gaussian	836616	11628.08	0.01900314
negative binomial	701818	94012.61	0.1441977

The Akaike information criterion (AIC) is a mathematical method for evaluating how well a model fits the data it was generated from. In statistics, AIC is used to compare different possible models and determine which one is the best fit for the data. The smaller the value of AIC, the better the model fitted.
From the form listed above, we could find that the value of AIC of gamma regression and negative binomial are smaller.

R-squared measures the strength of the relationship between linear model and the dependent variables on a 0 - 100% scale, which means that the higher the value of R-squared, the better the model fitted.
From the form listed above, we could find that the value of R-squared of gamma regression and negative binomial are higher.

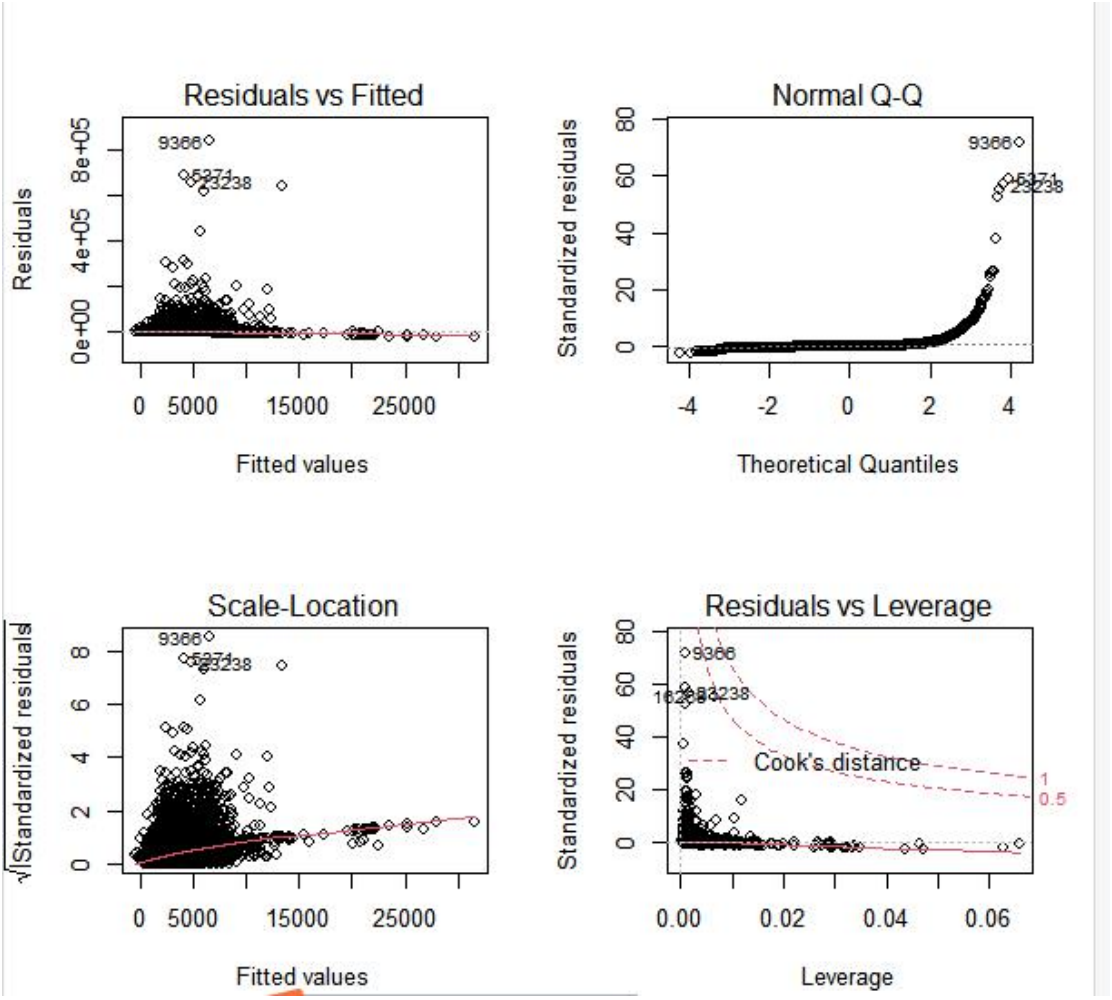
The RMSE is the square root of the square root of the deviation between the observed value and the truth value and the ratio of the observed number. Thus, it's always used to measure the deviation between the observed value and the truth value. It is more sensitive to outliers. This means that the smaller the value of RMSE, the better the model fitted.
From the form listed above, we could find that the value of RMSE of poisson regression and gaussian regression are smaller.

From what has been discussed above, although the value of RMSE of gamma is not the smallest one, the values of AIC and R-squared and null deviance of gamma regression are better than other regressions. Therefore, I decide to use glm with gamma regression and its link function of log.

2. Produce a 95% predictive interval for each fitted model (ignore uncertainty with respect to model parameters). Compare the results from using this model with those from using a multiple linear regression model (= general linear model) which you would have run for assignment 1. Include consideration of significant variables, measures of goodness of fit, predictive intervals and related plots. Discuss theoretical and practical differences between the two models.

When I use summary() function of multiple linear regression model(mlr) and generalised linear model(glm), I found that significant variables of mlr are num_hrefs and num_self_hrefs and data_channel_is_entertainment and data_channel_is_tech and data_channel_is_world, significant variables of glm are n_tokens_title and n_tokens_content and num_hrefs and num_self_hrefs and num_keywords and data_channel_is_world .

(1) Plots of measures of goodness of fit of mlr



(a)Linearity

From the Residuals vs Fitted plot, we could found that the red line is almost flat at

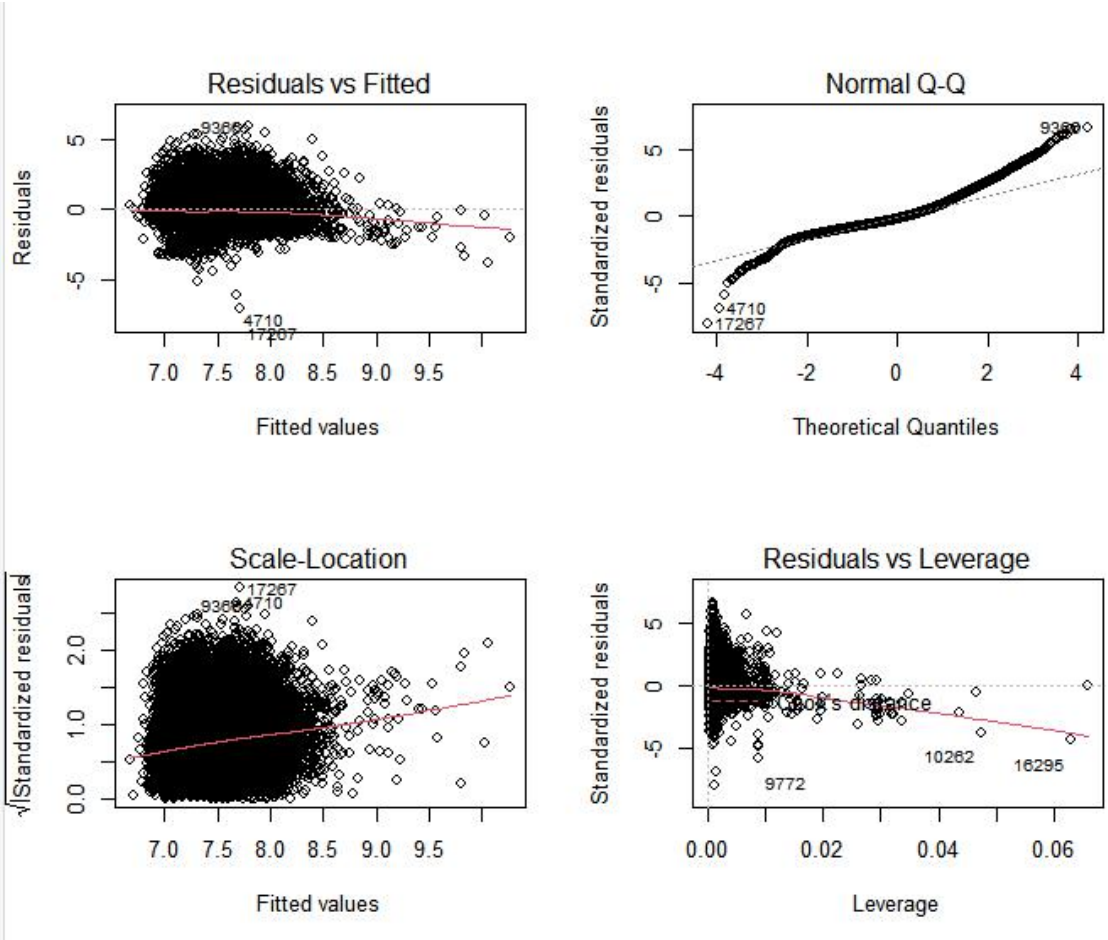
zero value of residuals and a large number of points of fitted values are nearly close and some even match the red line, which means the data fit the linearity assumption.

(b)Normal distribution
From the Q-Q plot and Histogram of residuals, we could found that the points are not fitted to the straight line and the density of residuals is skewed to the right. Therefore, these data are not normally distribution.

(c)Homoscedasticity
From the Scale-Location plot, it could be found out that the red line is positive, which means that the value of the square root of standardized residuals become larger when fitted values become larger. Therefore, these data are not satisfied the homoscedasticity assumption.

(d)High leverage points
From the Residual vs Leverage plot, the top right corner have no points. This means that there are no extreme values.

(2) Plots of measures of goodness of fit of mlr after transformation

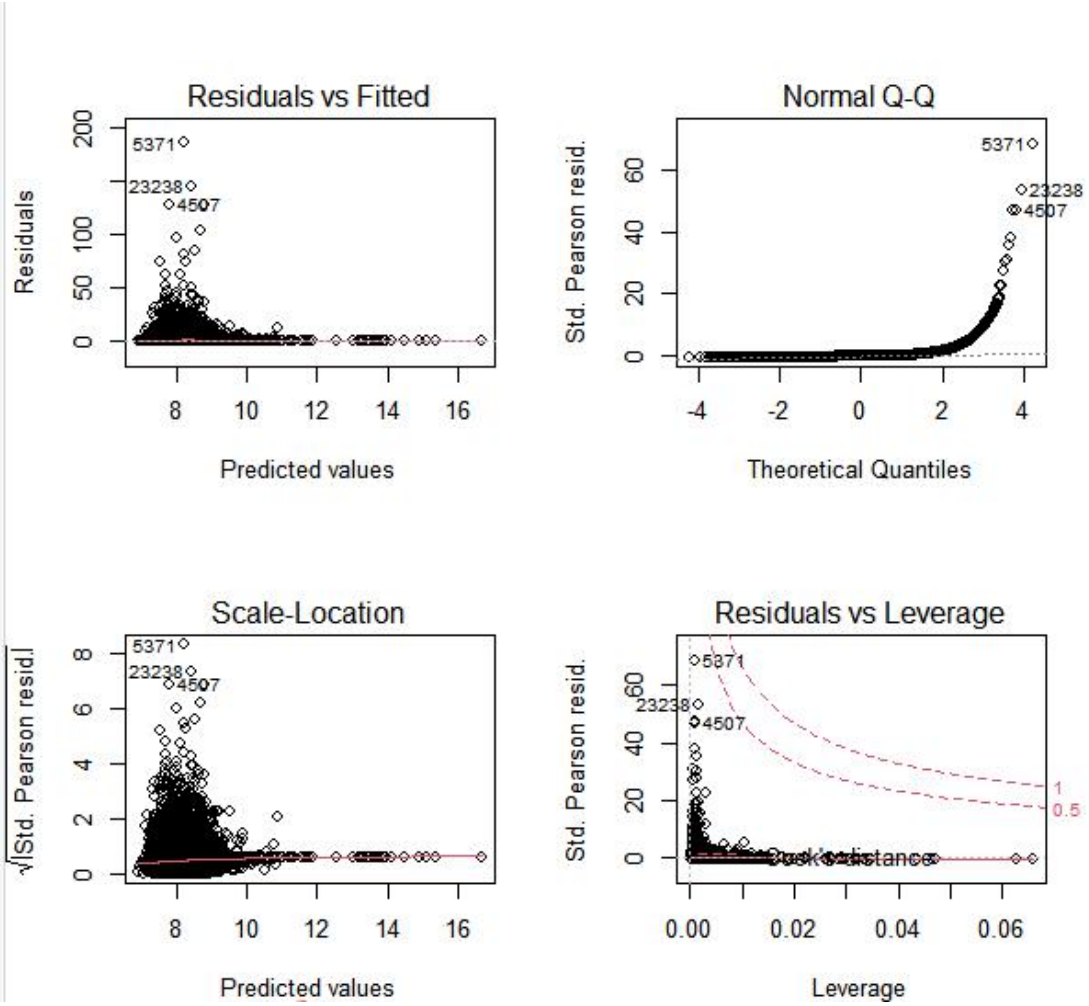


(a) Because the series of data are very different, I take the log of them to make them

become exponential. Specifically, for the data with the attribute: kw_min_min, kw_max_min, kw_avg_min, kw_min_max, kw_max_max, kw_avg_max, kw_min_avg, kw_max_avg, kw_avg_avg. Because the smallest value of these data is -1, which would generate error if taking log of them directly, I add 1 to all the values of these attributes to make them not become negative value and then I take the log of these adjusted values.

(b)From the pictures listed above, from the Q-Q plot and Histogram of residuals, we could found that the points are almost fitted to the straight line. This means that these data are normally distribution after the adjustment.

(3) Plots of measures of goodness of fit of glm before transformation



(a)Linearity

From the Residuals vs Fitted plot, we could found that the red line is almost flat at zero value of residuals and a large number of points of fitted values are nearly close and some even match the red line, which means the data fit the linearity assumption.

(b)Normal distribution

From the Q-Q plot and Histogram of residuals, we could found that the points are not fitted to the straight line and the density of residuals is skewed to the right. Therefore, these data are not normally distribution.

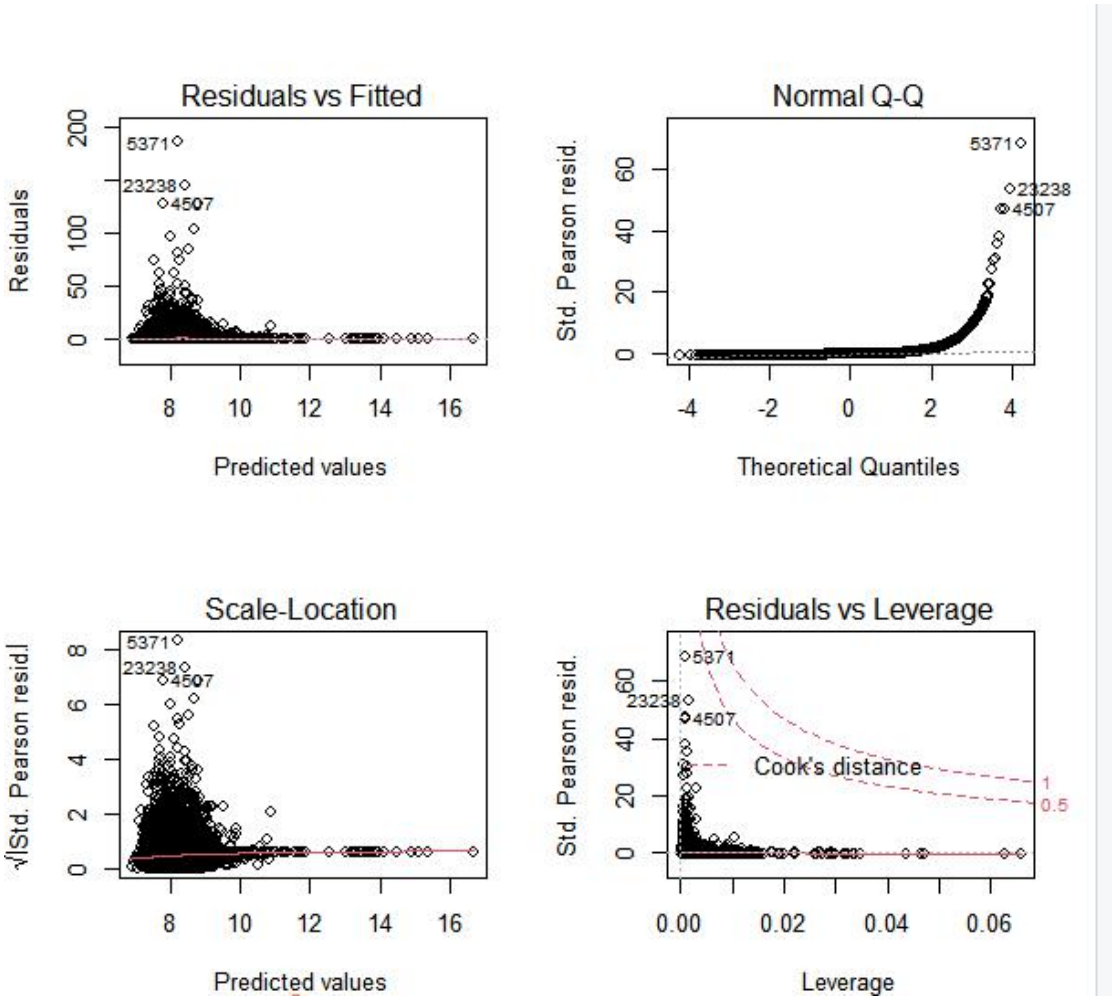
(c)Homoscedasticity

From the Scale-Location plot, it could be found out that the red line is positive, which means that the value of the square root of standardized residuals become larger when fitted values become larger. Therefore, these data are not satisfied the homoscedasticity assumption.

