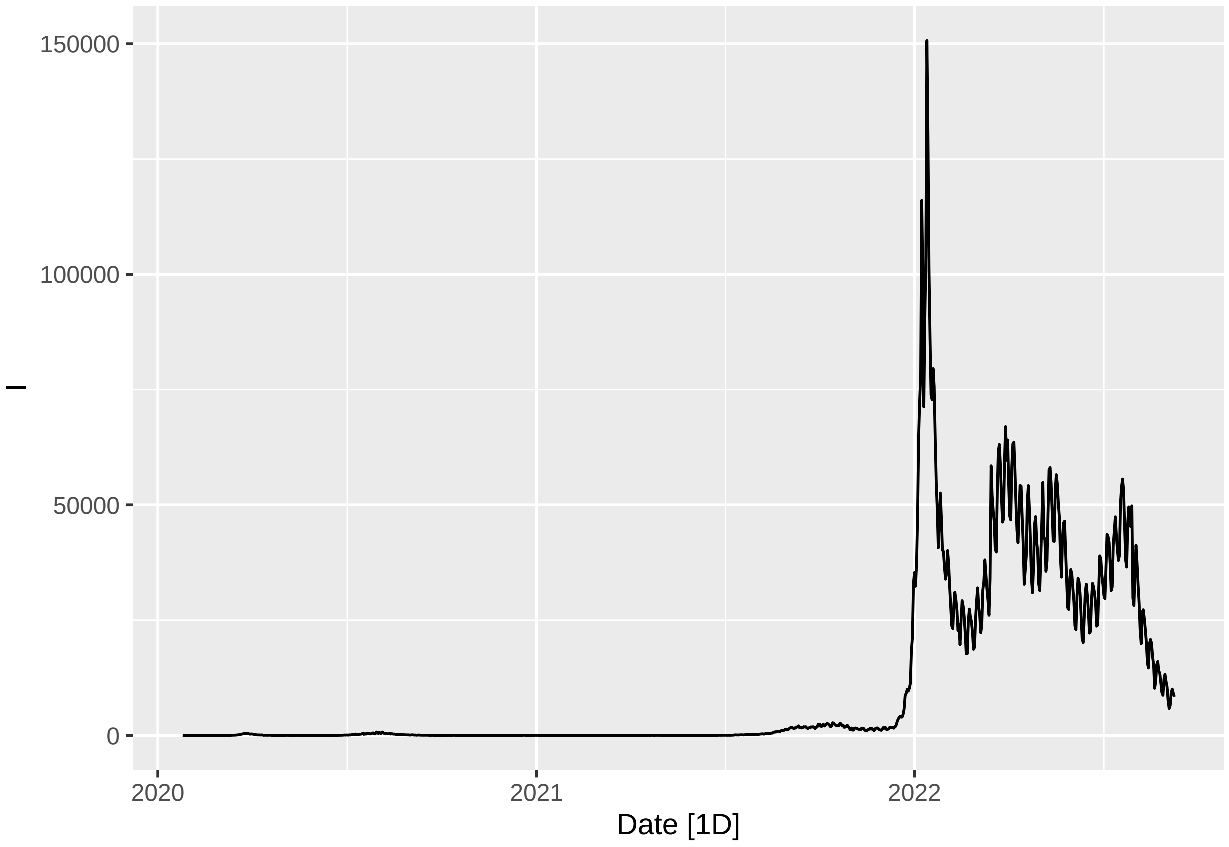
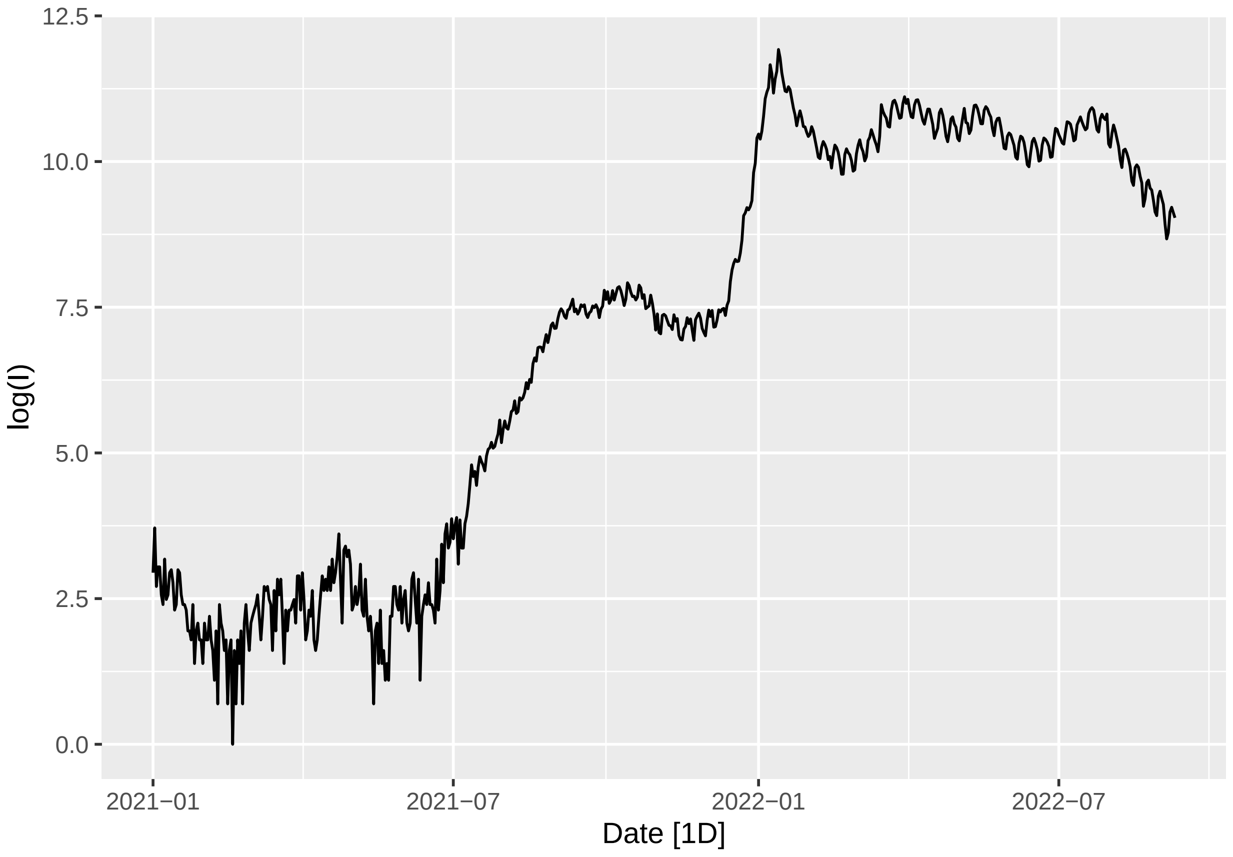
Back transformation before forecasting

