207 Final Project - Trading Analysis and Stock Price Prediction

```r
library(tidyverse)
library(magrittr)
library(patchwork)

library(lubridate)

library(tsibble)
library(feasts)
install.packages('forecast')
library(forecast)

library(sandwich)
library(lmtest)

library(nycflights13)
install.packages('blsR')
library(blsR)

install.packages('gridExtra')
library(gridExtra)
```

Load in the Tesla Stock Data

```r
tesla_df <- read_csv("tesla_processed_data.csv")
```

```
## Rows: 2223 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): news
## dbl  (8): close, open, lowest, highest, total_vol, mean_vol, std_vol, is_up
## date (1): Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(tesla_df)
```

```
## # A tibble: 6 x 10
##   Date       close  open lowest highest total_vol mean_vol std_vol news    is_up
##   <date>     <dbl> <dbl>  <dbl>   <dbl>     <dbl>    <dbl>   <dbl> <chr>   <dbl>
## 1 2015-01-02  14.6  14.9   14.2    14.9  59157390  151686. 155761.  <NA>       0
## 2 2015-01-05  14.0  14.4   13.8    14.4  68662800  176058. 168291.  <NA>       1
## 3 2015-01-06  14.1  14.0   13.6    14.3  80752635  207058. 152662.  <NA>       0
## 4 2015-01-07  14.1  14.2   14.0    14.3  38728110   99558. 100907. "['BMW~     0
## 5 2015-01-08  14.0  14.2   14.0    14.3  43839960  112699. 130931. "['How~     0
## 6 2015-01-09  13.8  13.9   13.7    14.0  59398215  152695. 153615. "['How~     0
```

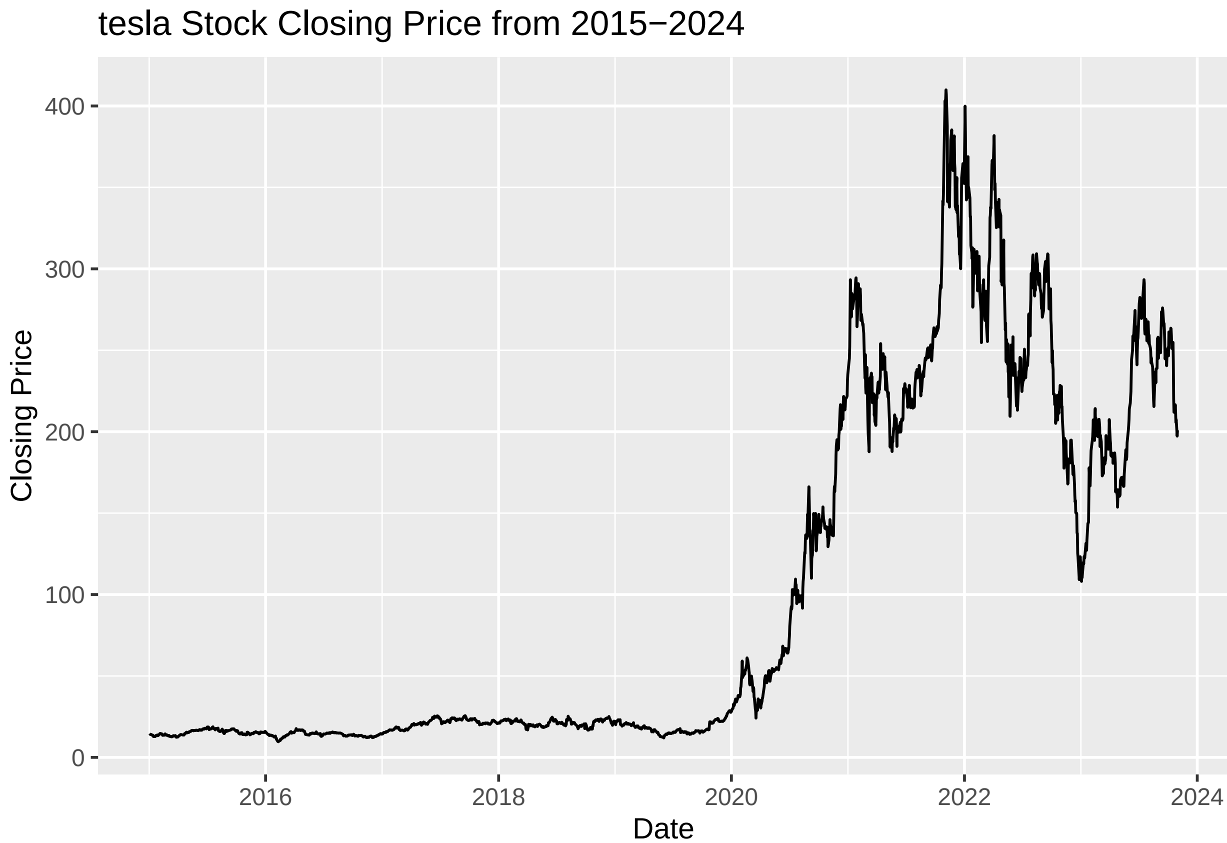
```r
summary(tesla_df)
```

```
##       Date                close              open             lowest
##  Min.   :2015-01-02   Min.   :  9.575   Min.   :  9.411   Min.   :  9.403
##  1st Qu.:2017-03-18   1st Qu.: 16.595   1st Qu.: 16.555   1st Qu.: 16.380
##  Median :2019-06-04   Median : 22.985   Median : 22.991   Median : 22.583
##  Mean   :2019-06-02   Mean   :100.383   Mean   :100.449   Mean   : 97.965
##  3rd Qu.:2021-08-16   3rd Qu.:207.390   3rd Qu.:207.060   3rd Qu.:200.762
##  Max.   :2023-10-31   Max.   :409.910   Max.   :410.150   Max.   :405.667
##     highest          total_vol            mean_vol          std_vol
##  Min.   : 10.33   Min.   :  8842065   Min.   :  39650   Min.   :  38794
##  1st Qu.: 16.83   1st Qu.: 46340730   1st Qu.: 119297   1st Qu.: 110841
##  Median : 23.32   Median : 66853260   Median : 172239   Median : 160293
##  Mean   :102.67   Mean   : 82685855   Mean   : 212614   Mean   : 192670
##  3rd Qu.:211.06   3rd Qu.:103454842   3rd Qu.: 265433   3rd Qu.: 229195
##  Max.   :414.50   Max.   :501420585   Max.   :1285694   Max.   :1946474
##      news               is_up
##  Length:2223        Min.   :0.0000
##  Class :character   1st Qu.:0.0000
##  Mode  :character   Median :1.0000
##                     Mean   :0.5196
##                     3rd Qu.:1.0000
##                     Max.   :1.0000
```

```r
tesla_tsib <- tesla_df %>% as_tsibble(index=Date)
tesla_tsib
```

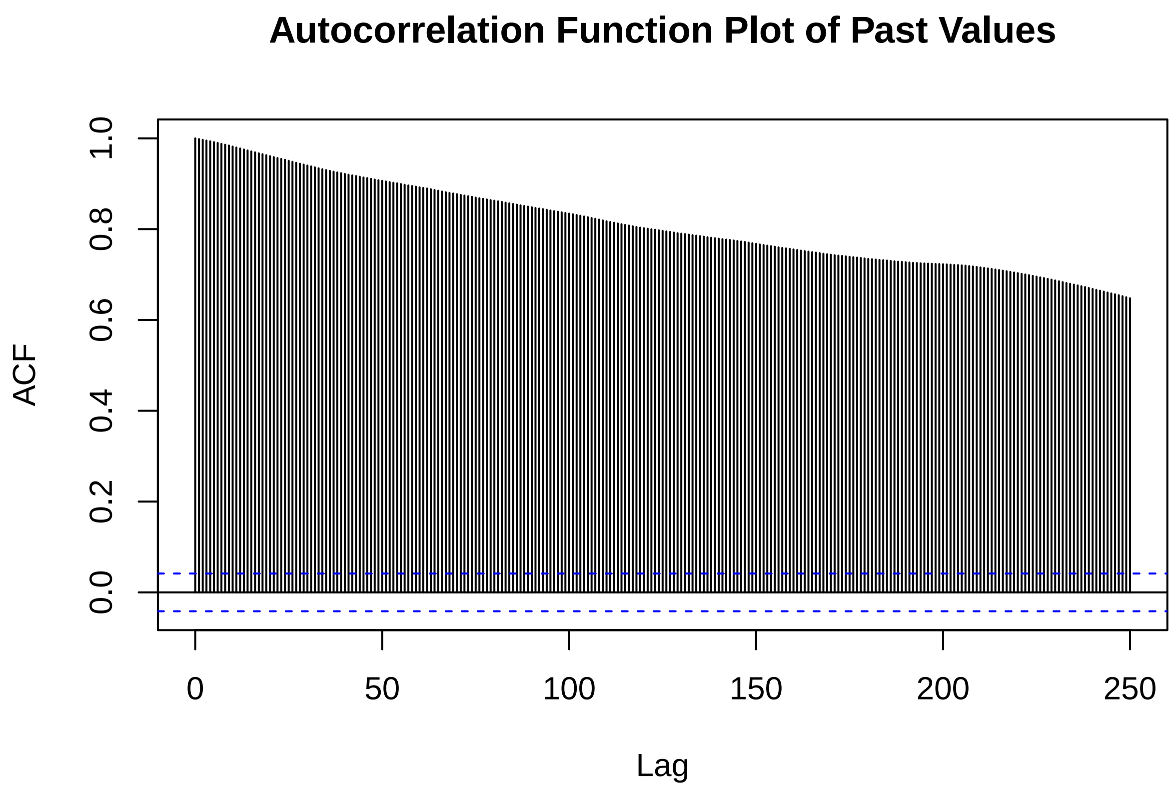
```
## # A tsibble: 2,223 x 10 [1D]
##    Date       close  open lowest highest total_vol mean_vol std_vol news   is_up
##    <date>     <dbl> <dbl>  <dbl>   <dbl>     <dbl>    <dbl>   <dbl> <chr>  <dbl>
##  1 2015-01-02  14.6  14.9   14.2    14.9  59157390  151686. 155761. <NA>       0
##  2 2015-01-05  14.0  14.4   13.8    14.4  68662800  176058. 168291. <NA>       1
##  3 2015-01-06  14.1  14.0   13.6    14.3  80752635  207058. 152662. <NA>       0
##  4 2015-01-07  14.1  14.2   14.0    14.3  38728110   99558. 100907. "['BM~      0
##  5 2015-01-08  14.0  14.2   14.0    14.3  43839960  112699. 