(d)High leverage points

From the Residual vs Leverage plot, the top right corner have no points. This means that there are no extreme values.

(4) Plots of measures of goodness of fit of glm after transformation

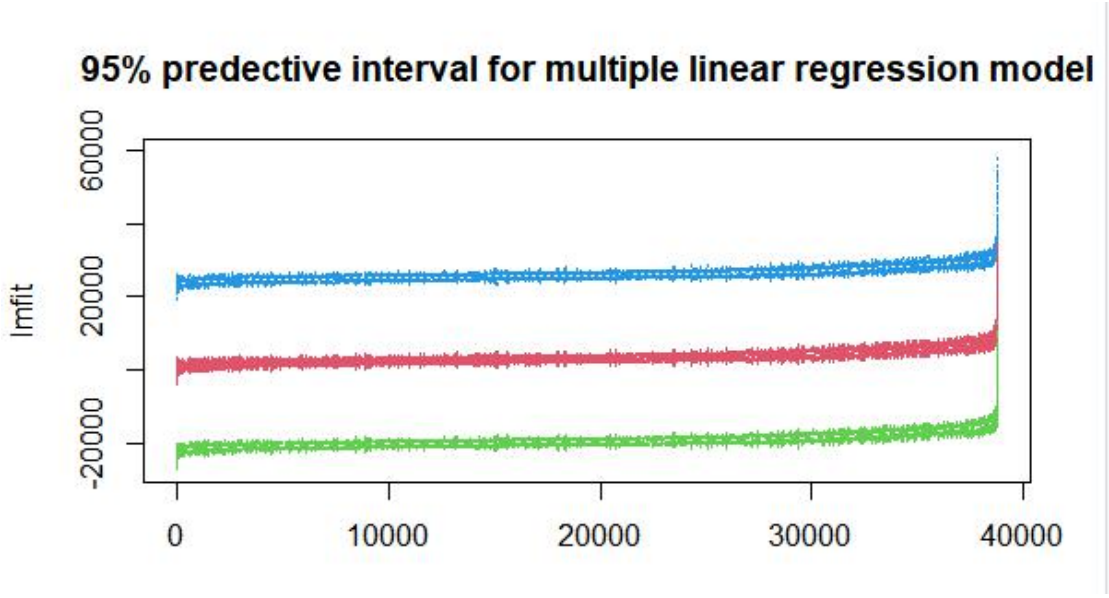


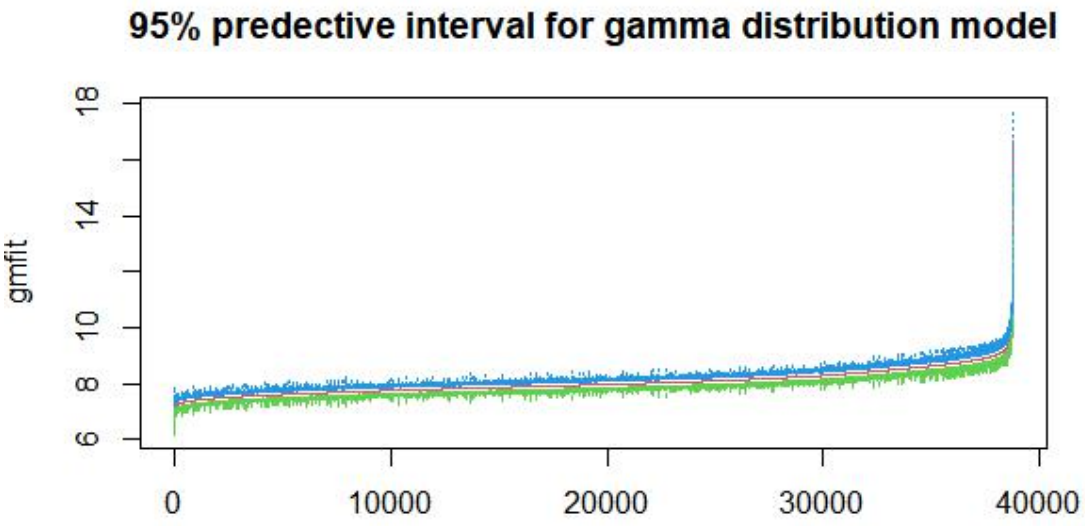
(a) Because the series of data are very different, I take the log of them to make them become exponential. Specifically, for the data with the attribute: kw_min_min, kw_max_min, kw_avg_min, kw_min_max, kw_max_max, kw_avg_max, kw_min_avg, kw_max_avg, kw_avg_avg. Because the smallest value of these data is -1, which

would generate error if taking log of them directly, I add 1 to all the values of these attributes to make them not become negative value and then I take the log of these adjusted values.

(b)From the pictures listed above, from the Q-Q plot and Histogram of residuals, we could found that the points are almost fitted to the straight line. This means that these data are normally distribution after the adjustment.

(5) Plot of predictive interval for two models





The confidence interval is a range of values that's likely to include a population value with a certain degree of confidence. It is often expressed a 95% whereby a population means lies between an upper and lower interval. It measures the degree

of uncertainty or certainty in a sampling method. Wider confidence intervals in relation to the estimate itself indicate instability.

As can be seen from the diagram of the confidence intervals of these two models, we could find that the confidence intervals of mlr is nearly [-20000,21000] and glm is nearly [7,8] with gamma distribution is narrower than mlr. This means glm with gamma distribution is more stable than mlr.

(6) The AIC of mlr is 726471.6 after I use step() function to stepwise regression on mlr and the AIC of glm with gamma distribution is 702279. We could find that AIC of glm with gamma distribution is smaller than mlr, which means glm with gamma distribution is fits the data better.

Part II

3. Check for stationarity and seasonality of the time-series data. Explain how you have done this and include relevant graphs and numerical summaries.

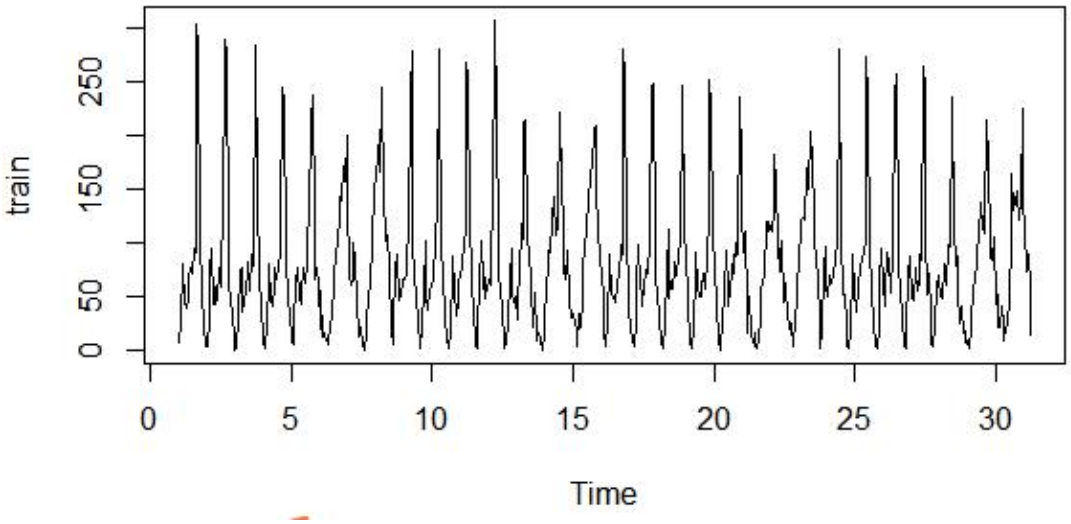
Stationarity means that the statistical properties of a time series (or rather the process generating it) do not change over time. Stationarity is important because many useful analytical tools and statistical tests and models rely on it.

Seasonality is a characteristic of a time series in which the data experiences regular and predictable changes that recur every calendar year. Any predictable fluctuation or pattern that recurs or repeats over a one-year period is said to be seasonal.

Before categorizing the data, I did some processing on the database. The time_id of original dataset is from 1 to 24, but as we all know, when time_id from 1 to 5, there is no passengers to take bus because buses stop running at that time. Therefore, I decide to delete time_id from 1 to 5, because these data are noises.

A. time series data (delete time_id from 1 to 5)

(1) pictures representing stationary of time series data (delete time_id from 1 to 5)

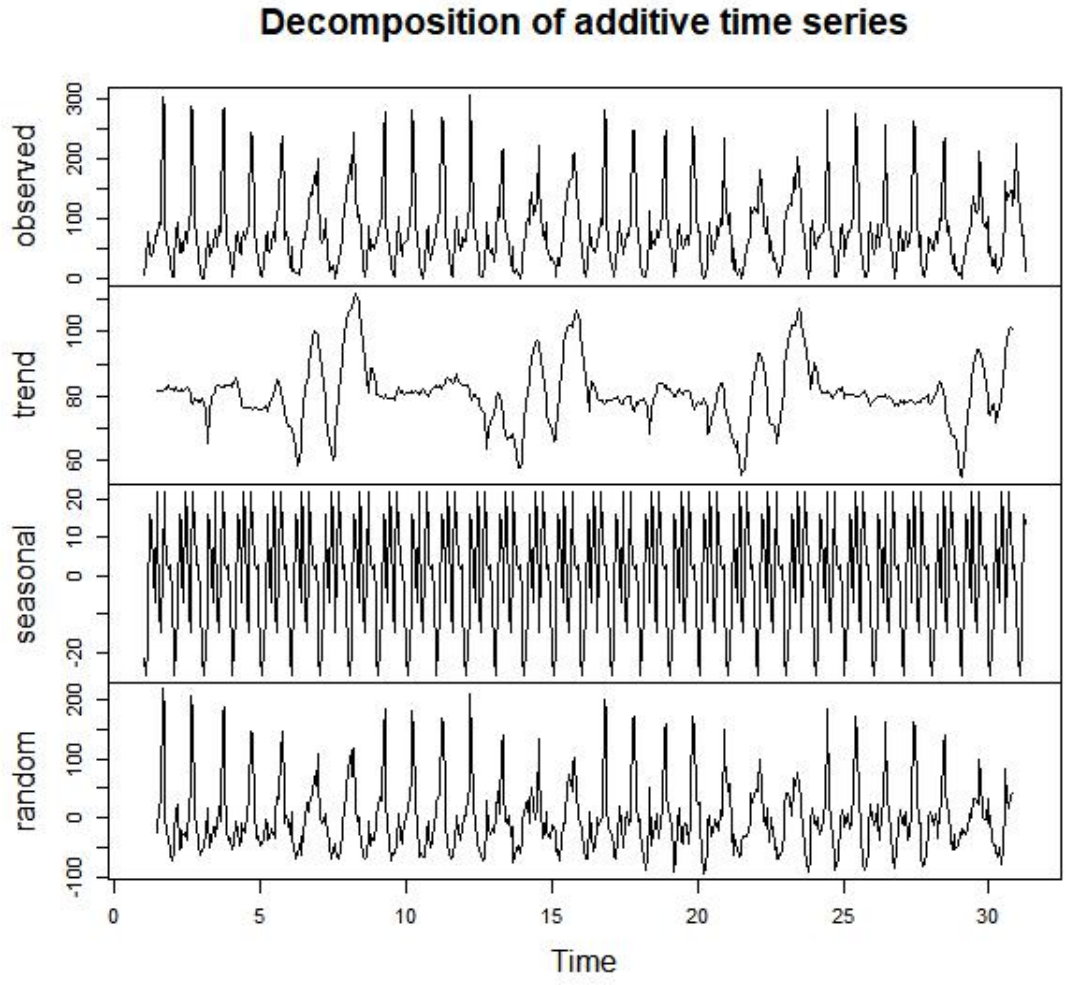


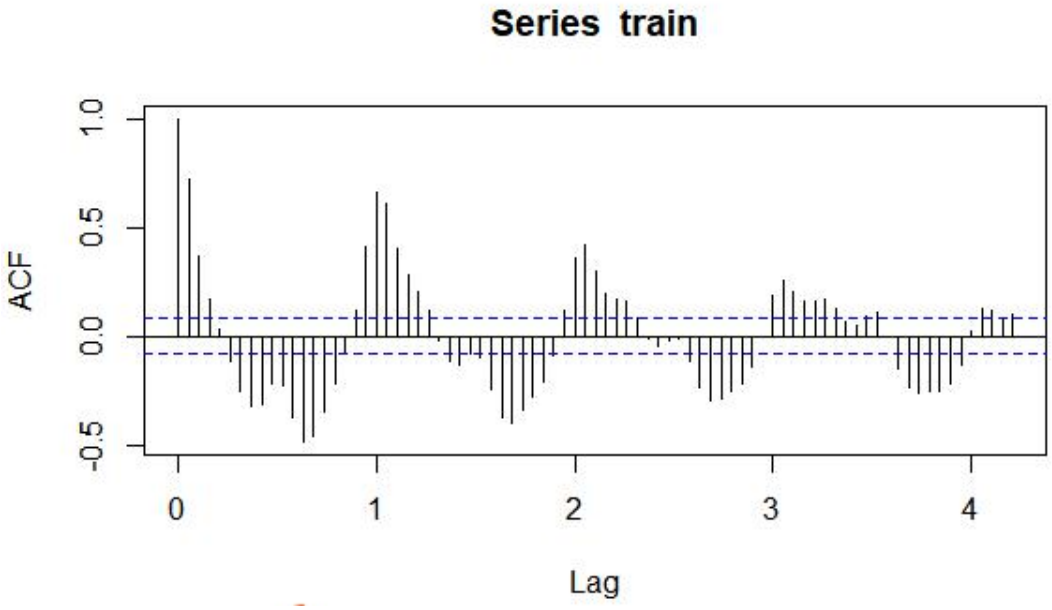
Because the stationary time series data usually shows a process of constant fluctuation around its mean on the graph, while the non-stationary time series data usually shows different means in different time periods (such as continuously rising or falling). Therefore, from the plot of time series data (delete time_id from 1 to 5) listed above, we could find out that it shows a process of constant fluctuation around its mean, which means this data is stationary.