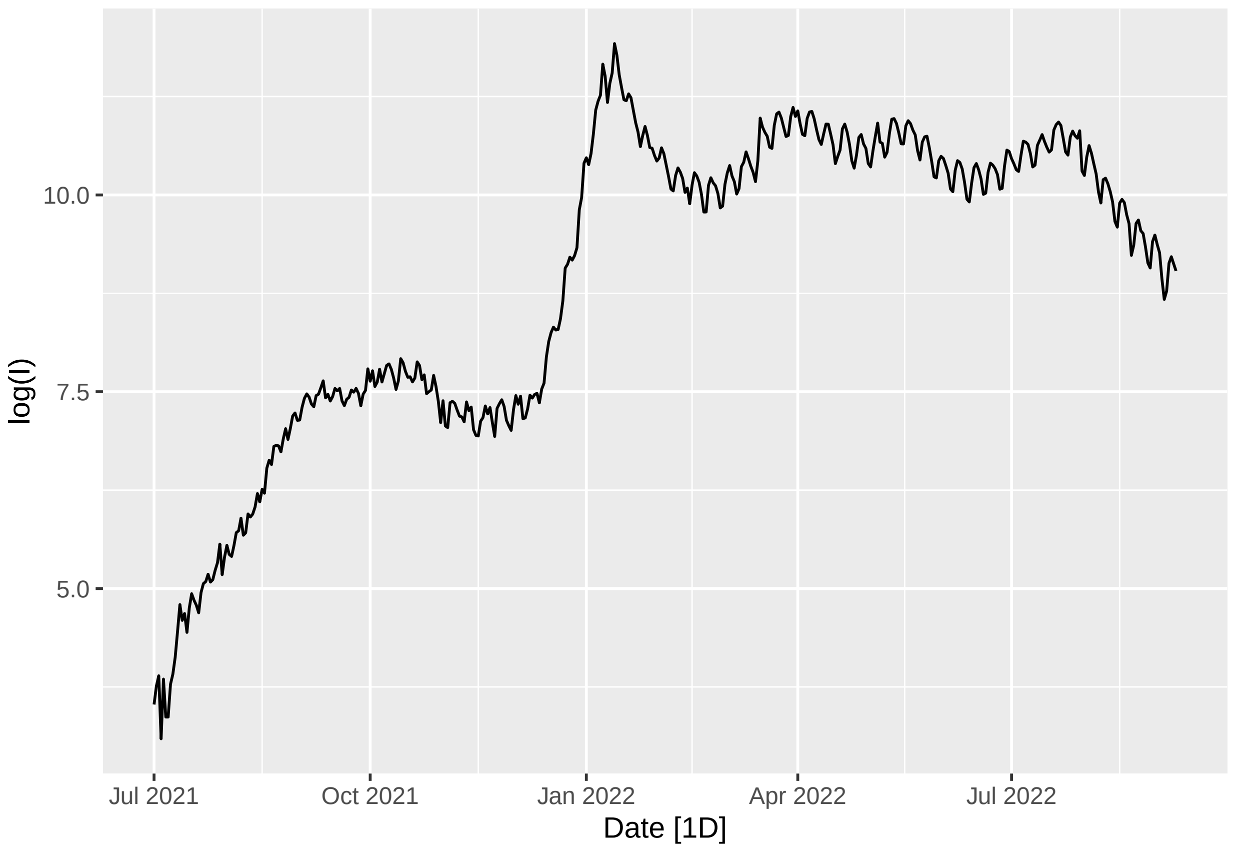
Orginal data

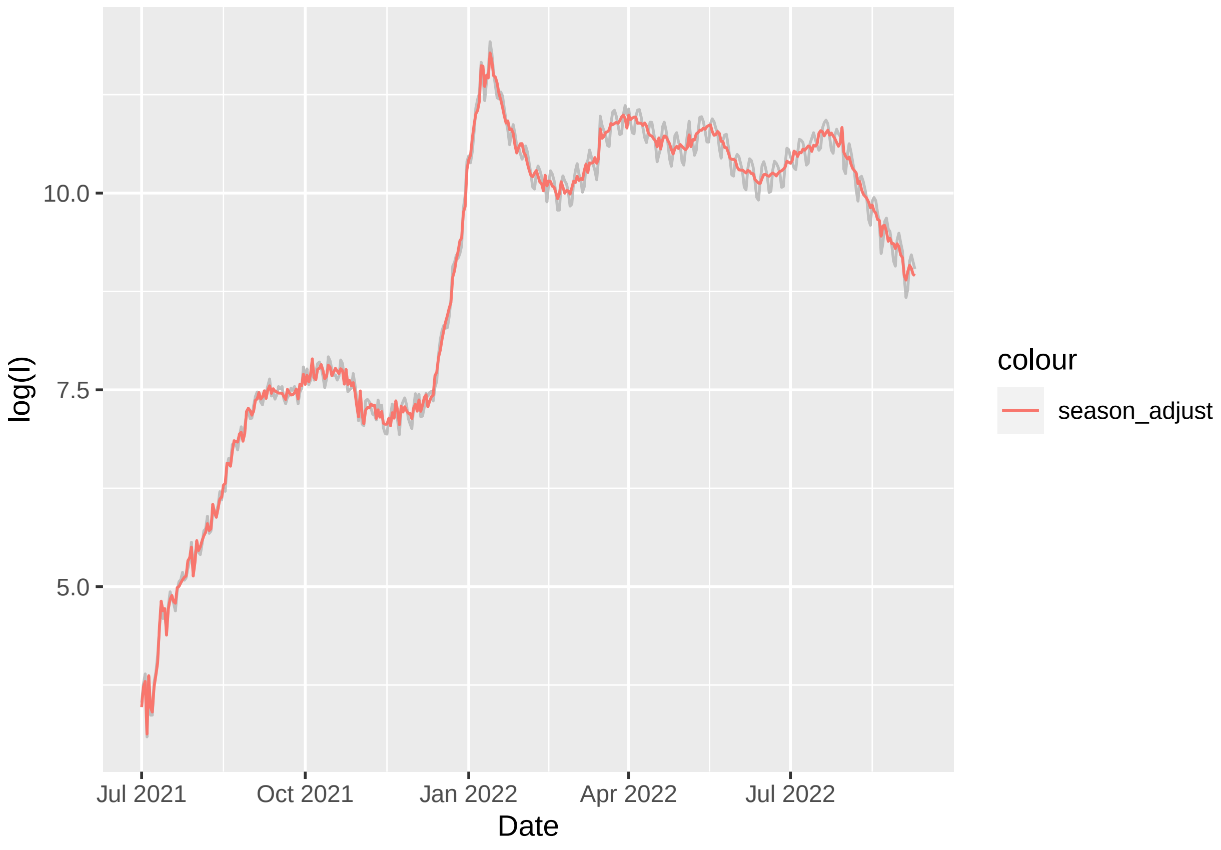
```
data_ts %>%
  autoplot()
```



log transformation




log(I)
10.0
7.5
5.0
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Date [1D]