130931. "['Ho~      0
##  6 2015-01-09  13.8  13.9   13.7    14.0  59398215  152695. 153615. "['Ho~      0
##  7 2015-01-12  13.5  13.6   13.3    13.6  72115905  184913. 210097. "['Ho~      1
##  8 2015-01-13  13.6  13.6   13.4    13.8  49602285  127512. 127221. "['Ho~      0
##  9 2015-01-14  12.8  12.7   12.3    13.0 137698725  353074. 550815. "['Ho~      0
## 10 2015-01-15  12.8  13.0   12.7    13.0  67520070  173128. 164068. "['Ho~      1
## # i 2,213 more rows
```

```r
tesla_close_price_stock_plot <- tesla_tsib |>
  ggplot(aes(x=Date, y = close)) +
  geom_line() +
  labs(x = "Date", y = "Closing Price",
       title = "tesla Stock Closing Price from 2015-2024")
tesla_close_price_stock_plot
```

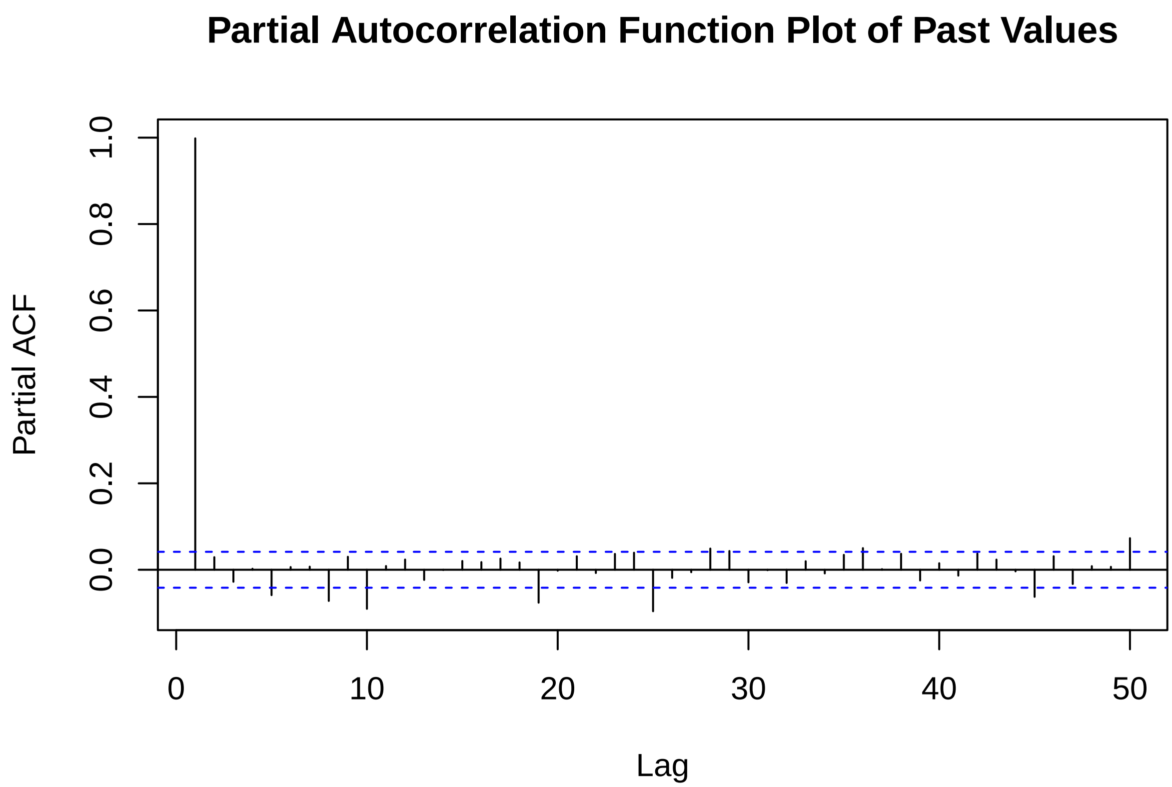


Clear non-stationarity, and increased variance throughout the right side of the plot.

```
acf(tesla_tsib$close, lag.max = 250, main = "Autocorrelation Function Plot of Past Values")
```




```
pacf(tesla_tsib$close, lag.max = 50, main = "Partial Autocorrelation Function Plot of Past Values")
```



The partial autocorrelations drop drastically after the first partial autocorrelation and remain insignificant throughout.