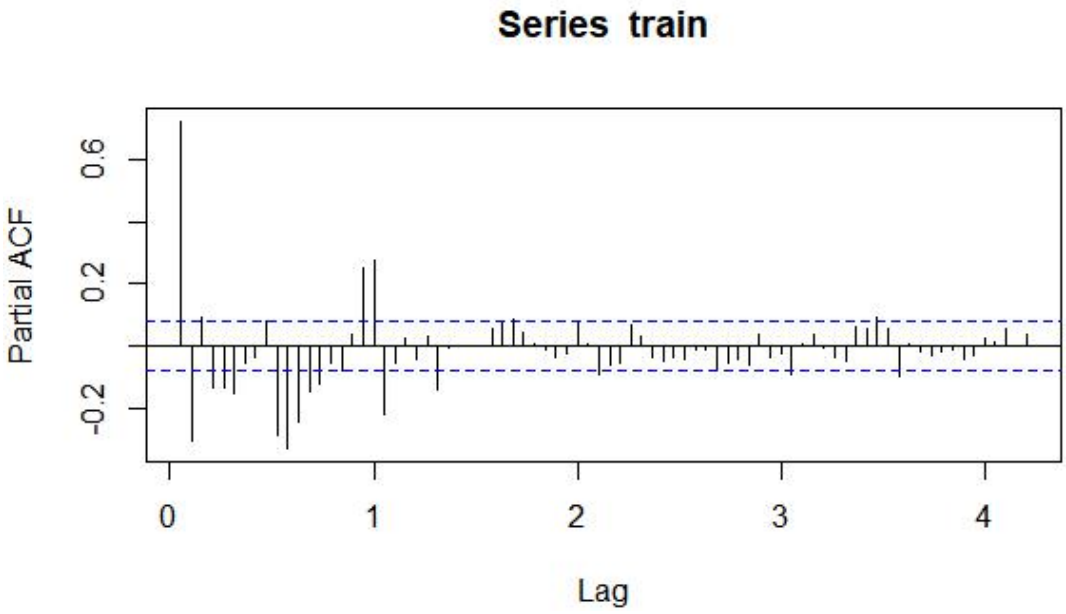
```
> adf.test(train)

        Augmented Dickey-Fuller Test

data:  train
Dickey-Fuller = -8.7128, Lag order = 8, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(train) : p-value smaller than printed p-value
```

I use adf.test() function to test if my time series data (delete time_id from 1 to 5) is value is stationary. From the snapshot listed above, I find that the p-value of these data is smaller than 0.01, which means that the time-series data is stationary. Besides, the alternative hypothesis is also listed by adf.test() funtion, which is represent stationary.

(2) pictures representing seasonality of time series data (delete time_id from 1 to 5)


Decomposition of additive time series
observed
trend
seasonal
random
Time


Series train
ACF
Lag

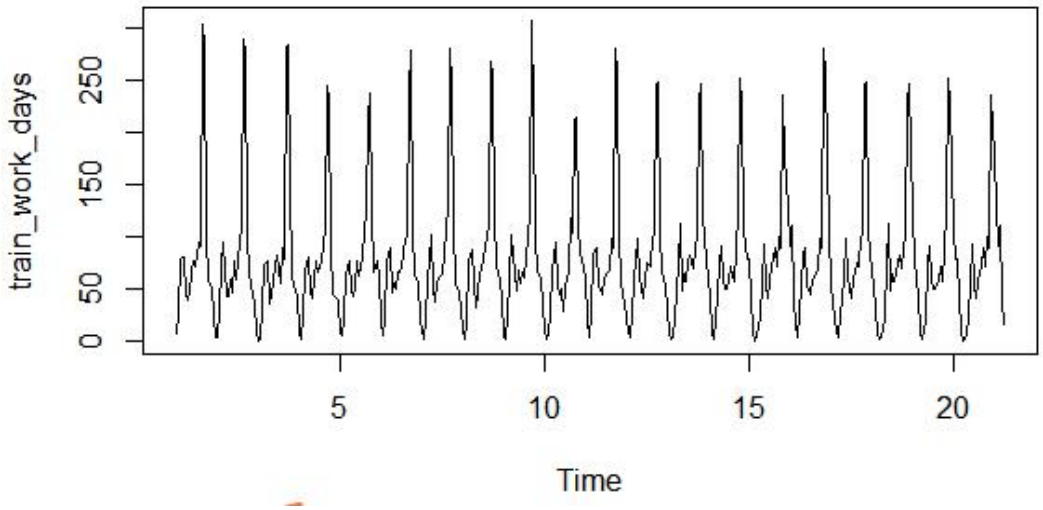


Stationary data is not trend, there is no seasonality; Its mean has a constant amplitude on the time axis, and its variance tends to the same stable value on the time axis.

However from the plot of decomposition of time series data(delete time_id from 1 to 5), we could find out that this data has seasonality, because the curve of seasonal repeats over a specific period. Besides, from the plots of acf and pacf, we could find out that there are spikes at lag 0,1,2,3,4 in the ACF and a single significant spike at lag 0 in the PACF. These also means that this data has seasonality.

B. time series data (delete weekend)

(1) pictures representing stationary of time series data(delete weekend)

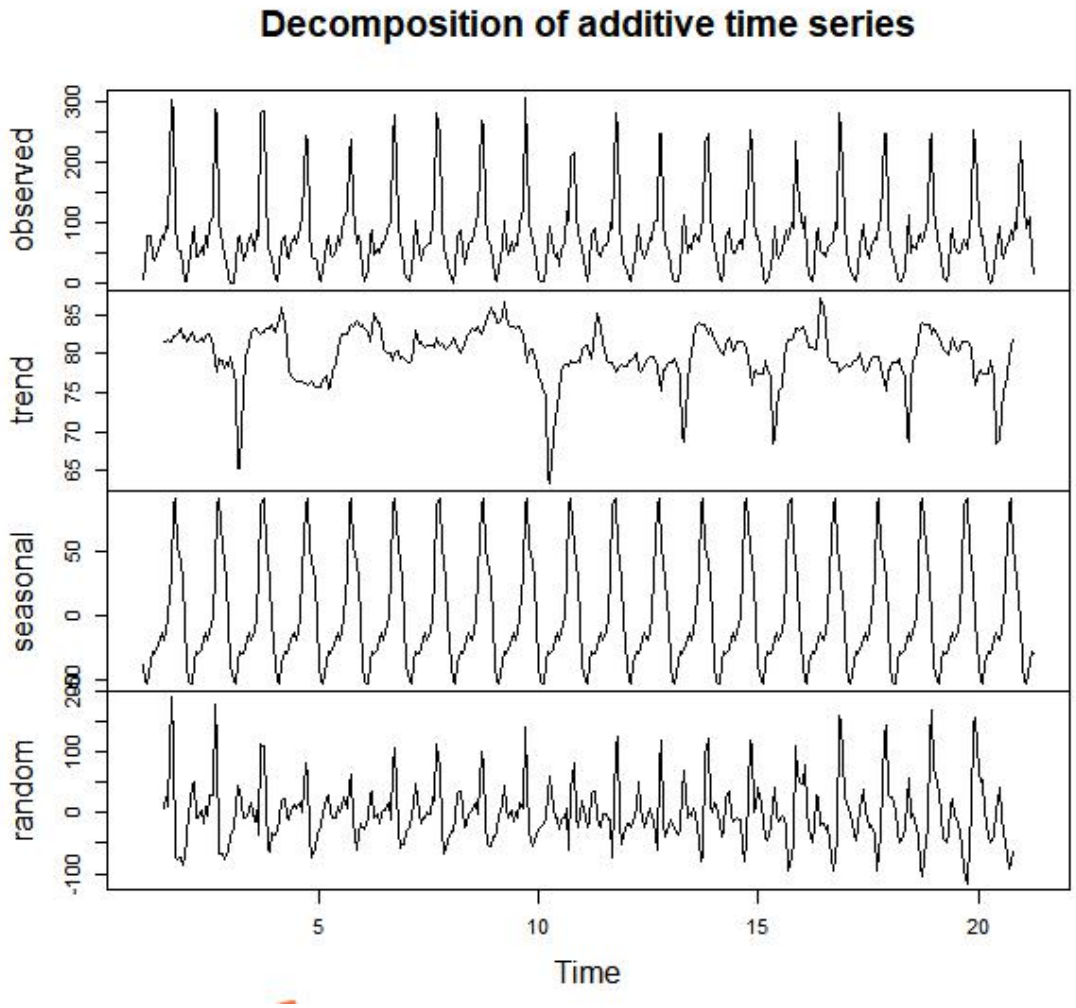


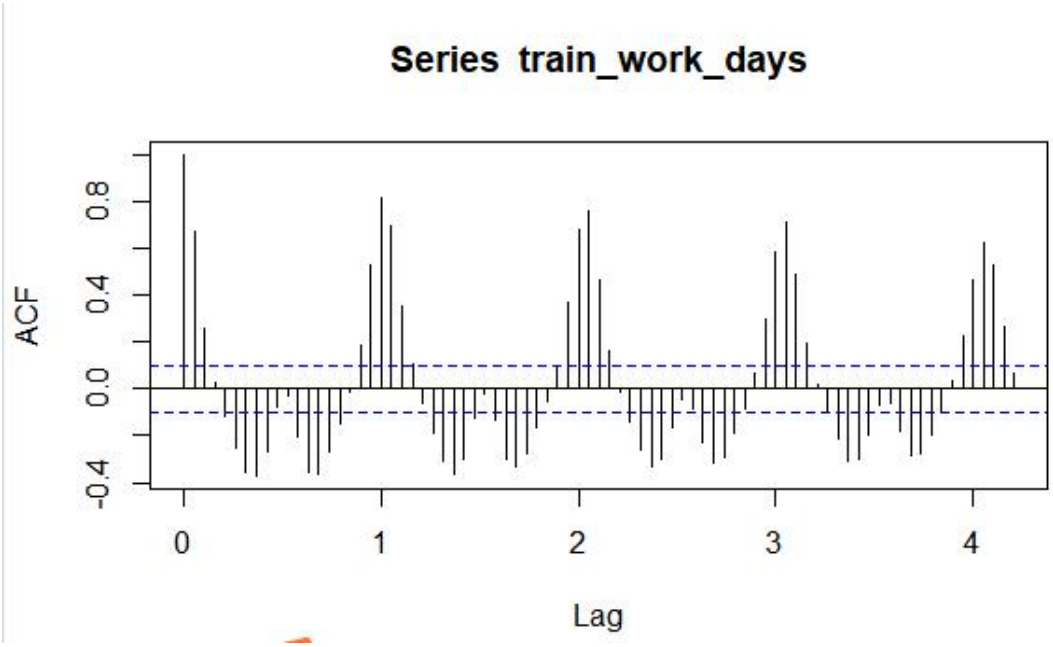
Because the stationary time series data usually shows a process of constant fluctuation around its mean on the graph, while the non-stationary time series data usually shows different means in different time periods (such as continuously rising or falling). Therefore, from the plot of time series data (delete weekend) listed above, we could find out that it shows a process of constant fluctuation around its mean, which means this data is stationary.

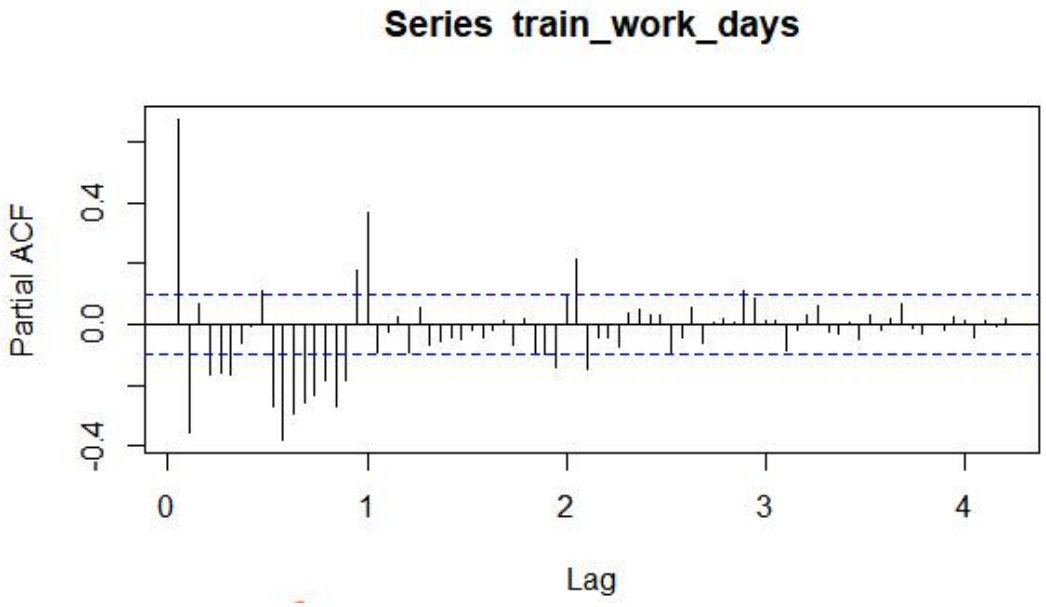
```
> #find stationary
> adf.test(train_work_days)

        Augmented Dickey-Fuller Test

data:  train_work_days
Dickey-Fuller = -8.8067, Lag order = 7, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(train_work_days) : p-value smaller than printed p-value
```

I use adf.test() function to test if my time series data (delete weekend) is value is stationary. From the snapshot listed above, I find that the p-value of these data is smaller than 0.01, which means that the time-series data is stationary. Besides, the alternative hypothesis is also listed by adf.test() funtion, which is represent stationary.

(2) pictures representing seasonality of time series data(delete weekend)


Decomposition of additive time series
observed
trend
seasonal
random
Time


Series train_work_days
ACF
Lag

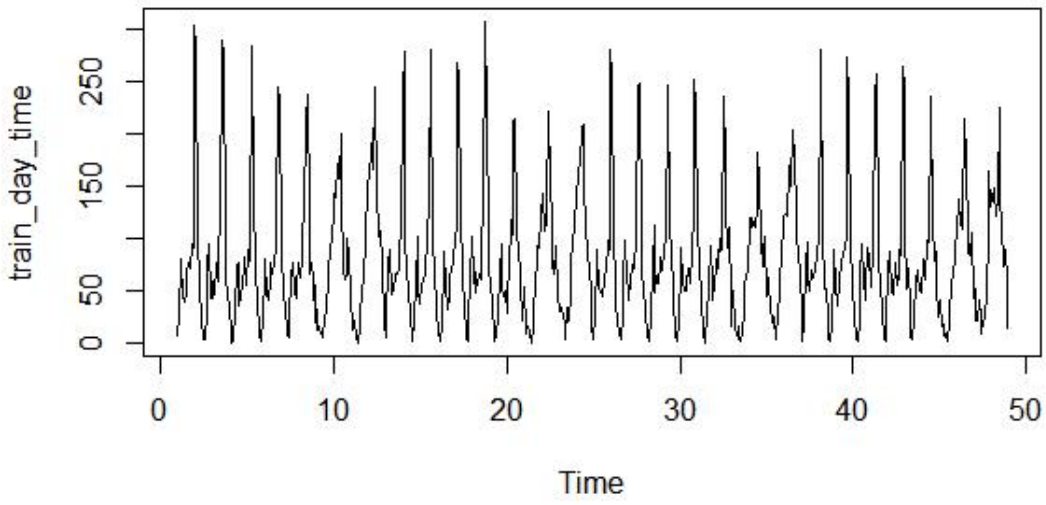


Stationary data is not trend, there is no seasonality; Its mean has a constant amplitude on the time axis, and its variance tends to the same stable value on the time axis.

However from the plot of decomposition of time series data(delete weekend), we could find out that this data has seasonality, because the curve of seasonal repeats over a specific period. Besides, from the plots of acf and pacf, we could find out that there are spikes at lag 0,1,2,3,4 in the ACF and a single significant spike at lag 0 in the PACF. These also means that this data has seasonality.