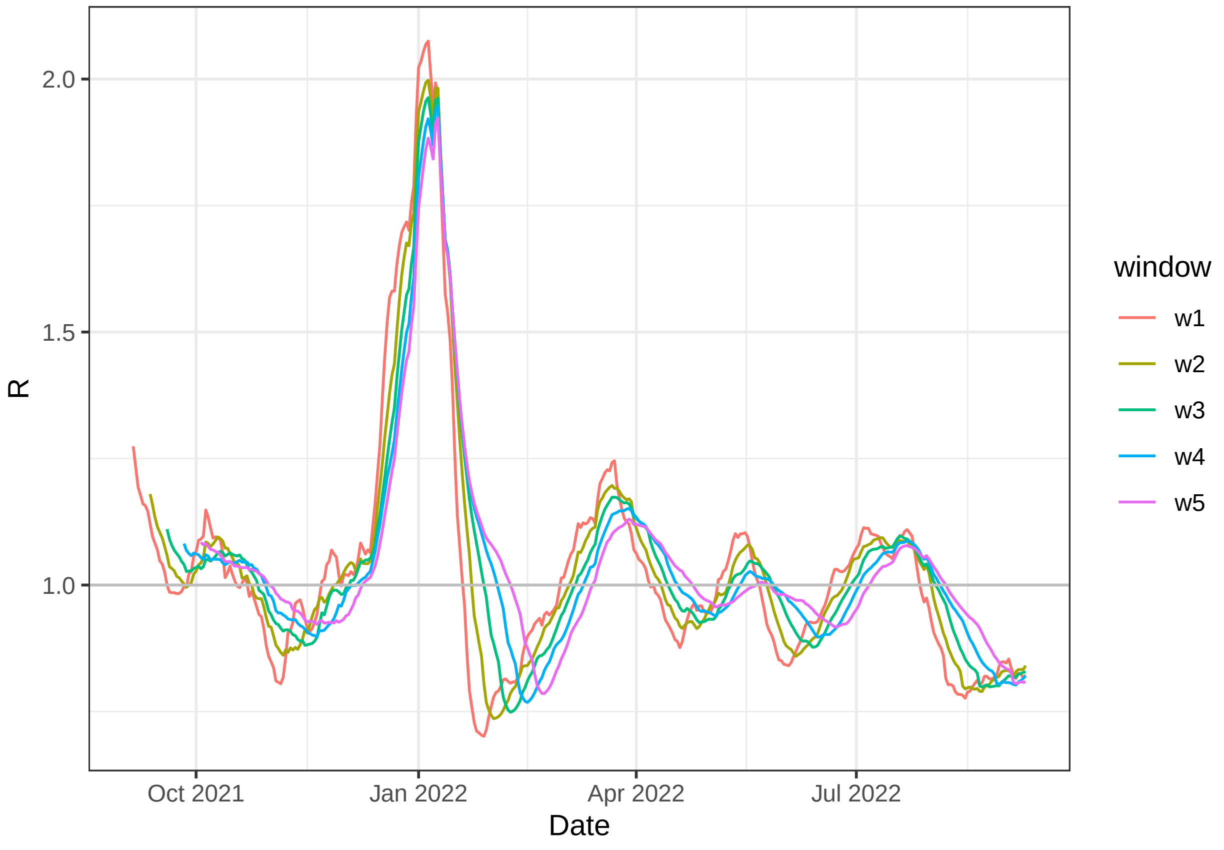

log(I)
10.0
7.5
5.0
Jul 2021
Oct 2021
Jan 2022
Apr 2022
Jul 2022
Date
colour
season_adjust

Function

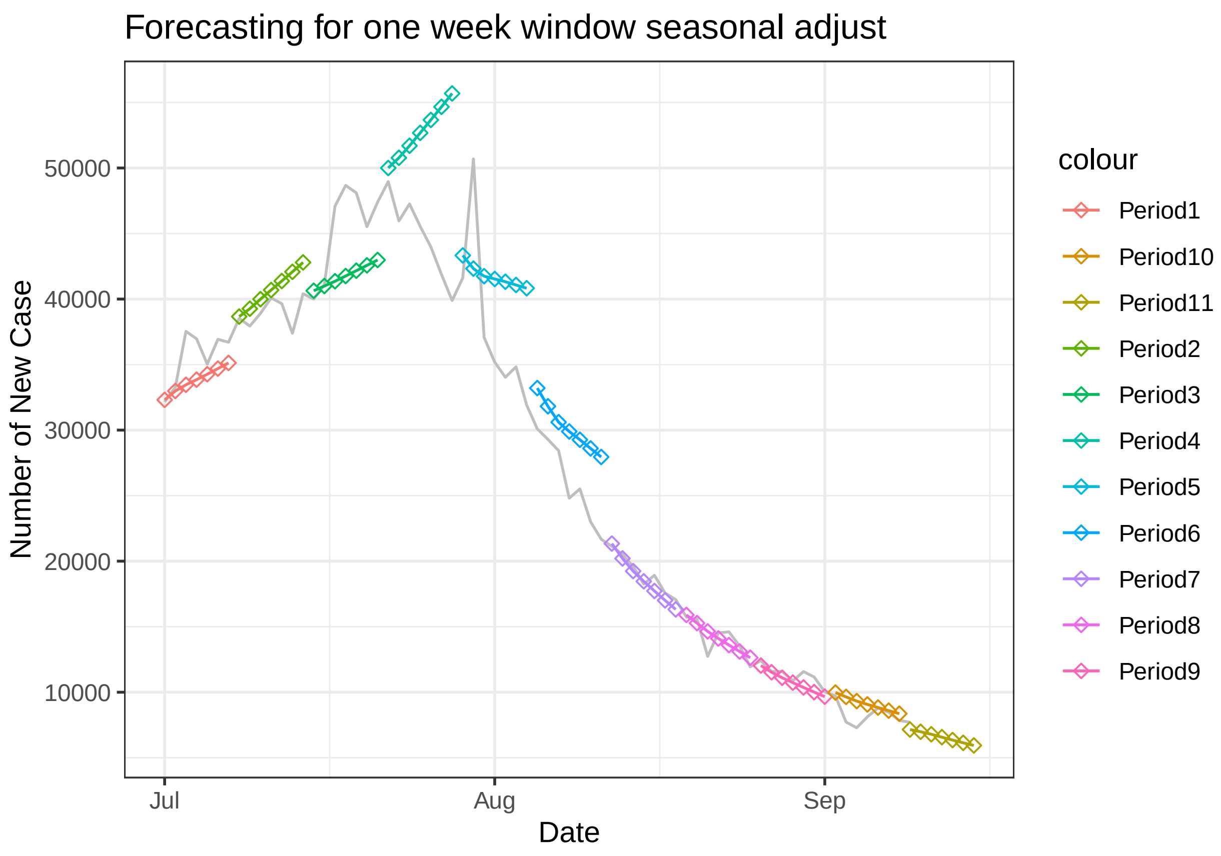
```r
r_estiamte <- function(df, start, end, mean, std){
  output <- estimate_R(df, method = "parametric_si",
                config = make_config(list(mean_si = mean, std_si = std,
                                          t_start = start,
                                          t_end = end)))
  output$dates <- data$Date

  return(output)
}

find_r <- function(date, r_df, full_df){
  r_matrix <- matrix(NA, nrow = length(date))
  for (i in 1:length(date)) {
    date_index <-which(full_df$Date == date[i])
    r_index <- which(r_df$R$t_end == date_index)
    r_matrix[i,] <- r_df$R$`Mean(R)`[r_index]
  }
  date_r = tibble("Date" = range, "R" = r_matrix[,1])
  return(date_r)
}

forecast_i <- function(r_date, full_df, r_df){

  output_df <- tibble("Date"= as.Date(NA),
                      "I" = as.numeric(NA),
                      "Week" = as.numeric(NA))

  for (i in 1:dim(r_date)[1]) {

    I_renew<-full_df$I[which(full_df$Date <= r_date$Date[i])]
    I_lambda <-I_renew[(length(I_renew) - 99):length(I_renew)]
    data.frame(r_df$si_distr)[,1][1:100] -> si
    predict_w1 <- matrix(NA, nrow = 7, ncol = 1)

    for (j in 1:7) {
      element <-  overall_infectivity(I_lambda, si)[100+j-1] * r_date$R[i]
      predict_w1[j,1] <- element
      I_lambda <- append(I_lambda, element)
      si <- append(si, 0)

      temp <- tibble("Date" = seq(ymd(r_date$Date[i]), ymd(r_date$Date[i])+6, "day"),
                     "I" = predict_w1[,1],
                     "Week" = i)
    }

    output_df <- bind_rows(output_df, temp)

  }

  output_df %>%
    drop_na() %>%
    mutate(Week = paste0("Period", Week)) -> output_df
  return(output_df)

}


add_season_pattern <- function(f_data, seasona_data){

  f_data$Date - 7 -> season_date

  seasona_data %>%
    filter(Date %in% season_date) -> seasona_df

  f_data$I <- f_data$I + seasona_df$season_week
  return(f_data)
}
```