```
head(tesla_tsib)
```

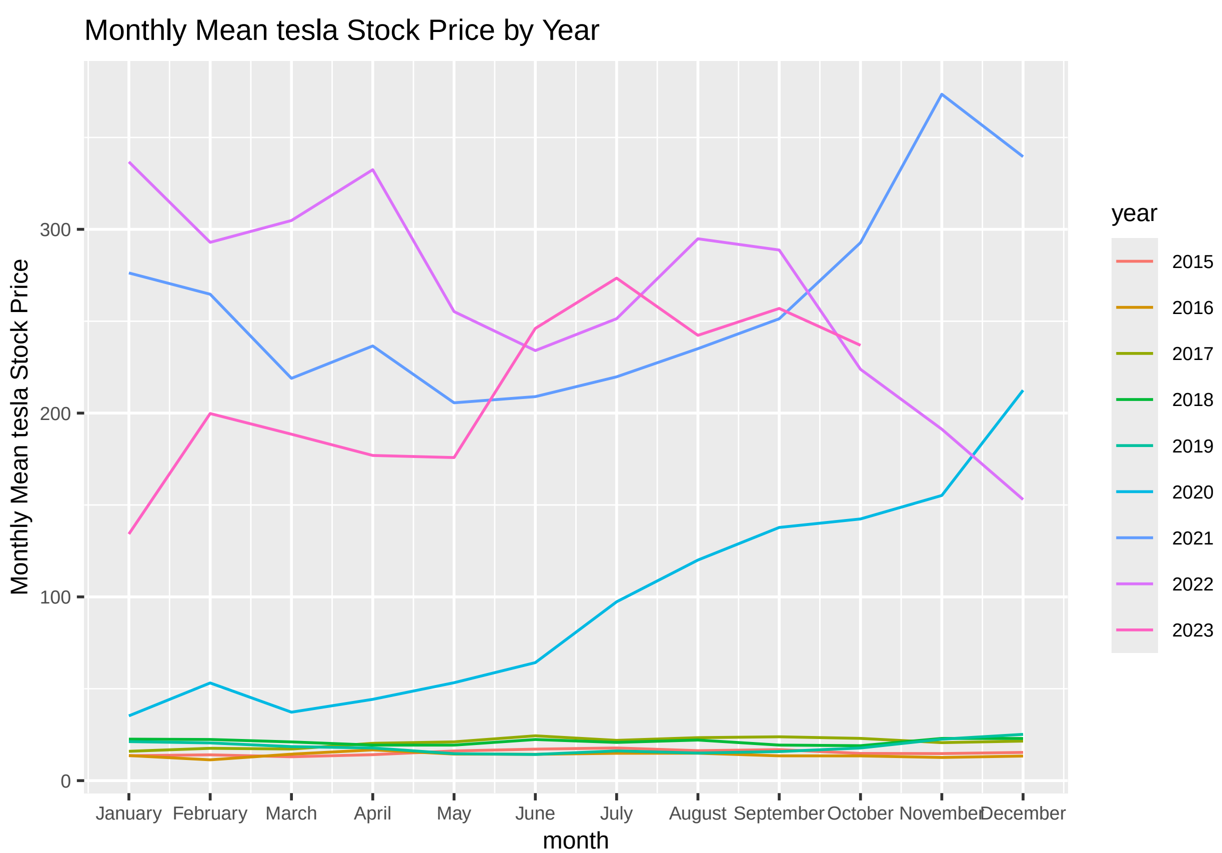
```
## # A tsibble: 6 x 10 [1D]
##   Date       close  open lowest highest total_vol mean_vol std_vol news    is_up
##   <date>     <dbl> <dbl>  <dbl>   <dbl>     <dbl>    <dbl>   <dbl> <chr>   <dbl>
## 1 2015-01-02  14.6  14.9   14.2    14.9  59157390  151686. 155761.  <NA>       0
## 2 2015-01-05  14.0  14.4   13.8    14.4  68662800  176058. 168291.  <NA>       1
## 3 2015-01-06  14.1  14.0   13.6    14.3  80752635  207058. 152662.  <NA>       0
## 4 2015-01-07  14.1  14.2   14.0    14.3  38728110   99558. 100907. "['BMW~     0
## 5 2015-01-08  14.0  14.2   14.0    14.3  43839960  112699. 130931. "['How~     0
## 6 2015-01-09  13.8  13.9   13.7    14.0  59398215  152695. 153615. "['How~     0
```

```
tesla_tsib_month_year <-tesla_tsib