C. time series data (just take time_id from 6 to 18)

(1) pictures representing stationary of time series data (just take time_id from 6 to 18)

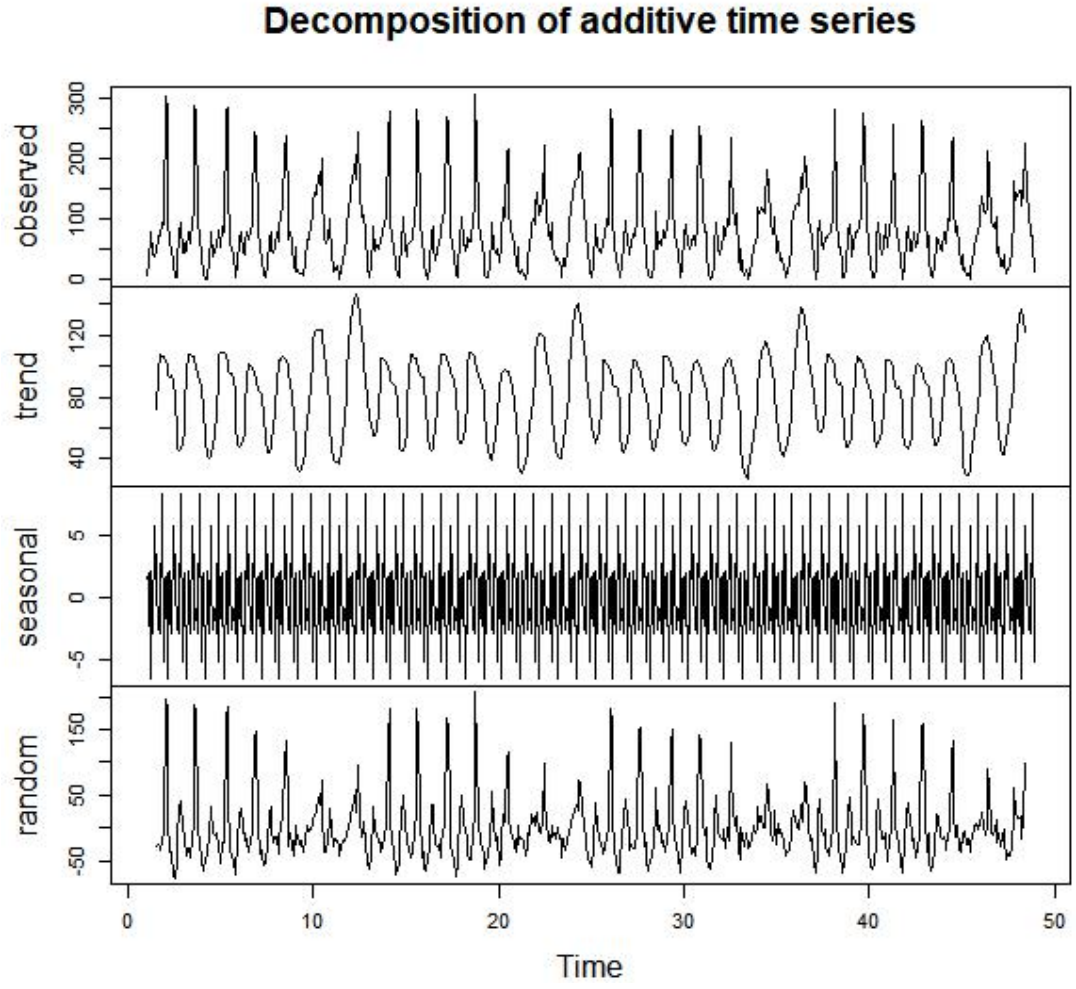


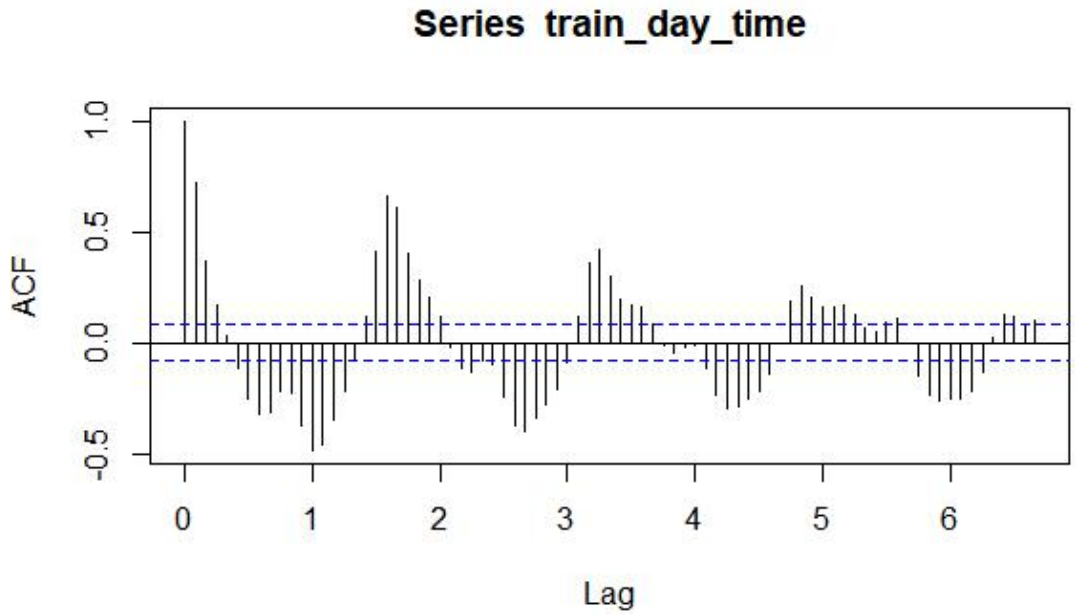
Because the stationary time series data usually shows a process of constant fluctuation around its mean on the graph, while the non-stationary time series data usually shows different means in different time periods (such as continuously rising or falling). Therefore, from the plot of time series data (delete weekend) listed above, we could find out that it shows a process of constant fluctuation around its mean, which means this data is stationary.

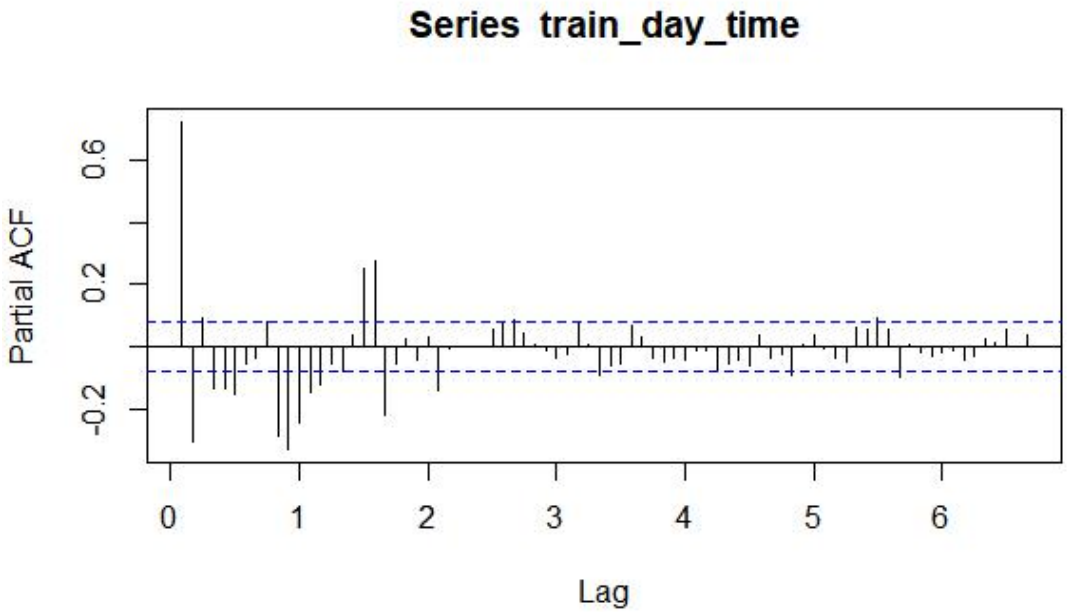
```
> adf.test(train_day_time)

        Augmented Dickey-Fuller Test

data:  train_day_time
Dickey-Fuller = -8.7128, Lag order = 8, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(train_day_time) : p-value smaller than printed p-value
```

I use adf.test() function to test if my time series data (just take time_id from 6 to 18) is value is stationary. From the snapshot listed above, I find that the p-value of these data is smaller than 0.01, which means that the time-series data is stationary. Besides, the alternative hypothesis is also listed by adf.test() funtion, which is represent stationary.

(2) pictures representing seasonality of time series data (just take time_id from 6 to 18)


Decomposition of additive time series
observed
trend
seasonal
random
Time


Series train_day_time
ACF
Lag



Stationary data is not trend, there is no seasonality; Its mean has a constant amplitude on the time axis, and its variance tends to the same stable value on the time axis.

However from the plot of decomposition of time series data(just take time_id from 6 to 18), we could find out that this data has seasonality, because the curve of seasonal repeats over a specific period. Besides, from the plots of acf and pacf, we could find out that there are spikes at lag 0,1,2,3,4 in the ACF and a single significant spike at lag 0 in the PACF. These also means that this data has seasonality.

4. Choose and detail each type of model used (including mathematical form and explanation of notation) and some details of how it was fitted. Explain why each model may be suitable for this type of data.

(1) I use the aggregate() function to build the first model(without a stochastic model), which is based entirely on reasonable summaries from past passengers flow, the formulas of this data are listed below:

$$\text{Predict\_mean}=\left(\sum_{i=1}^{amount.days} daily.passengers\right)/\text{amount.days}$$

$$\text{Predict\_mean(except weekend)} =\left(\sum_{i=1}^{amount.days} daily.passengers\right)/\text{amount.days}$$

This first formula is fit for time series data (just take time_id from 6 to 18) and time series data (delete time_id from 1 to 5).

This second formula is fit for time series data(delete weekend).

(2) In terms of building the second one(involve the fitting of a stochastic time series model), I choose to build seasonal arima model.

A time series {Xt} is called an autoregressive integrated moving average (ARIMA) process with order p, d, and q, denoted {Xt} ~ ARIMA(p, d, q), if its d-th order difference

$$Z_t = (1-B)^d X_t$$

is a stationary ARMA(p, q) process, where d ≥ 1 is an integer:

$$b(B)(1-B)^d X_t = a(B)\varepsilon_t$$

{Xt} is called ARIMA as it is the integration of a differenced series {Zt}. ARIMA processes are nonstationary. An ARIMA(p, d, q) model is an ARMA(p + d, q) model.

The extension formula of ARIMA is:

$$\hat{y}_t = \mu + \phi_1 * y_{t-1} + ... + \phi_p * y_{t-p} + \theta_1 * e_{t-1} + ... + \theta_q * e_{t-q}$$

ARMA models are not "natural" in most cases and so orders of AR(p) and MA(q) parts are not usually known (they are parameters too) and need to be estimated. There are statistical issues with estimating p and q. These arise largely from the situation that different hypotheses specifying the model give the same model.

Therefore, there are several approaches for order determination.

- Use the autocorrelation function (ACF).
- Use the partial autocorrelation function (PACF).
- Use information criteria such as AIC

The ARIMA(p, d, q) × (ps, ds, qs)s is a seasonal model, which has a process:

$$(1-\beta_1 B - ... - \beta_p B^P) \times (1-\beta_1^* B^s - ... - \beta_{p^s}^* (B^s)^{p_s}) \times (\delta^d (\delta_s^{d_s} X_t) - \mu$$

$$= \quad (1-\alpha_1 B - ... - \alpha_q B^q) \times (1-\alpha_1^* B^s - ... - \alpha_{q^s}^* (B^s)^{q_s}) \varepsilon_t$$

From the formula listed above, we could find that the parameters of seasonal ARIMA are p and d and q. The meanings of each parameter are listed below:

- p: auto-regressive (lags of y values) , which is choosed from partial autocorrelation function(PACF) according to the number of lagged forecast errors in the prediction equation.
- d: differencing order (Intergrated), which is according to the number of nonseasonal differences needed for stationarity.
- q: moving average (lags of predictive error), which is choosed from autocorrelation function (ACF) according to the number of autoregressive terms.

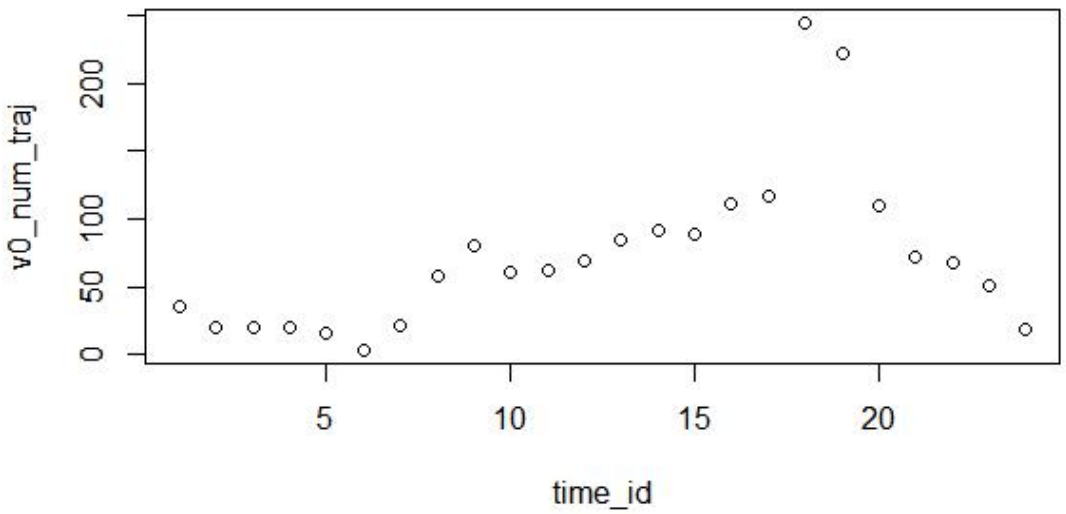
Beside, we could also use auto.arima() function to find suitable parameters p,d and q.

5. Report full details of each fitted model.

A. **time series data (delete time_id from 1 to 5)**

(1) The first type of model

```
> #the first type model
> pred_mean <- aggregate(v0_num_traj~time_id, data=flow, mean)
> pred_mean
   time_id v0_num_traj
1        1   35.083333
2        2   19.875000
3        3   20.125000
4        4   19.750000
5        5   15.250000
6        6    3.678571
7        7   22.035714
8        8   57.857143
9        9   79.678571
10      10   60.464286
11      11   61.964286
12      12   69.464286
13      13   83.928571
14      14   91.892857
15      15   89.392857
16      16  110.857143
17      17  116.964286
18      18  244.178571
19      19  221.321429
20      20  109.107143
21      21   71.607143
22      22   67.535714
23      23   50.535714
24      24   18.464286
```

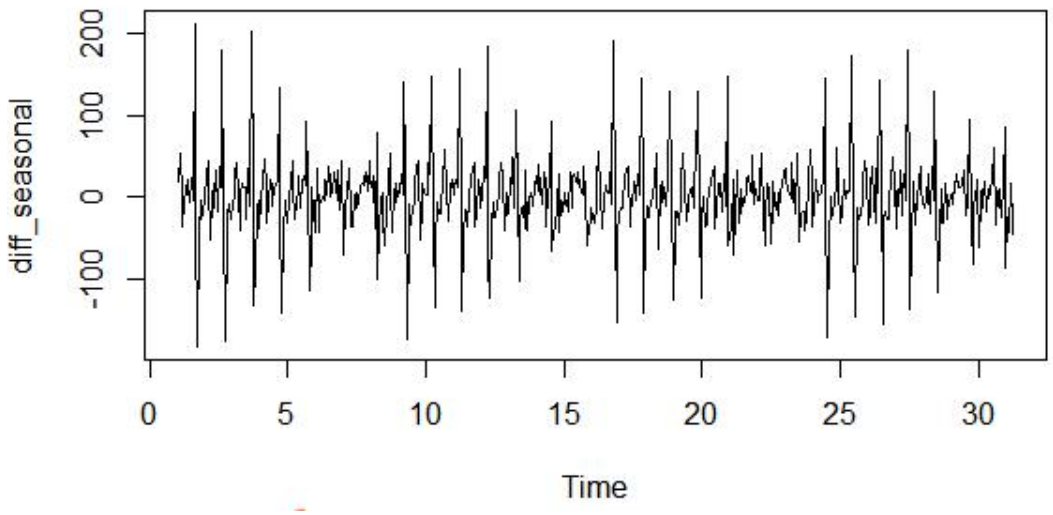


From the plot of first model of time series data (delete time_id from 1 to 5), we could find out that passenger flows are very high when time_id from 16 to 20 and at peak when time_id is 18. That make sense, because this range of time_id is the rush hour between work and school.

(2) The second type of model

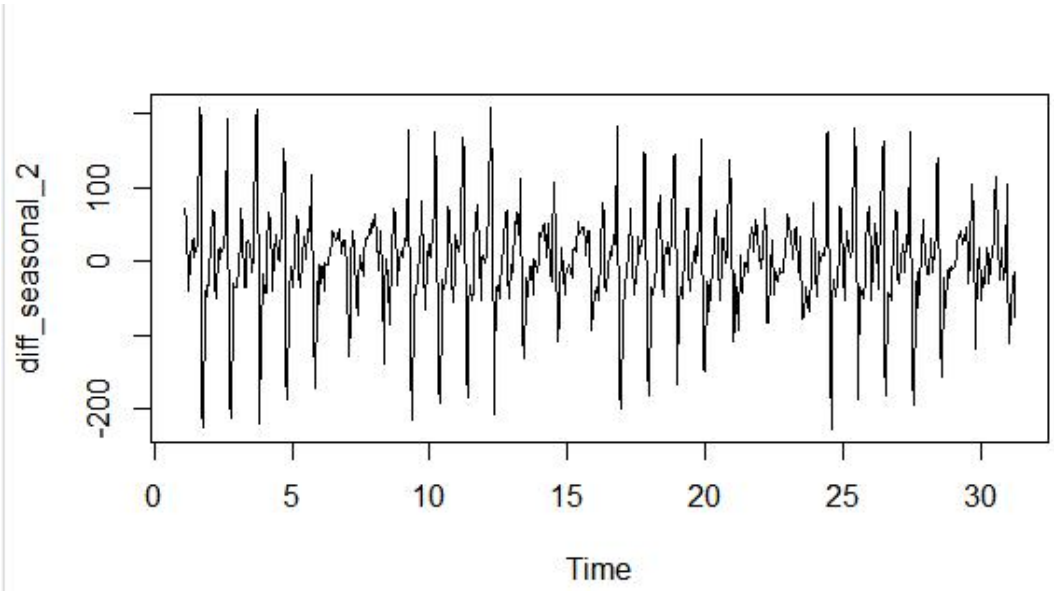
As I mentioned in question 3, time series data (delete time_id from 1 to 5) has seasonality, although it is stationary. Therefore, we should do difference on this data to find its parameters p,d and q.