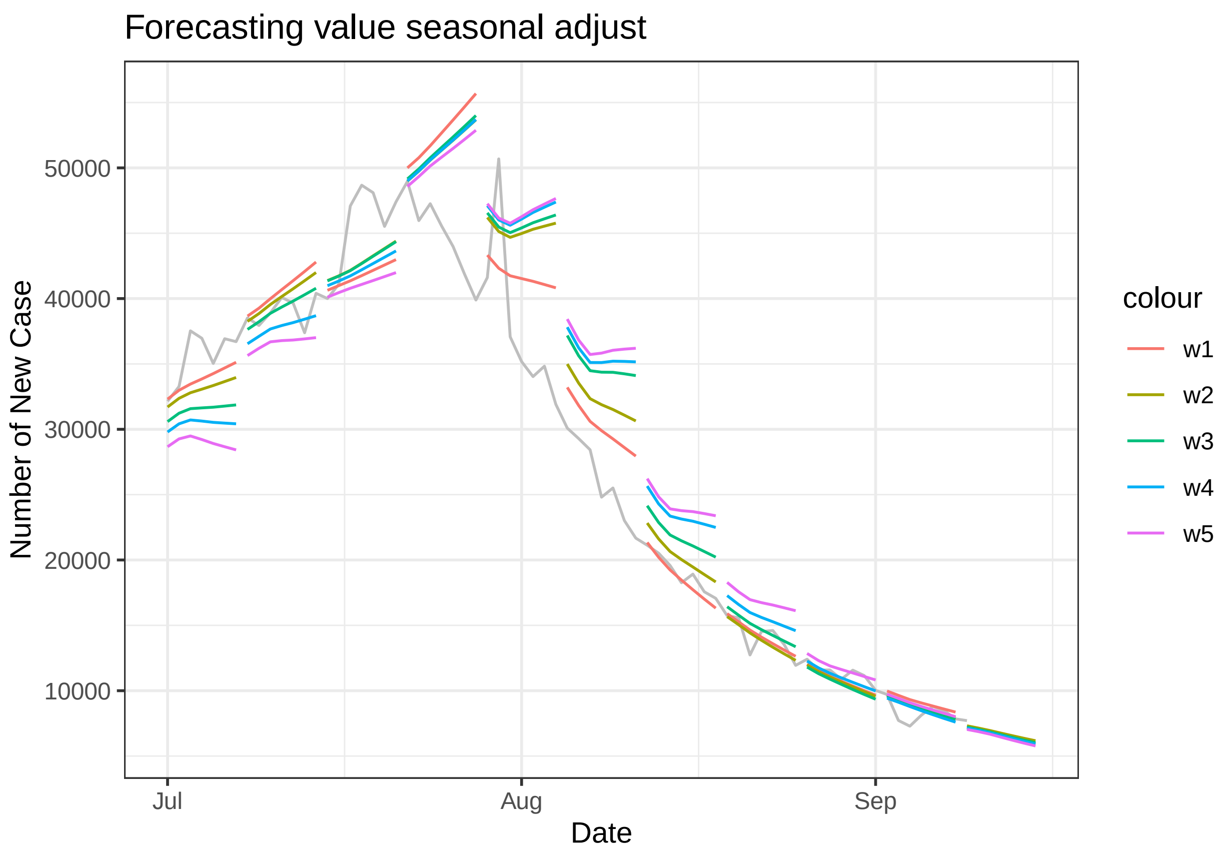
Estimate R

Using SI mean 4.7, std 2.9

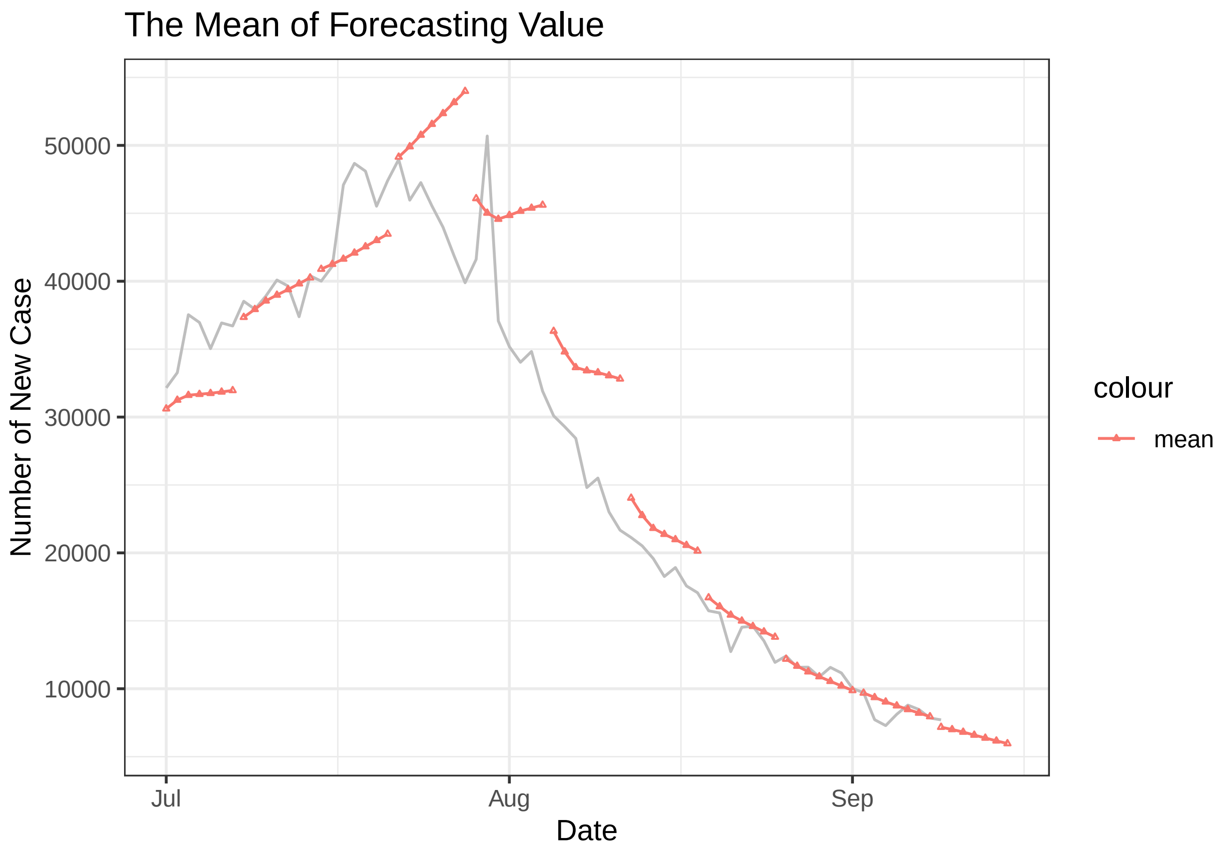


Graph for seasonal adjust





Forecasting value seasonal adjust
Number of New Case
50000
40000
30000
20000
10000
Jul
Aug
Sep
Date
colour
w1
w2
w3
w4
w5