monthly_price_avg_tsib <- tesla_tsib_month_year %>% index_by(yearMonth=yearmonth(Date)) %>% group_by(yea
monthly_price_avg_tsib$year <- year(monthly_price_avg_tsib$yearMonth)
monthly_price_avg_tsib$month <- month(monthly_price_avg_tsib$yearMonth)
monthly_price_avg <- as_tibble(monthly_price_avg_tsib) %>% select(mean_price, year, month)

monthly_price_avg$year <- as.character(monthly_price_avg$year)

monthly_price_avg
```

```
## # A tibble: 106 x 3
```

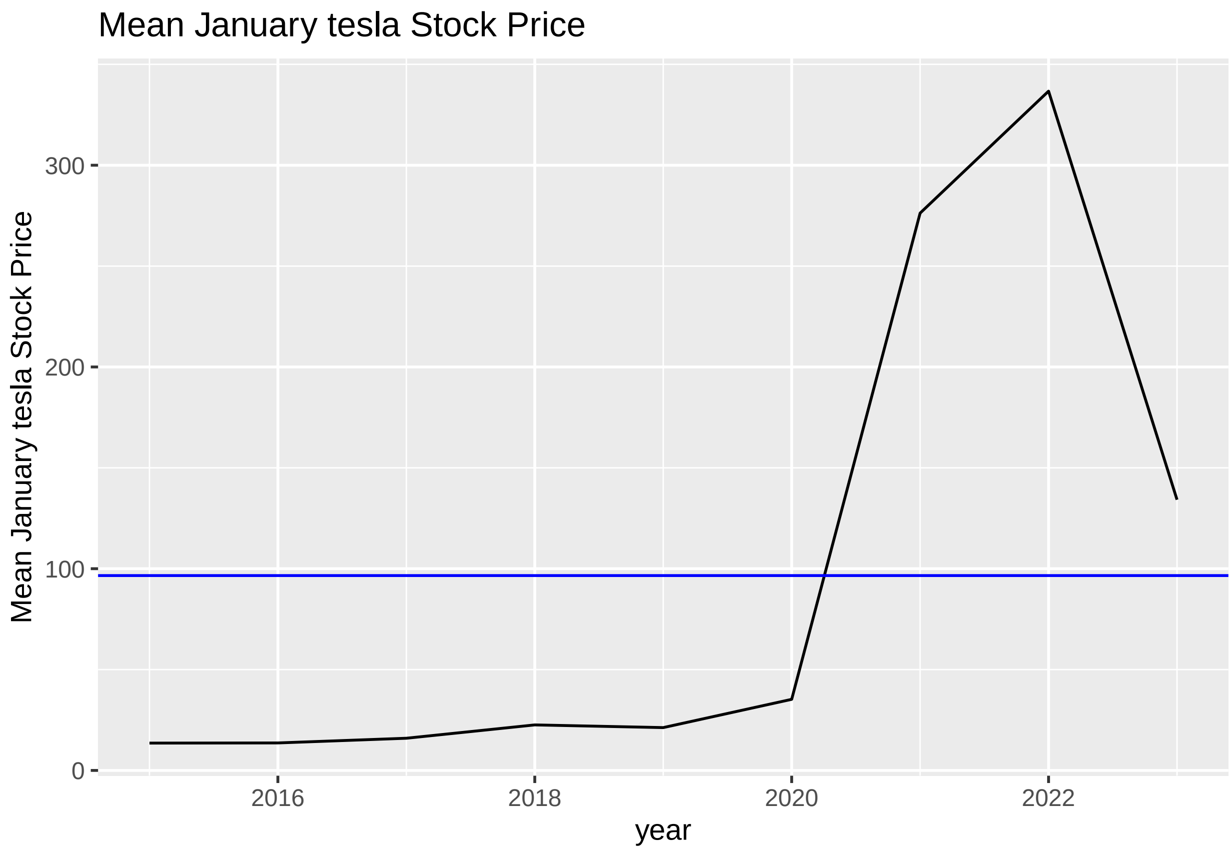
```
##    mean_price year  month
##         <dbl> <chr> <dbl>
##  1       13.5 2015      1
##  2       14.0 2015      2
##  3       13.0 2015      3
##  4       14.1 2015      4
##  5       16.1 2015      5
##  6       17.1 2015      6
##  7       17.8 2015      7
##  8       16.3 2015      8
##  9       16.9 2015      9
## 10       14.8 2015     10
## # i 96 more rows
```