(a) When differencing order (Intergrated) is equal to 1



The RMSE value of the time-series data after the first difference is 103.2569.

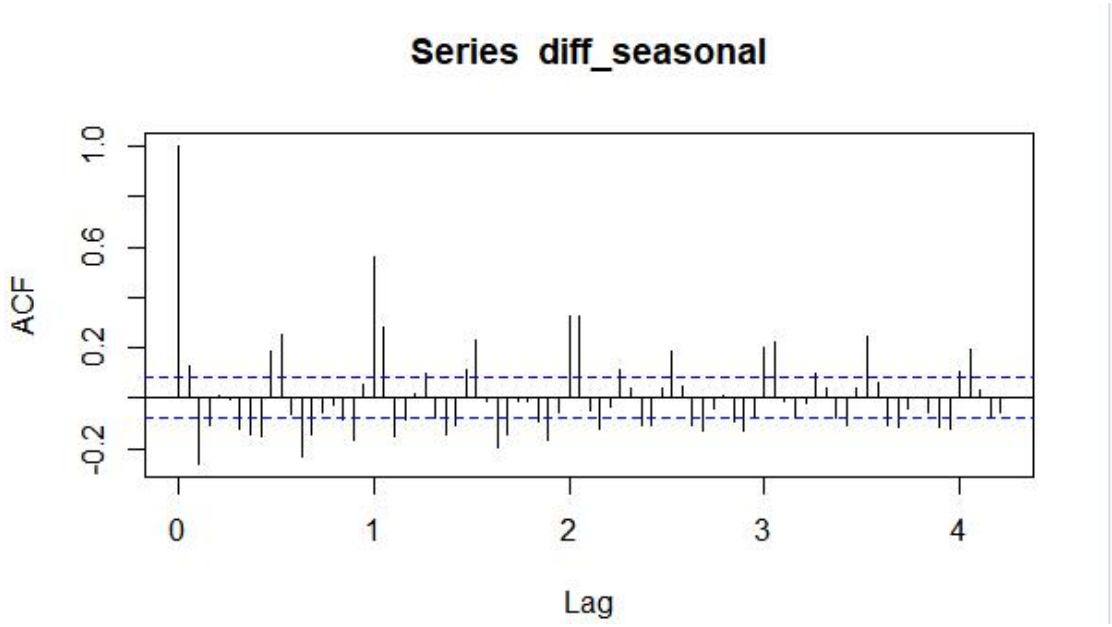
(b) When differencing order (Intergrated) is equal to 2

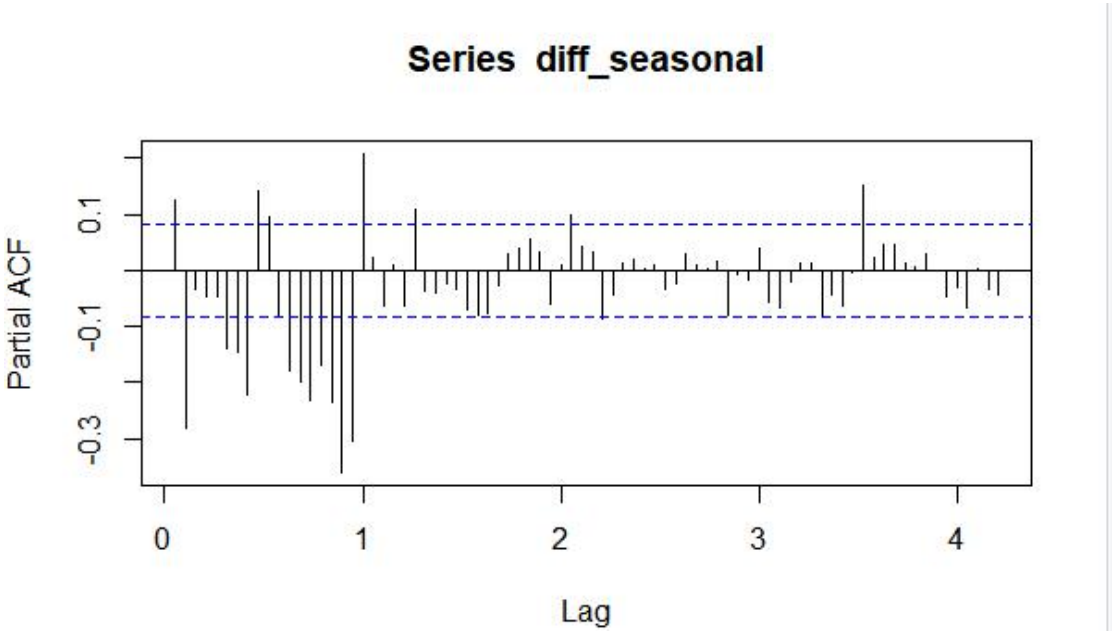


The RMSE value of the time-series data after the second difference is 103.3175

By comparing RMSE values of the time-series data with two different deferencing order, we could find out that the better time-series data is that after the first difference, which means d should equals to 1.

(c) After determining d equals to 1, we should use acf() and pacf() function to determine parameters p and q. The plots are listed below:





We could find that the suitable parameters p and q are 1 and 2 respectively, so I could find the suitable ARIMA model is ARIMA(1,1,2).

(d) And then I also use auto.arima() function to help me find out suitable paramters p,d and q of ARIMA. The snapshot of details is listed below:

```
> arima_auto
Series: train
ARIMA(1,0,1)(1,0,0)[19] with non-zero mean

Coefficients:
         ar1     ma1    sar1     mean
      0.5209  0.2483  0.5913  81.7591
s.e.  0.0529  0.0620  0.0356   8.7726

sigma^2 estimated as 1208:  log likelihood=-2863.44
AIC=5736.88   AICc=5736.99   BIC=5758.66
```

We could find out that ARIMA model is good when p and q both equal to 1 and d equals to 0. This means that the good ARIMA model is ARIMA(1,0,1)(1,0,0)

And then I use Arima() function and take (1,0,1)(1,0,0) into it to deal with my time series data, the details of ARIMA(1,0,1)(1,0,0) model is listed below:

```
> arima_auto_fit <- Arima(train,order = c(1,0,1),seasonal=c(1,0,0))
> arima_auto_fit
Series: train
ARIMA(1,0,1)(1,0,0)[19] with non-zero mean

Coefficients:
         ar1     ma1    sar1     mean
      0.5209  0.2483  0.5913  81.7591
s.e.  0.0529  0.0620  0.0356   8.7726

sigma^2 estimated as 1208:  log likelihood=-2863.44
AIC=5736.88   AICc=5736.99   BIC=5758.66
```

Besides, I also use Arima() function and take (1,1,2)(1,0,0), which is found out from pictures of ACF and PACF of the time series data after first difference, into it to deal with my time series data, the details of ARIMA(1,1,2)(1,0,0) model is listed below:

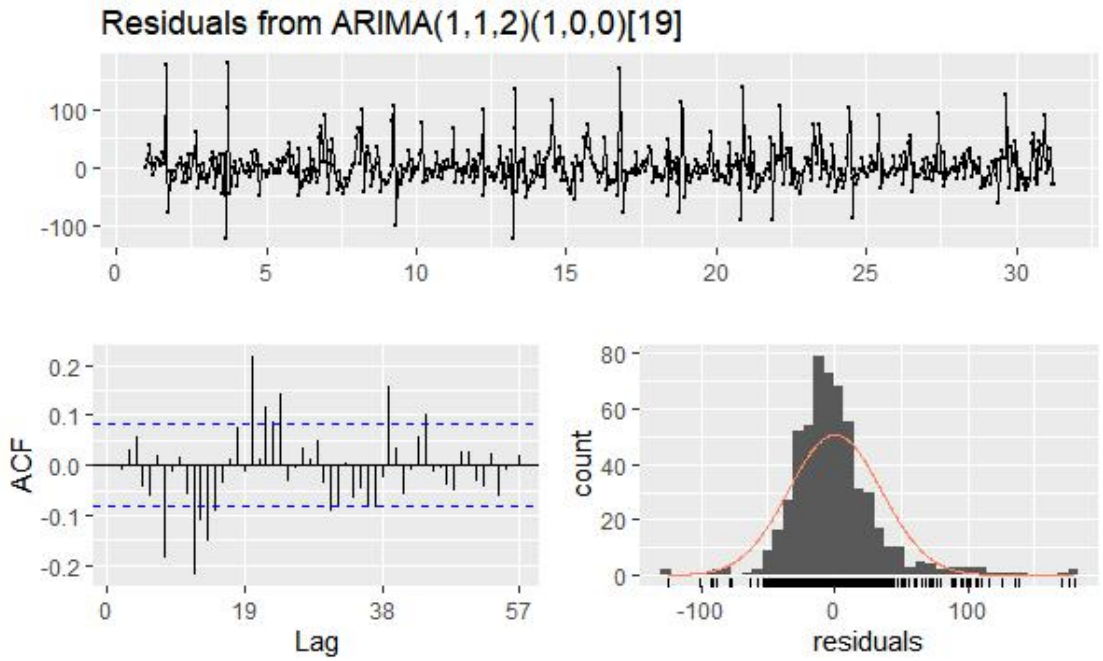
```
> arima_fit <- Arima(train,order = c(1,1,2),seasonal=c(1,0,0))
> arima_fit
Series: train
ARIMA(1,1,2)(1,0,0)[19]

Coefficients:
         ar1      ma1      ma2    sar1
      0.5248  -0.7544  -0.2456  0.5937
s.e.  0.0530   0.0623   0.0622  0.0357

sigma^2 estimated as 1209:  log likelihood=-2860.34
AIC=5730.69   AICc=5730.8   BIC=5752.46
```

AIC works on the principle of “the smaller, the better”,with a penalty for increasing the number of parameters and based on treating them as asymptotic χ^2 statistics.

Therefore, compared AIC of ARIMA(1,1,2)(1,0,0) with ARIMA(1,0,1)(1,0,0), we could find that AIC of ARIMA(1,1,2)(1,0,0) is smaller, which means ARIMA(1,1,2)(1,0,0) is better.

Finally, I checked the residuals of the model, the details is listed below:

```
> checkresiduals(arima_fit)

        Ljung-Box test

data:  Residuals from ARIMA(1,1,2)(1,0,0)[19]
Q* = 167.15, df = 34, p-value < 2.2e-16

Model df: 4.   Total lags used: 38
```

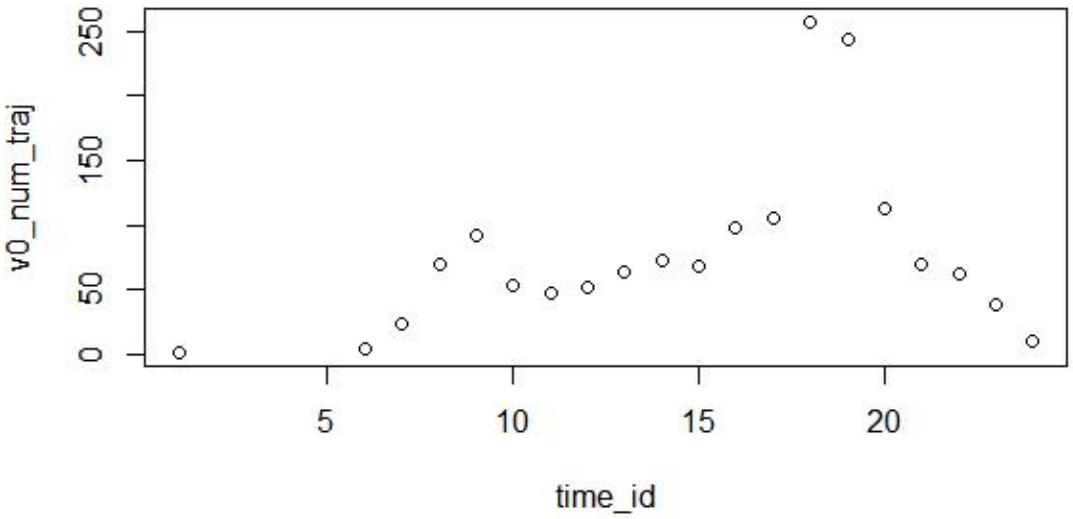


We could find that the p-value here is relatively small, which means the model is good enough.

B. time series data (delete weekend)

(1) The first type of model

```
> #the first type model of work days data
> pred_mean_work <- aggregate(v0_num_traj~time_id, data=flow_work, mean)
> pred_mean_work
   time_id v0_num_traj
1        1        2.00
2        6        3.95
3        7       24.10
4        8       69.65
5        9       92.70
6       10       52.85
7       11       47.40
8       12       52.60
9       13       63.30
10      14       72.25
11      15       68.45
12      16       97.75
13      17      105.05
14      18      256.65
15      19      242.70
16      20      112.50
17      21       69.30
18      22       62.25
19      23       37.95
20      24       11.05
```

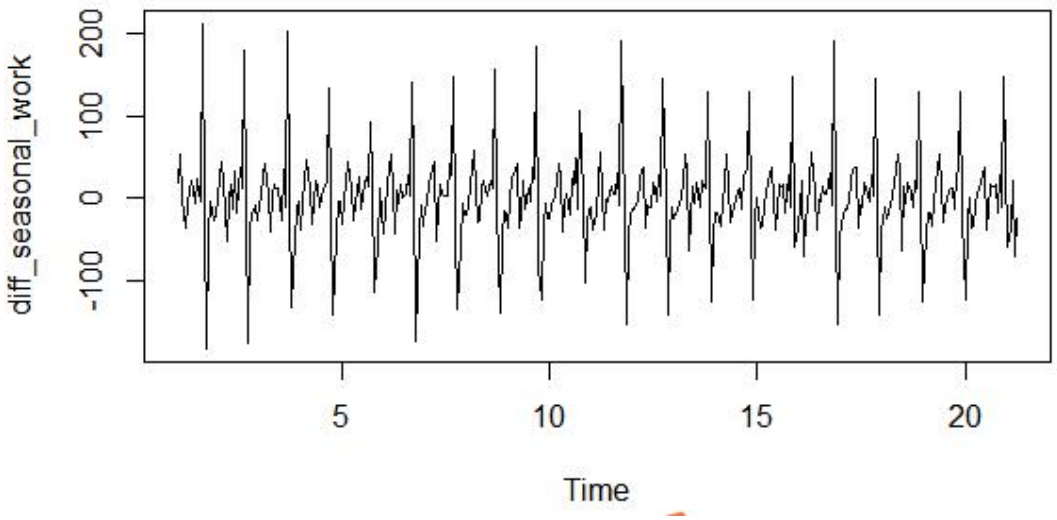


From the plot of first model of time series data (delete weekend), we could find out that passenger flows are very high when time_id from 13 to 16 and at peak when time_id is 14. This doesn't correspond with the actual situation , because this range of time_id is not the rush hour between work and school.