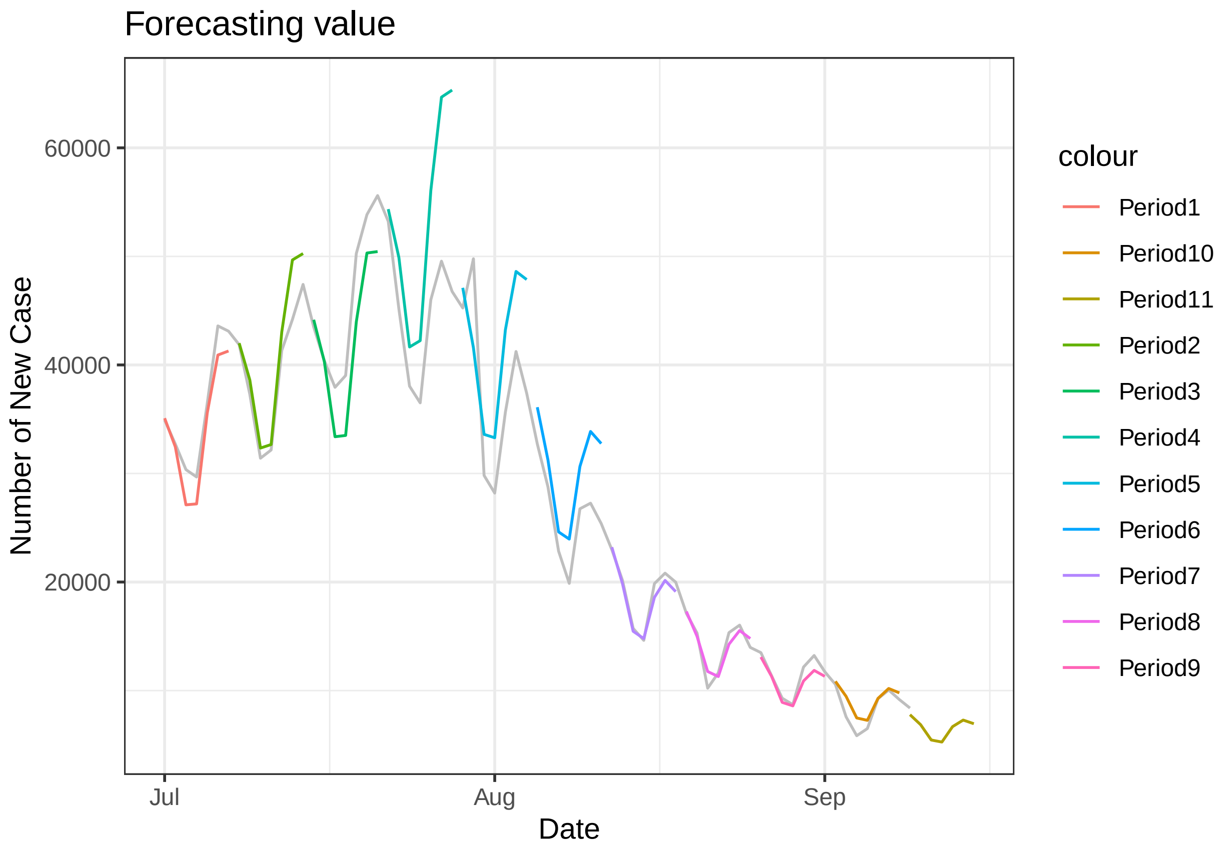
The Mean of Forecasting Value
Number of New Case
50000
40000
30000
20000
10000
Jul
Aug
Sep
Date
colour
mean