```r
options(repr.plot.width =15, repr.plot.height =15)

monthly_avg_plot <- monthly_price_avg %>% ggplot(aes(x=month, y = mean_price, color= year)) +
  geom_line() + ylab("Monthly Mean tesla Stock Price") +
  scale_x_continuous(
    breaks = seq_along(month.name),
    labels = month.name
  ) +
  ggtitle('Monthly Mean tesla Stock Price by Year') + theme(text = element_text(size = 9))
monthly_avg_plot
```

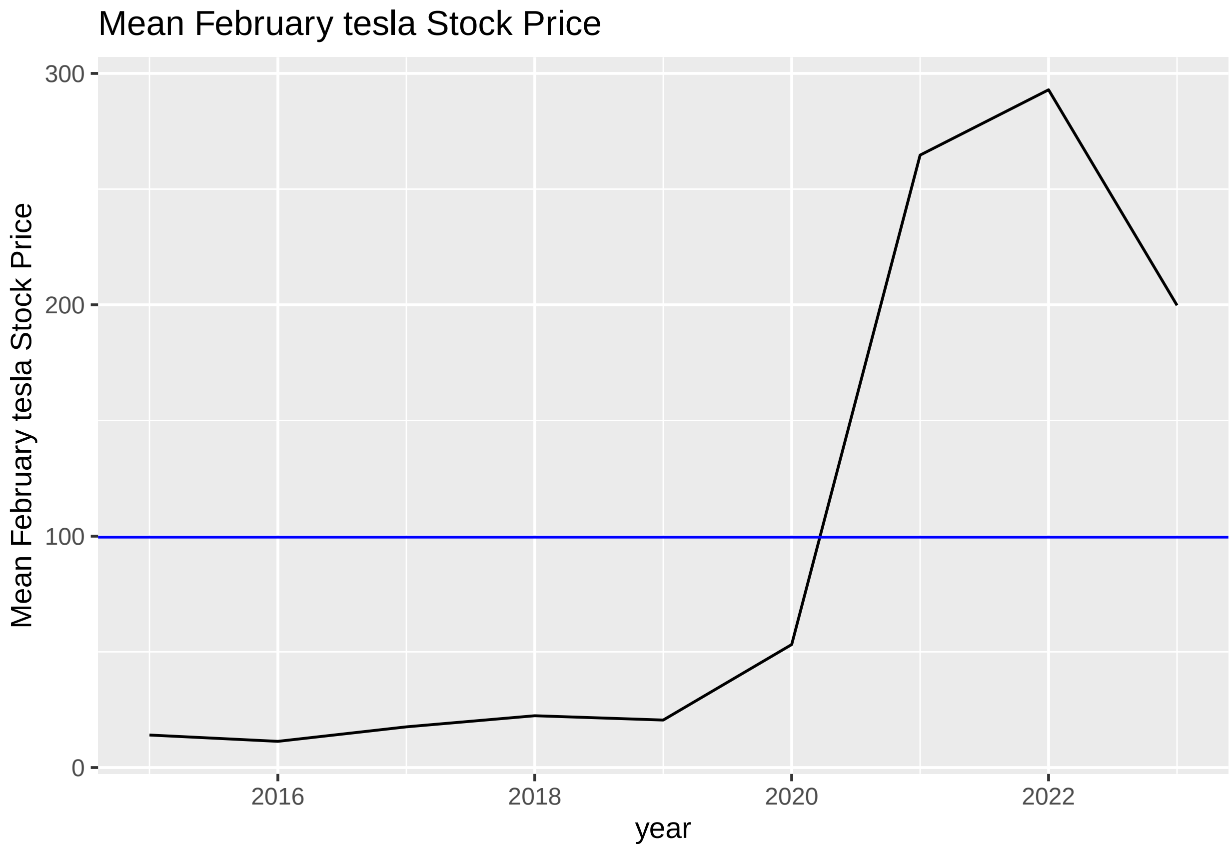


We see that tesla stock was at a low in 2016 before rising consistently from 2017 to 2020. We see that stocks fall at the end of 2021 and goes down in 2022. We cannot really see strong evidence of seasonality. However, we can later look at the components of the monthly average price to detect any seasonal component.