(2) The second type of model

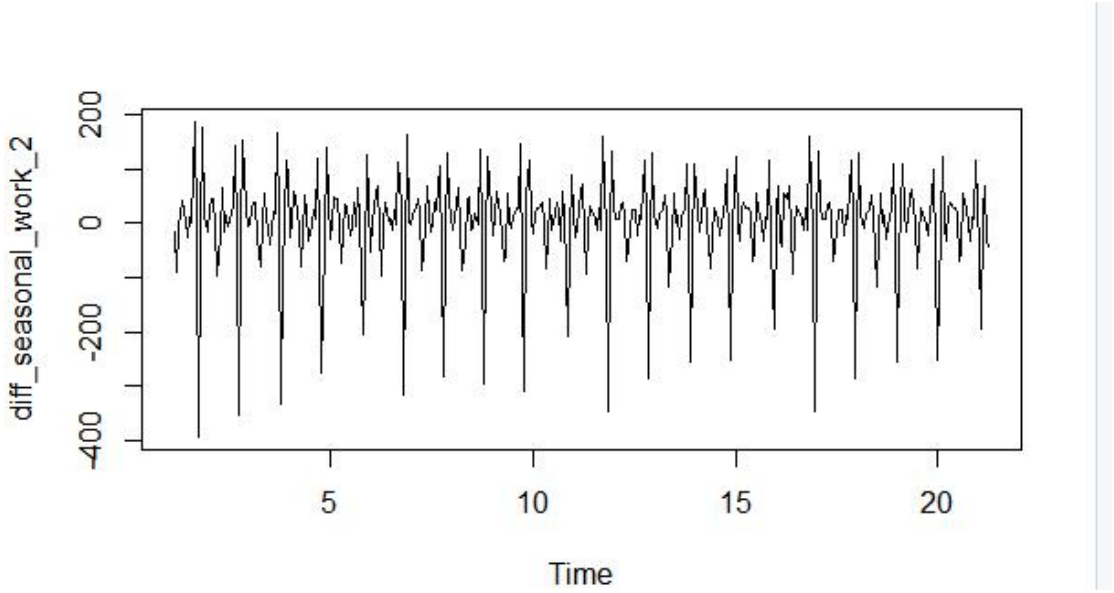
As I mentioned in question 3, time series data (delete weekend) has seasonality, although it is stationary. Therefore, we should do difference on this data to find its parameters p,d and q.

(a) When differencing order (Intergrated) is equal to 1



The RMSE value of the time-series data after the first difference is 103.7794.

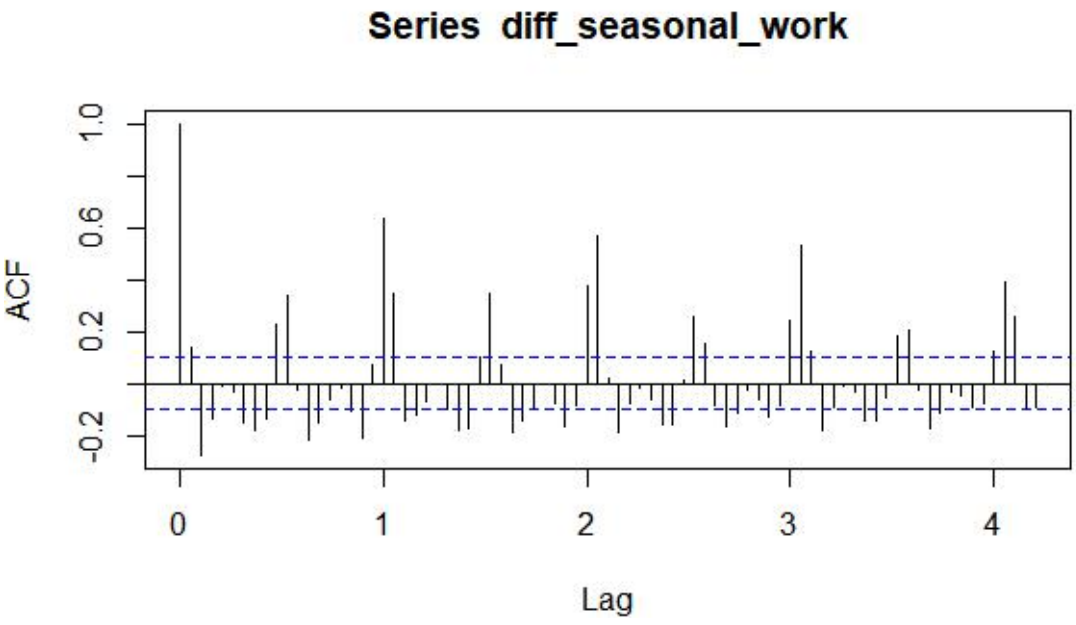
(b) When differencing order (Intergrated) is equal to 2

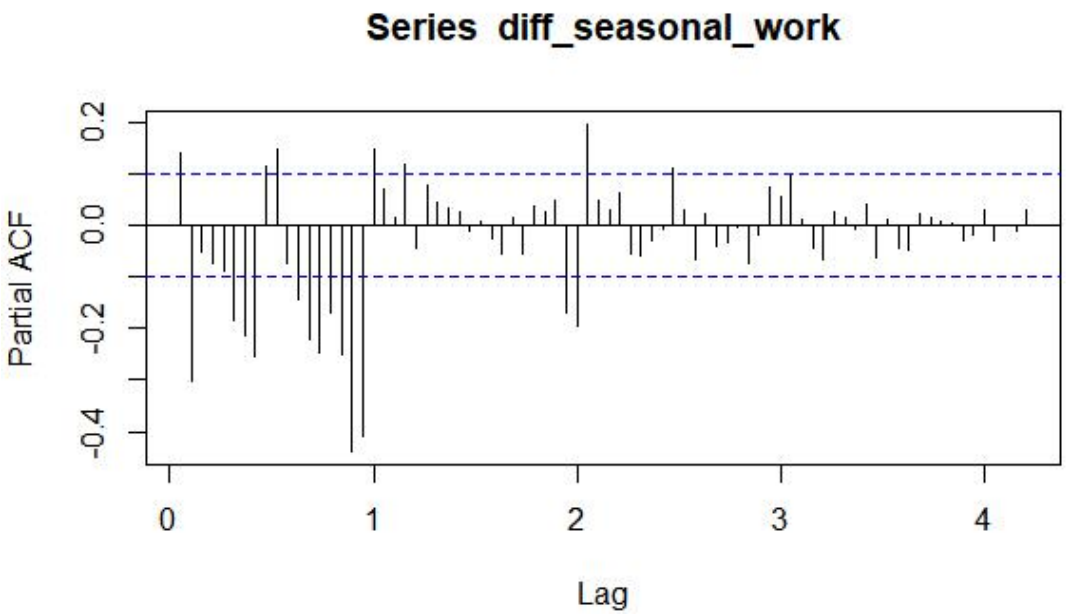


The RMSE value of the time-series data after the second difference is 131.4262.

By comparing RMSE values of the time-series data with two different deferencing order, we could find out that the better time-series data is that after the first difference, which means d should equals to 1.

(c) After determining d equals to 1, we should use acf() and pacf() function to determine parameters p and q. The plots are listed below:





We could find that the suitable parameters p and q are 1 and 2 respectively, so I could find the suitable ARIMA model is ARIMA(1,1,2).

(d) And then I also use auto.arima() function to help me find out suitable paramters p,d and q of ARIMA. The snapshot of details is listed below:

```
> arima_auto
Series: train_work_days
ARIMA(1,0,1)(0,1,1)[19]

Coefficients:
          ar1     ma1     sma1
      -0.1896  0.5084  -0.1749
s.e.   0.1318  0.1144   0.0580

sigma^2 estimated as 1067:  log likelihood=-1799
AIC=3605.99   AICc=3606.1   BIC=3621.61
```

We could find out that ARIMA model is good when p and q both equal to 1 and d equals to 0. This means that the good ARIMA model is ARIMA(1,0,1)(0,1,1).

And then I use Arima() function and take (1,0,1)(0,1,1) into it to deal with my time series data, the details of ARIMA(1,0,1)(0,1,1) model is listed below:

```
> arima_auto_fit_work <- Arima(train_work_days,order = c(1,0,1),seasonal = c(0,1,1))
> arima_auto_fit_work
Series: train_work_days
ARIMA(1,0,1)(0,1,1)[19]

Coefficients:
          ar1     ma1     sma1
      -0.1896  0.5084  -0.1749
s.e.   0.1318  0.1144   0.0580

sigma^2 estimated as 1067:  log likelihood=-1799
AIC=3605.99   AICc=3606.1   BIC=3621.61
```

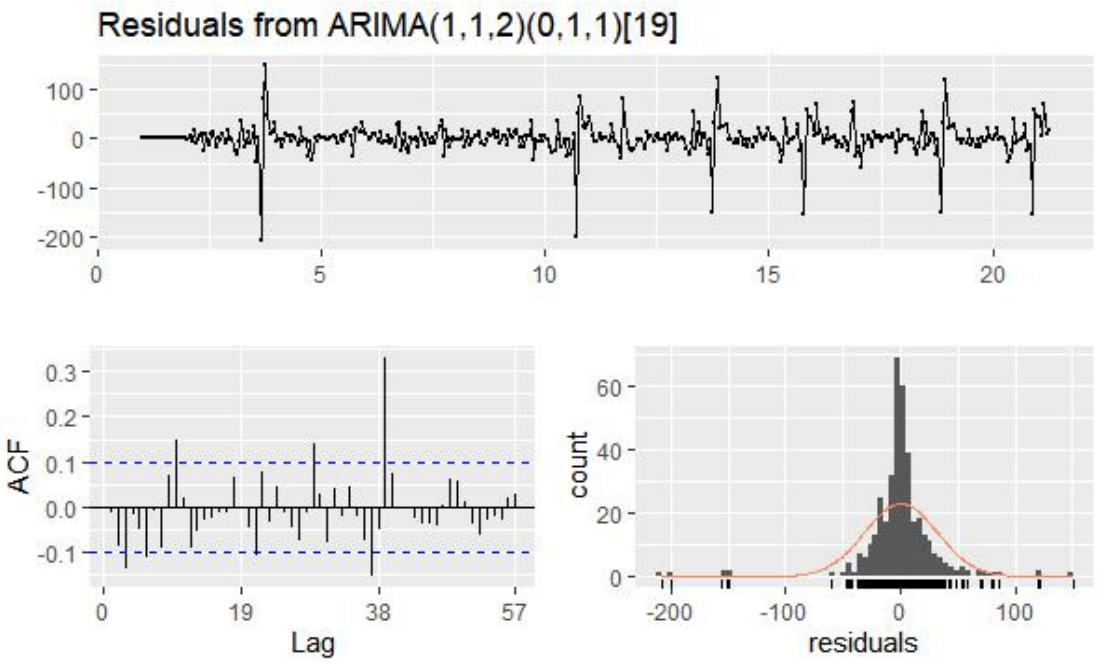
Besides, I also use Arima() function and take (1,1,2)(0,1,1), which is found out from pictures of ACF and PACF of the time series data after first difference, into it to deal with my time series data, the details of ARIMA(1,1,2)(0,1,1) model is listed below:

```
> arima_fit_work <- Arima(train_work_days,order = c(1,1,2),seasonal = c(0,1,1))
> arima_fit_work
Series: train_work_days
ARIMA(1,1,2)(0,1,1)[19]

Coefficients:
          ar1      ma1      ma2     sma1
      -0.1835  -0.4956  -0.5044  -0.171
s.e.   0.1317   0.1146   0.1144   0.058

sigma^2 estimated as 1073:  log likelihood=-1797.49
AIC=3604.98   AICc=3605.14   BIC=3624.49
```

Compared AIC of ARIMA(1,1,2)(0,1,1) with ARIMA(1,0,1)(0,1,1), we could find that AIC of ARIMA(1,1,2)(0,1,1) is smaller, which means ARIMA(1,1,2)(0,1,1) is better.

Finally, I checked the residuals of the model, the details is listed below:

```
> checkresiduals(arima_fit_work)

        Ljung-Box test

data:  Residuals from ARIMA(1,1,2)(0,1,1)[19]
Q* = 76.183, df = 34, p-value = 4.508e-05

Model df: 4.   Total lags used: 38
```

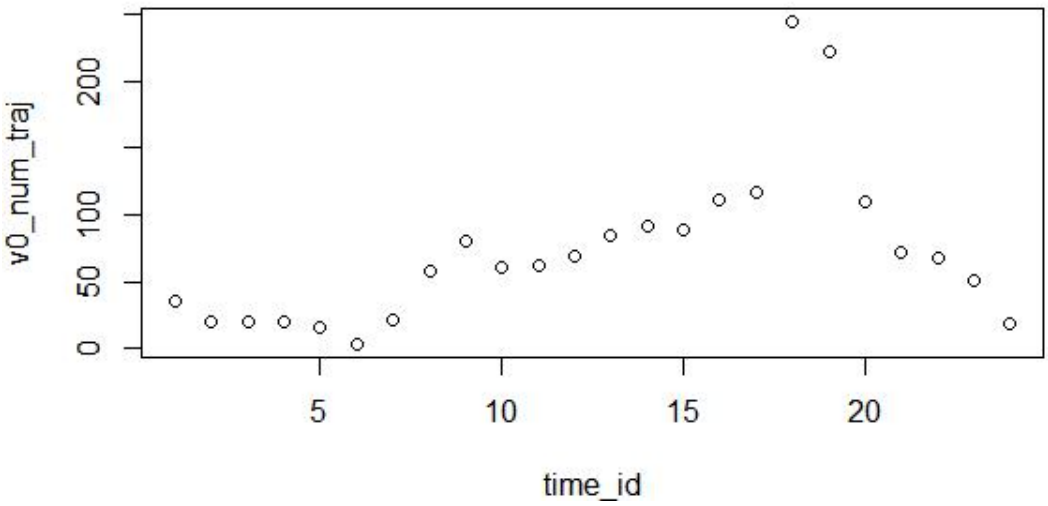


We could find that the p-value here is relatively small, which means the model is good enough.

C. time series data (just take time_id from 6 to 18)

(1) The first type of model

```
> pred_mean_daytime <- aggregate(v0_num_traj~time_id, data=flow, mean)
> pred_mean_daytime
   time_id v0_num_traj
1        1   35.083333
2        2   19.875000
3        3   20.125000
4        4   19.750000
5        5   15.250000
6        6    3.678571
7        7   22.035714
8        8   57.857143
9        9   79.678571
10      10   60.464286
11      11   61.964286
12      12   69.464286
13      13   83.928571
14      14   91.892857
15      15   89.392857
16      16  110.857143
17      17  116.964286
18      18  244.178571
19      19  221.321429
20      20  109.107143
21      21   71.607143
22      22   67.535714
23      23   50.535714
24      24   18.464286
```



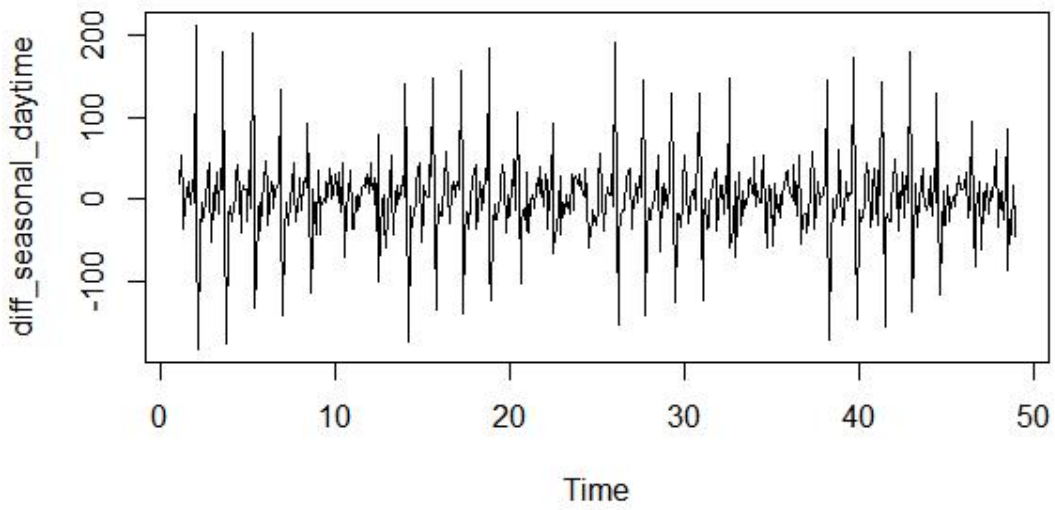
From the plot of first model of time series data (delete weekend), we could find out that passenger flows are very high when time_id from 16 to 18 and at peak when time_id is 18. This make sense, because this range of time_id is the rush hour between work and school.