It looks good for seasonal adjust

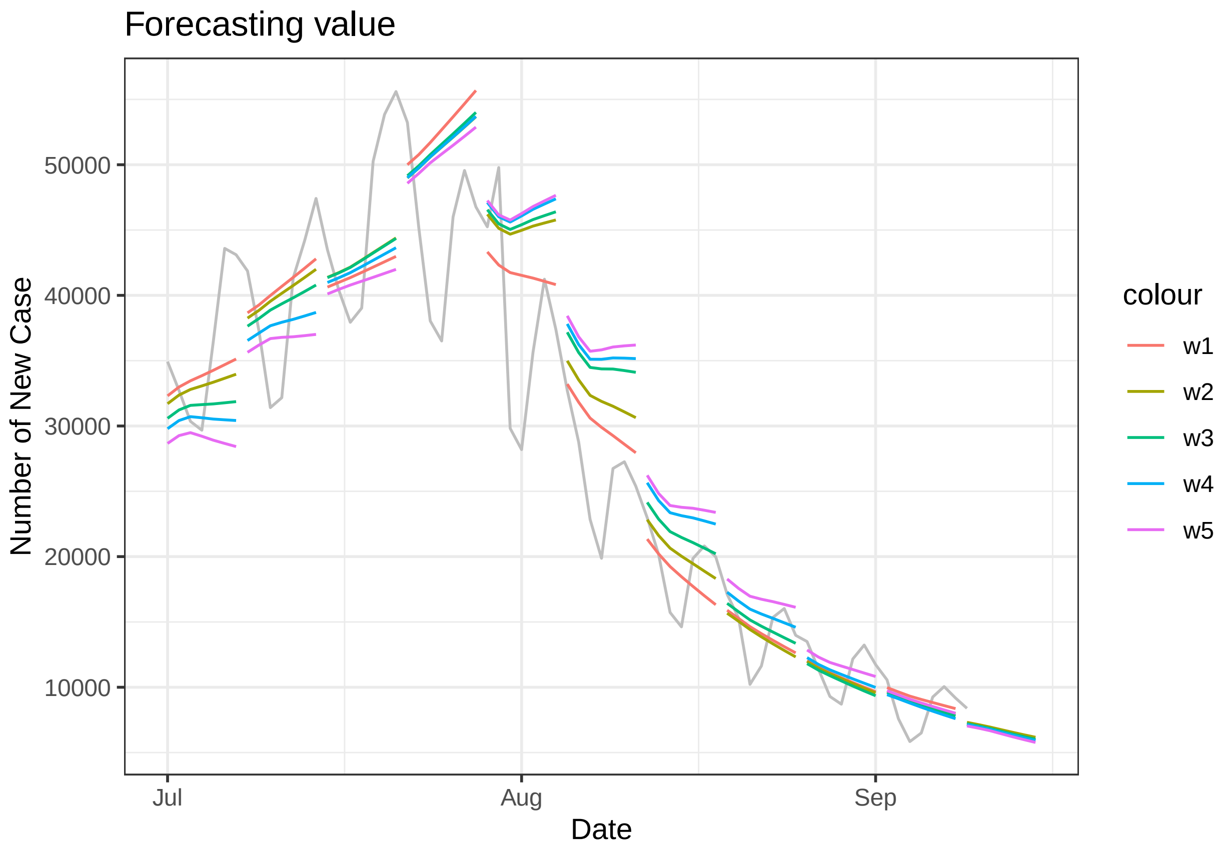
Add Back Seasonal pattern

```
add_season_pattern <- function(f_data, seasona_data){

  f_data$Date - 7 -> season_date

  seasona_data %>%
    filter(Date %in% season_date) -> seasona_df

  f_data$I <- f_data$I + seasona_df$season_week
  return(f_data)
}
```

```
full_data %>%
  filter(Date >= "2022-07-01") %>%
  ggplot() +
  geom_line(aes(x = Date,  y = I,color = "Actual"),color = "grey") +
  geom_line(aes(x = Date, y = I, color = Week) ,data = window1_predict)+
  ggtitle("Forecasting value") +
  ylab("Number of New Case") +
  theme_bw()
```



```
full_data %>%
  filter(Date >= "2022-07-01") %>%
  ggplot() +
  geom_line(aes(x = Date,  y = I,color = "Actual"),color = "grey") +
  geom_line(aes(x = Date, y = I,group = interaction(week, Windows),
              color = Windows) ,data = predict_long)+
  ggtitle("Forecasting value") +
  ylab("Number of New Case") +
  theme_bw()
```



I use back transformed seasonal adjust into my model. Then, I add seasonal pattern back. For the seasonal pattern, I do exp for seasonal_week from decomp_comp for. However, it is not very significant

```
decomp_comp
```

```
## # A dable: 436 x 7 [1D]
## # Key:     .model [1]
## # :        log(I) = trend + season_week + remainder
##    .model Date       `log(I)` trend season_week remainder season_adjust
##    <chr>  <date>        <dbl> <dbl>       <dbl>     <dbl>         <dbl>
##  1 stl    2021-07-01     3.53  3.51      0.0566   -0.0416          3.47
##  2 stl    2021-07-02     3.76  3.53      0.0373    0.189           3.72
##  3 stl    2021-07-03     3.89  3.56      0.0959    0.238           3.80
##  4 stl    2021-07-04     3.09  3.59     -0.0362   -0.465           3.13
##  5 stl    2021-07-05     3.85  3.63     -0.0186    0.243           3.87
##  6 stl    2021-07-06     3.37  3.67     -0.0970   -0.201           3.46
##  7 stl    2021-07-07     3.37  3.71     -0.0391   -0.299           3.41
##  8 stl    2021-07-08     3.78  3.85      0.0578   -0.120           3.73
##  9 stl    2021-07-09     3.91  3.99      0.0384   -0.114           3.87
## 10 stl    2021-07-10     4.13  4.14      0.0954   -0.111           4.03
## # ... with 426 more rows
```

Accuary

```
full_data %>%
  filter(Date %in% window1_predict$Date) -> fc_actual

window1_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I) %>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE   MAPE
##   <dbl> <dbl>  <dbl>
## 1 2841. 4517. 0.0838
```

```
window2_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I) %>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE   MAPE
##   <dbl> <dbl>  <dbl>
## 1 3239. 5013. 0.0945
```

```
window3_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I) %>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE  MAPE
##   <dbl> <dbl> <dbl>
## 1 3846. 5713. 0.114
```

```
window4_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I)%>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE  MAPE
##   <dbl> <dbl> <dbl>
## 1 4390. 6200. 0.131
```

```
window5_predict %>%
  filter(Date <= "2022-09-09") %>%
  mutate(resid = I -fc_actual$I,
         p = resid/I) %>%
  summarise(MAE = mean(abs(resid)),
            RMSE = sqrt(mean(resid^2)),
            MAPE = mean(abs(p)))
```

```
## # A tibble: 1 x 3
##     MAE  RMSE  MAPE
##   <dbl> <dbl> <dbl>
## 1 4957. 6655. 0.151
```