(3) The second type of model

As I mentioned in question 3, time series data (just take time_id from 6 to 18) has

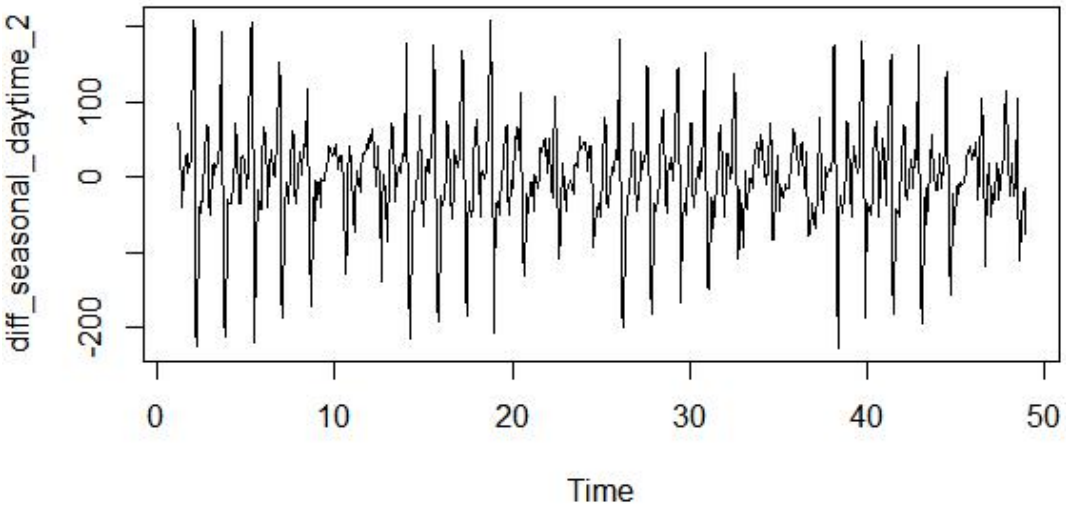
seasonality, although it is stationary. Therefore, we should do difference on this data to find its parameters p,d and q.

(e) When differencing order (Intergrated) is equal to 1



The RMSE value of the time-series data after the first difference is 103.2569.

(f) When differencing order (Intergrated) is equal to 2

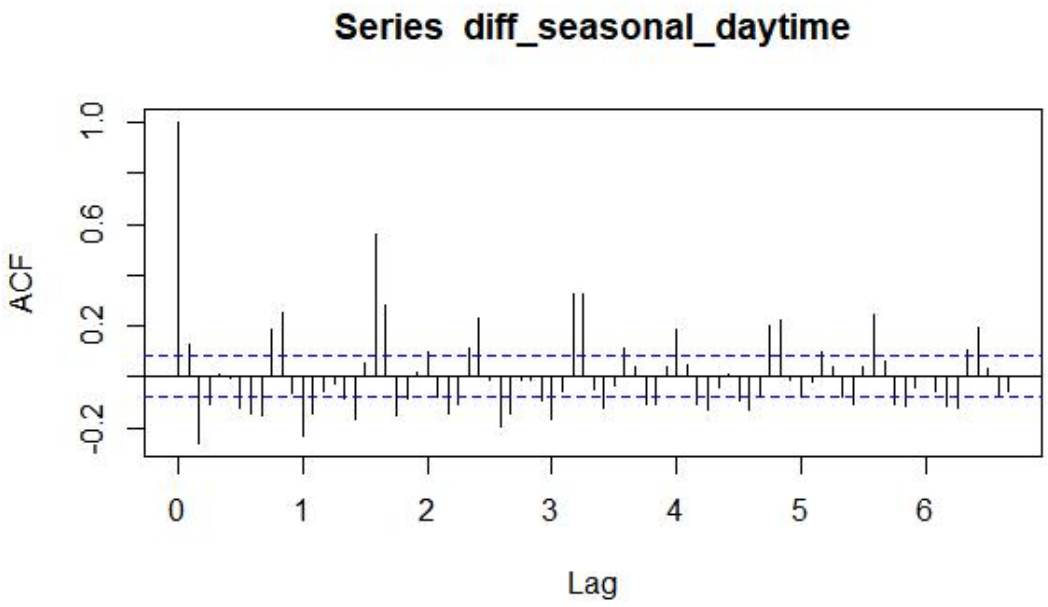


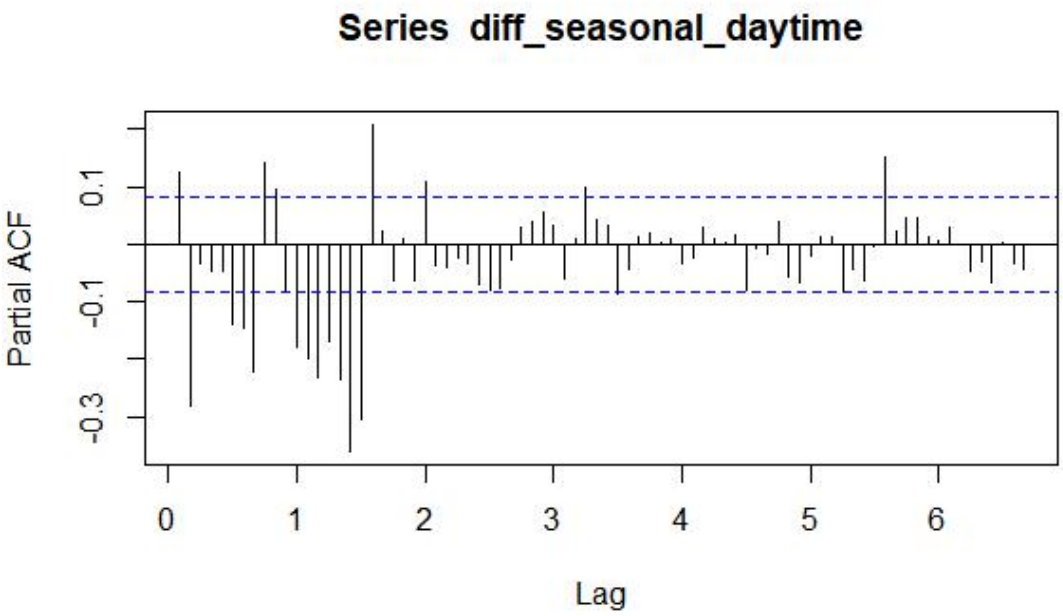
The RMSE value of the time-series data after the second difference is 103.3175.

By comparing RMSE values of the time-series data with two different deferencing order, we could find out that the better time-series data is that after the first difference, which means d should equals to 1.

(g) After determining d equals to 1, we should use acf() and pacf() function to

determine parameters p and q. The plots are listed below:





We could find that the suitable parameters p and q are 1 and 2 respectively, so I could find the suitable ARIMA model is ARIMA(1,1,2).

(h) And then I also use auto.arima() function to help me find out suitable paramters p,d and q of ARIMA. The snapshot of details is listed below:

```
> arima_auto_daytime
Series: train_day_time
ARIMA(4,0,5)(2,0,1)[12] with non-zero mean

Coefficients:
          ar1     ar2      ar3      ar4     ma1      ma2     ma3     ma4     ma5
      -0.0090  0.4892  -0.1570  -0.7784  0.8117  -0.1492  0.0078  0.9724  0.6401
s.e.   0.0778  0.0750   0.0404   0.0618  0.0622   0.0432  0.0589  0.0540  0.0354
         sar1    sar2    sma1      mean
      -0.3862  0.1144  0.0103  80.7990
s.e.   0.2379  0.1123  0.2353   2.7278

sigma^2 estimated as 1361:  log likelihood=-2891.28
AIC=5810.55   AICc=5811.3   BIC=5871.54
```

We could find out that ARIMA model is good when p equals to 4 and d equals to 0 and q equals to 5. This means that the good ARIMA model is ARIMA(4,0,5)(2,0,1).

And then I use Arima() function and take (4,0,5)(2,0,1) into it to deal with my time series data, the details of ARIMA(4,0,5)(2,0,1) model is listed below:

```
> # p=1, d=1, q=2
> arima_fit_daytime <- Arima(train_day_time,order = c(1,1,2))
> arima_fit_daytime
Series: train_day_time
ARIMA(1,1,2)

Coefficients:
         ar1      ma1      ma2
      0.5358  -0.5527  -0.4473
s.e.  0.0448   0.0477   0.0475

sigma^2 estimated as 1752:  log likelihood=-2964.08
AIC=5936.15   AICc=5936.22   BIC=5953.57
```

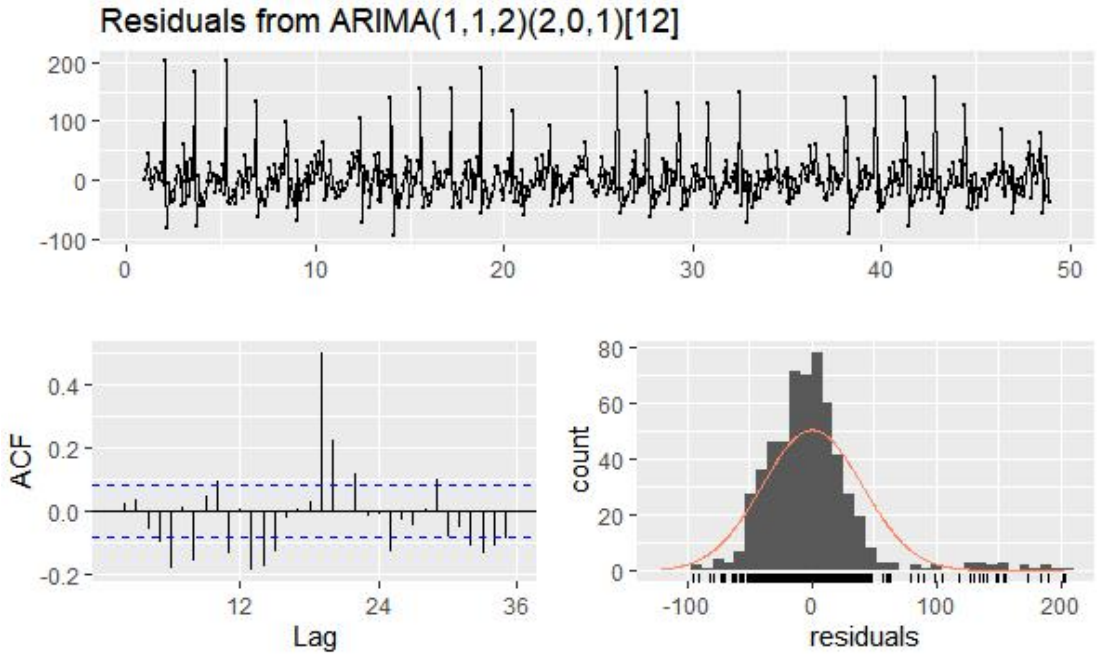
Besides, I also use Arima() function and take (1,1,2)(2,0,1), which is found out from pictures of ACF and PACF of the time series data after first difference, into it to deal with my time series data, the details of ARIMA(1,1,2)(2,0,1) model is listed below:

```
> # p=1, d=1, q=2
> arima_fit_daytime <- Arima(train_day_time,order = c(1,1,2),seasonal = c(2,0,1))
> arima_fit_daytime
Series: train_day_time
ARIMA(1,1,2)(2,0,1)[12]

Coefficients:
         ar1      ma1      ma2     sar1     sar2    sma1
      0.4956  -0.5755  -0.4245  -0.7495  -0.0493  0.5175
s.e.     NaN      NaN      NaN      NaN      NaN     NaN

sigma^2 estimated as 1636:  log likelihood=-2943.72
AIC=5901.45   AICc=5901.65   BIC=5931.93
```

Compared AIC of ARIMA(1,1,2)(2,0,1) with ARIMA(4,0,5)(2,0,1), we could find that AIC of ARIMA(1,1,2)(2,0,1) is smaller, which means ARIMA(1,1,2)(2,0,1) is better.

Finally, I checked the residuals of the model, the details is listed below:

```
> checkresiduals(arima_fit_daytime)

        Ljung-Box test

data:  Residuals from ARIMA(1,1,2)(2,0,1)[12]
Q* = 300.51, df = 18, p-value < 2.2e-16

Model df: 6.   Total lags used: 24
```



We could find that the p-value here is relatively small, which means the model is good enough.

6. Discuss the limitations of each model with respect to this dataset.

Judgmental forecasts are subjective, and therefore do not come free of bias or limitations.

For seasonal ARIMA, there are some limitations of it is its hypothesis. The fist one is this time series data is stationary, or stationary after differencing, which means there is no trend and seasonality in this data and variance should be a constant. The second one is this data must be autocorrelated with the previous one and have linear relationship, which means ARIMA only capture linear relationship of data and it don't fit those data which don't have linear relationship.

Moreover, the SARIMA model can get a very beautiful prediction based on the previous data. But if there is some accident, like travel ban during period of COVID-19, which have a huge impact on data. This means that the prediction will fail in this situation.

For average method which is corresponding to my first model. All the data only comes from historical record. All trends can not replace the future, if using some test data for testing the model, the test result cannot represent the future results. As I

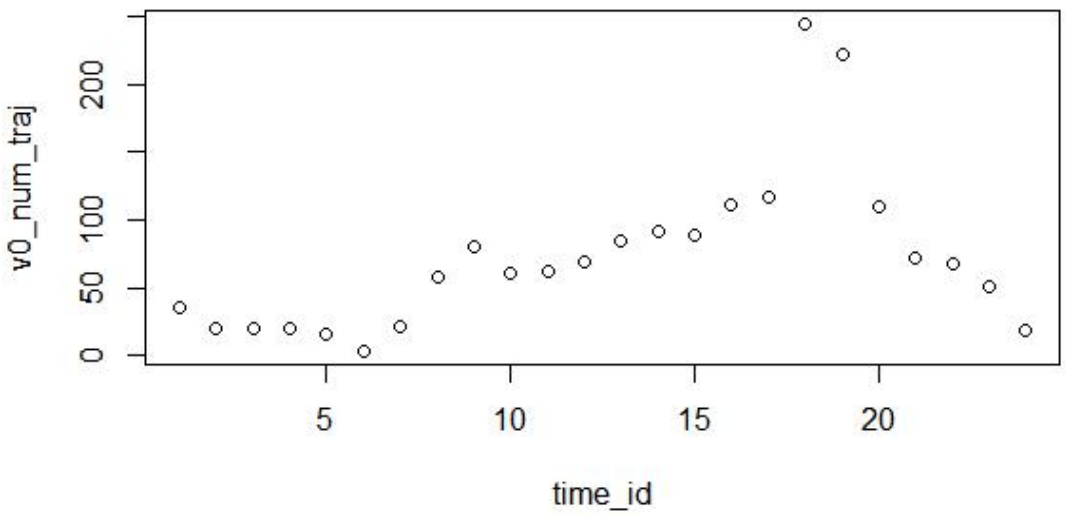
mentioned in question 4, some prediction doesn't match reality, like using this method to predict time series data (delete weekend).

7. Give and plot model predictions for each model over the observed data range (include the observed values somehow for comparison). Also give 95% predictive intervals for the stochastic model over this range.

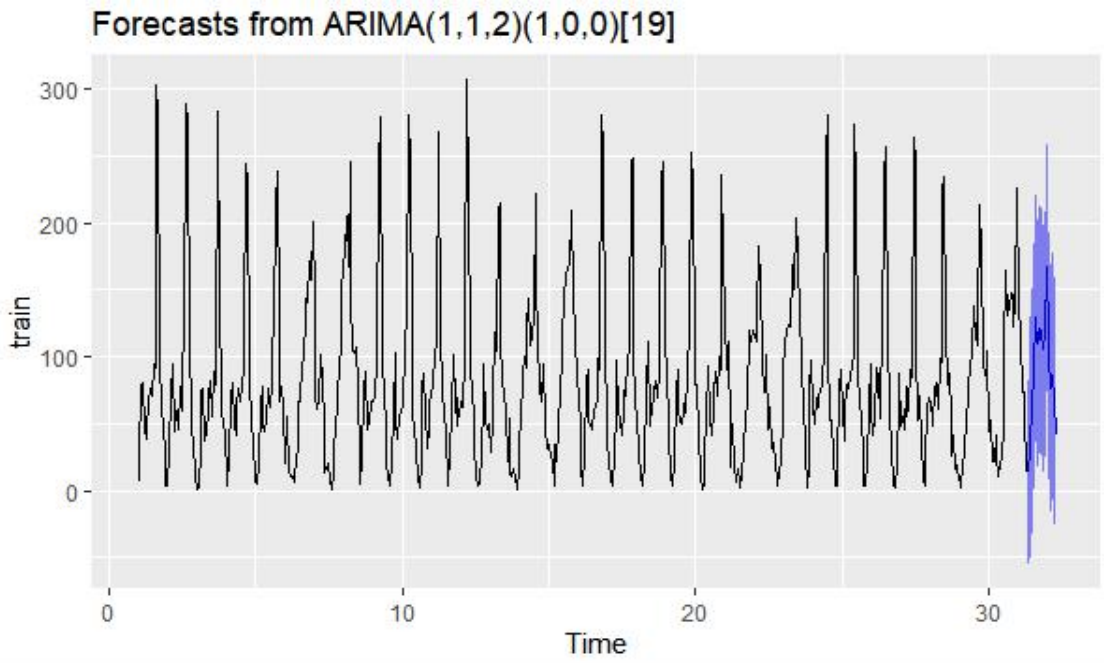
A. **time series data (delete time_id from 1 to 5)**

For Non-stochastic model:

```
> #the first type model
> pred_mean <- aggregate(v0_num_traj~time_id, data=flow, mean)
> pred_mean
   time_id v0_num_traj
1        1   35.083333
2        2   19.875000
3        3   20.125000
4        4   19.750000
5        5   15.250000
6        6    3.678571
7        7   22.035714
8        8   57.857143
9        9   79.678571
10      10   60.464286
11      11   61.964286
12      12   69.464286
13      13   83.928571
14      14   91.892857
15      15   89.392857
16      16  110.857143
17      17  116.964286
18      18  244.178571
19      19  221.321429
20      20  109.107143
21      21   71.607143
22      22   67.535714
23      23   50.535714
24      24   18.464286
```



For stochastic model:



From the forecast plot listed above, the blue line is the forecast result for next 19 hours. We could find that the 95% predictive interval is wider.

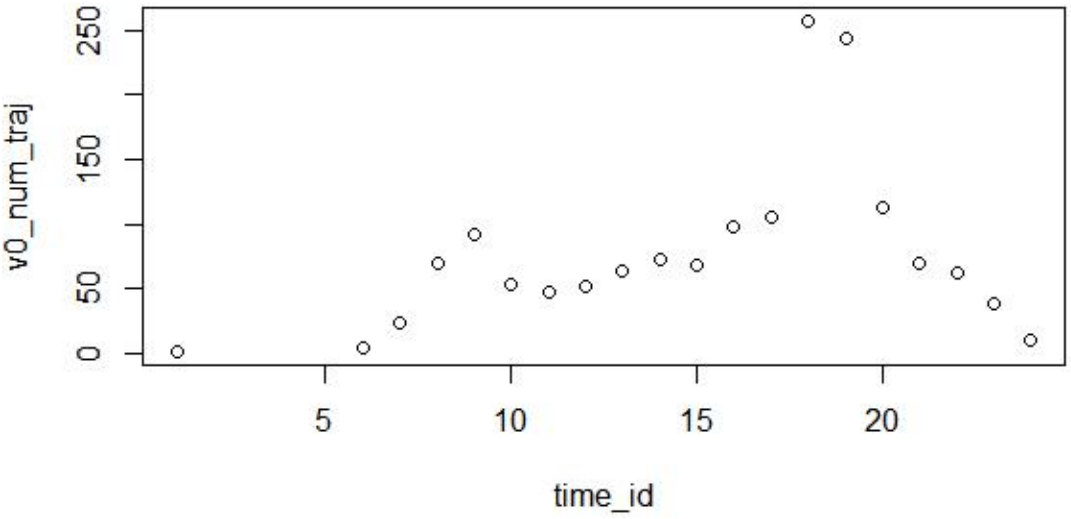
I also listed the form of 95% predictive interval of next 19 hours forecast.

```
> #predict next 19 hours
> next_day_fitted <- forecast(arima_fit,h=19,level=c(95))
> next_day_fitted
         Point Forecast       Lo 95     Hi 95
31.31579       13.07684 -55.1394517  81.29313
31.36842       31.37181 -54.8047998 117.54842
31.42105       39.63804 -50.9016989 130.17778
31.47368       58.47867 -33.2473175 150.20466
31.52632       92.32696   0.2659281 184.38799
31.57895      129.48276  37.3238929 221.64163
31.63158      110.38095  18.1921139 202.56978
31.68421      114.79528  22.5966103 206.99396
31.73684      120.86821  28.6659865 213.07043
31.78947      119.75179  27.5481590 211.95543
31.84211      105.53916  13.3349093 197.74342
31.89474      116.24619  24.0416427 208.45074
31.94737      167.31872  75.1140293 259.52341
32.00000      115.66811  23.4633430 207.87287
32.05263      101.42100   9.2162024 193.62580
32.10526       76.48511 -15.7197139 168.68993
32.15789       86.57959  -5.6252429 178.78442
32.21053       68.17383 -24.0310019 160.37867
32.26316       41.45542 -50.7494132 133.66026
> next_day_fitted_value <- round(next_day_fitted$mean,0)
> next_day_fitted_value
Time Series:
Start = c(31, 7)
End = c(32, 6)
Frequency = 19
 [1]  13  31  40  58  92 129 110 115 121 120 106 116 167 116 101  76  87  68  41
```

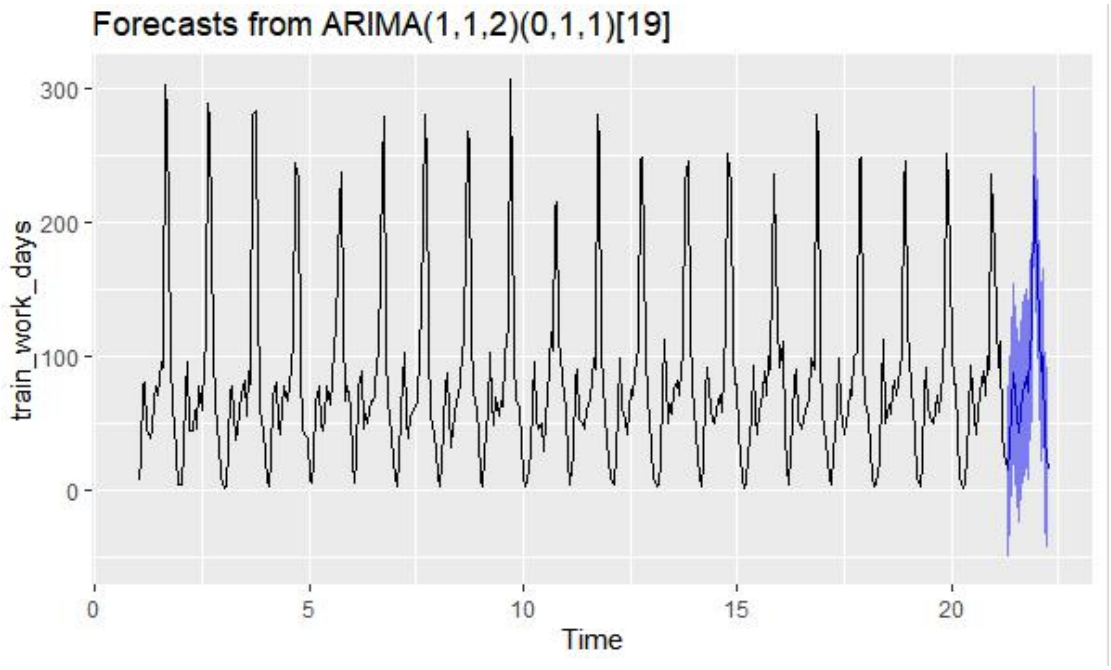
B. time series data (delete weekend)

For Non-stochastic model:

```
> #the first type model of work days data
> pred_mean_work <- aggregate(v0_num_traj~time_id, data=flow_work, mean)
> pred_mean_work
   time_id v0_num_traj
1        1        2.00
2        6        3.95
3        7       24.10
4        8       69.65
5        9       92.70
6       10       52.85
7       11       47.40
8       12       52.60
9       13       63.30
10      14       72.25
11      15       68.45
12      16       97.75
13      17      105.05
14      18      256.65
15      19      242.70
16      20      112.50
17      21       69.30
18      22       62.25
19      23       37.95
20      24       11.05
```



For stochastic model:



From the forecast plot listed above, the blue line is the forecast result for next 19 hours. We could find that the 95% predictive interval is wider.

I also listed the form of 95% predictive interval of next 19 hours forecast.

```
> #predict next 19 hours
> next_workday_fitted <- forecast(arima_fit_work,h=19,level=c(95))
> next_workday_fitted
         Point Forecast      Lo 95     Hi 95
21.31579       13.49518 -50.781113  77.77148
21.36842       31.68442 -35.874826  99.24367
21.42105       61.70889  -5.943141 129.36092
21.47368       87.16696  19.509057 154.82487
21.52632       54.60658 -13.051022 122.26418
21.57895       42.76915 -24.888532 110.42683
21.63158       58.64233  -9.015338 126.30000
21.68421       72.10348   4.445805 139.76115
21.73684       84.20858  16.550905 151.86624
21.78947       73.89164   6.233975 141.54931
21.84211      104.07665  36.418982 171.73432
21.89474      117.18157  49.523897 184.83924
21.94737      234.60902 166.951350 302.26669
22.00000      165.99761  98.339937 233.65528
22.05263      119.47356  51.815894 187.13123
22.10526       88.27141  20.613736 155.92908
22.15789       99.21429  31.556624 166.87196
22.21053       35.67351 -31.984158 103.33118
22.26316       14.35553 -53.302138  82.01320
> next_workday_fitted_value <- round(next_workday_fitted$mean,0)
> next_workday_fitted_value
Time Series:
Start = c(21, 7)
End = c(22, 6)
Frequency = 19
 [1]  13  32  62  87  55  43  59  72  84  74 104 117 235 166 119  88  99  36  14
```

However, compared the 95% predictive intervals by looking at pictures of it, we could find the predictive interval of seasonal ARIMA with time series data (delete time_id from 1 to 5) is wider than time series data (delete weekend). This means that seasonal ARIMA with time series data (delete weekend) is better.

8.Evaluate accuracy 1 and 2 hours ahead via final day.

From what we have discussed from question 7, we have known that seasonal ARIMA with time series data (delete weekend) is better. Therefore, I decided to use it.

Using the Non-stochastic Model can not predict related values because all the data only based on historical data.

So I use the ARIMA Model to make prediction:

(1)One hour ahead predict:
Firstly, I use the for round method to build the model in order to avoid step by step building models. The result of prediction is listed below:

```
> one_hour_predict
 [1] 109.550368  73.078420  57.381829  43.797462   7.032204   5.193238  19.862272
 [8]  70.475333  78.769490  54.164244  66.014001  44.292139  61.955559  70.870366
[15]  61.314812  96.468045 113.232780 246.856962 175.737990
```

And then I put the predicted values and actual values of passenger flow together to make comparison more convenient and direct. I put snapshot of result below:

```
> Compare_data_one_hour
         date time Predict passenger flow Actual passenger flow
1  2013-03-15    6              109.550368                   107
2  2013-03-15    7               73.078420                    79
3  2013-03-15    8               57.381829                    79
4  2013-03-15    9               43.797462                    43
5  2013-03-15   10                7.032204                    10
6  2013-03-15   11                5.193238                     1
7  2013-03-15   12               19.862272                     5
8  2013-03-15   13               70.475333                    24
9  2013-03-15   14               78.769490                    55
10 2013-03-15   15               54.164244                    93
11 2013-03-15   16               66.014001                    55
12 2013-03-15   17               44.292139                    41
13 2013-03-15   18               61.955559                    58
14 2013-03-15   19               70.870366                    72
15 2013-03-15   20               61.314812                    89
16 2013-03-15   21               96.468045                    71
17 2013-03-15   22              113.232780                   100
18 2013-03-15   23              246.856962                    90
19 2013-03-15   24              175.737990                   236
```

And then I use mse() function to calculate mse for 1 hour ahead predict to measure accuracy of prediction.

```
> mse(one_hour_predict,train_work_days[(nrow(flow_sub)-18):(nrow(flow_sub))])
[1] 1840.256
```

The mean squared error tells you how close a regression line is to a set of points. It does this by taking the distances from the points to the regression line (these distances are the “errors”) and squaring them. The squaring is necessary to remove any negative signs. It also gives more weight to larger differences. It’s called the

mean squared error as you're finding the average of a set of errors. The smaller the means squared error, the closer you are to finding the line of best fit.

From the snapshot of result of mse listed above, we could find that the value of mse is very high.

(2)two hour ahead predict:
Similarly, building the two hours ahead prediction model only needs to changes some parameters:

```
> two_hours_predict
 [1] 109.512957  73.912197  55.446375  36.703143   7.292261   4.227311  21.228296
 [8]  75.318920  94.013259  62.025117  53.287349  47.896490  63.025283  72.159752
[15]  60.944582  87.413918 121.521946 251.131344 227.978532
```

```
> Compare_data_two_hour
         date time Predict passenger flow Actual passenger flow
1  2013-03-15    6              109.512957                   107
2  2013-03-15    7               73.912197                    79
3  2013-03-15    8               55.446375                    79
4  2013-03-15    9               36.703143                    43
5  2013-03-15   10                7.292261                    10
6  2013-03-15   11                4.227311                     1
7  2013-03-15   12               21.228296                     5
8  2013-03-15   13               75.318920                    24
9  2013-03-15   14               94.013259                    55
10 2013-03-15   15               62.025117                    93
11 2013-03-15   16               53.287349                    55
12 2013-03-15   17               47.896490                    41
13 2013-03-15   18               63.025283                    58
14 2013-03-15   19               72.159752                    72
15 2013-03-15   20               60.944582                    89
16 2013-03-15   21               87.413918                    71
17 2013-03-15   22              121.521946                   100
18 2013-03-15   23              251.131344                    90
19 2013-03-15   24              227.978532                   236
```

```
> #Calculate the MSE for 2 hours ahead predict
> mse(two_hours_predict,train_work_days[(nrow(flow_sub)-18):(nrow(flow_sub))])
[1] 1770.841
```

Compared with the predict model of 1 hour ahead, the MSE of predict model of 2 hours ahead is lower, which means the accuracy of this predict model is higher than the one hour predict model.

From what has been discussed above, we could find that the predict model of 1 hour ahead is better.

9. Make predictions for test day, which will be evaluated by RMSE vs true counts for that day.

To use the new data, I combined the new data with the original data and converted the data type to integer to facilitate building the model.

```
#Add the known rows to the original data
row1 <- c("2013-03-25",6,2)
row2 <- c("2013-03-25",7,18)
row3 <- c("2013-03-25",8,34)
row4 <- c("2013-03-25",9,68)
new_data <- rbind(flow_sub,row1,row2,row3,row4)

#Transform the type of the data for building model
new_data$time_id = as.integer(new_data$time_id)
new_data$v0_num_traj = as.integer(new_data$v0_num_traj)
```

And then I use new dataset to build a new ARIMA Model:

```
#Build new time series model
new_train <- ts(new_data$v0_num_traj, frequency = 19,start=c(1,1))
new_arima_model = arima(new_train, order = c(1,1,2),seasonal = list(order = c(0,1,1), period=19))
```

Finally I reform the output data and combine the time_id with the predict result. The snapshot of predict result is listed below:

```
> predicted_next_15_hours
   Time ID Predict passenger flow
1         6                49.15296
2         7                48.73464
3         8                51.71732
4         9                64.67783
5        10                76.14609
6        11                67.30244
7        12                95.57358
8        13               102.86397
9        14               247.65277
10       15               226.38915
11       16               114.28842
12       17                75.88172
13       18                73.03952
14       19                38.02727
15       20                11.61409
```

And then, I take out data of time_id from 6 to 9:

```
> predicted_next_4_hours
  Time ID Predict passenger flow
1        6                49.15296
2        7                48.73464
3        8                51.71732
4        9                64.67783
```

And then I use RMSE function to evaluated predict values vs true counts for 25/03/2013. The value of RMSE is 29.55063, which means that the predicted value fits well with the actual value. It also means that the seasonal ARIMA(1,1,2)(0,1,1) is better.

```
> real <- c(2,18,34,68)
> predicted <-c(49.15296,48.73464,51.71732,64.67783)
> RMSE(real,predicted)
[1] 29.55063
```

10. Paragraph to Translink staff

Hi dear translink staff,
The seasonal ARIMA(1,1,2)(0,1,1) is built based on statistical methods and using the data that translink company has recorded to predict the passenger flow every day from Brisbane City to South Brisbane. Through the prediction value of passenger flow, you could find out that passenger flow are high from morning 9:00 to 20:00, especially the time range from 16:00 to 20:00 is the highest. Therefore, I suggest that you could arrange more bus/train/ferry services to run in this time range. I hope this model could help you to arrange how many bus/train/ferry services to run at appropriate times to meet demand. That's all, thank you.