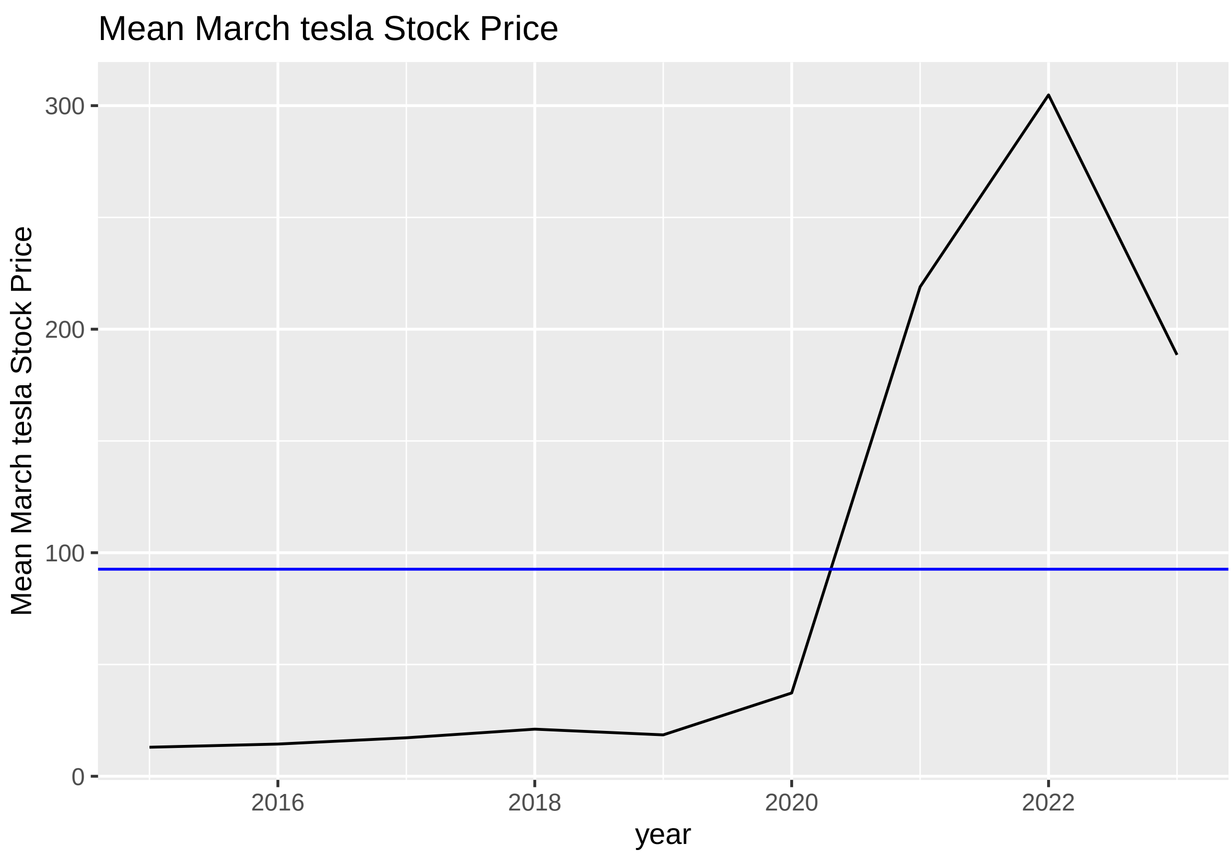
```
january_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>%
jan_tesla_price_plot <- january_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean January tesla Stock Price") +
  ggtitle('Mean January tesla Stock Price') + geom_hline(yintercept = mean(january_monthly_price$mean_pr
jan_tesla_price_plot
```



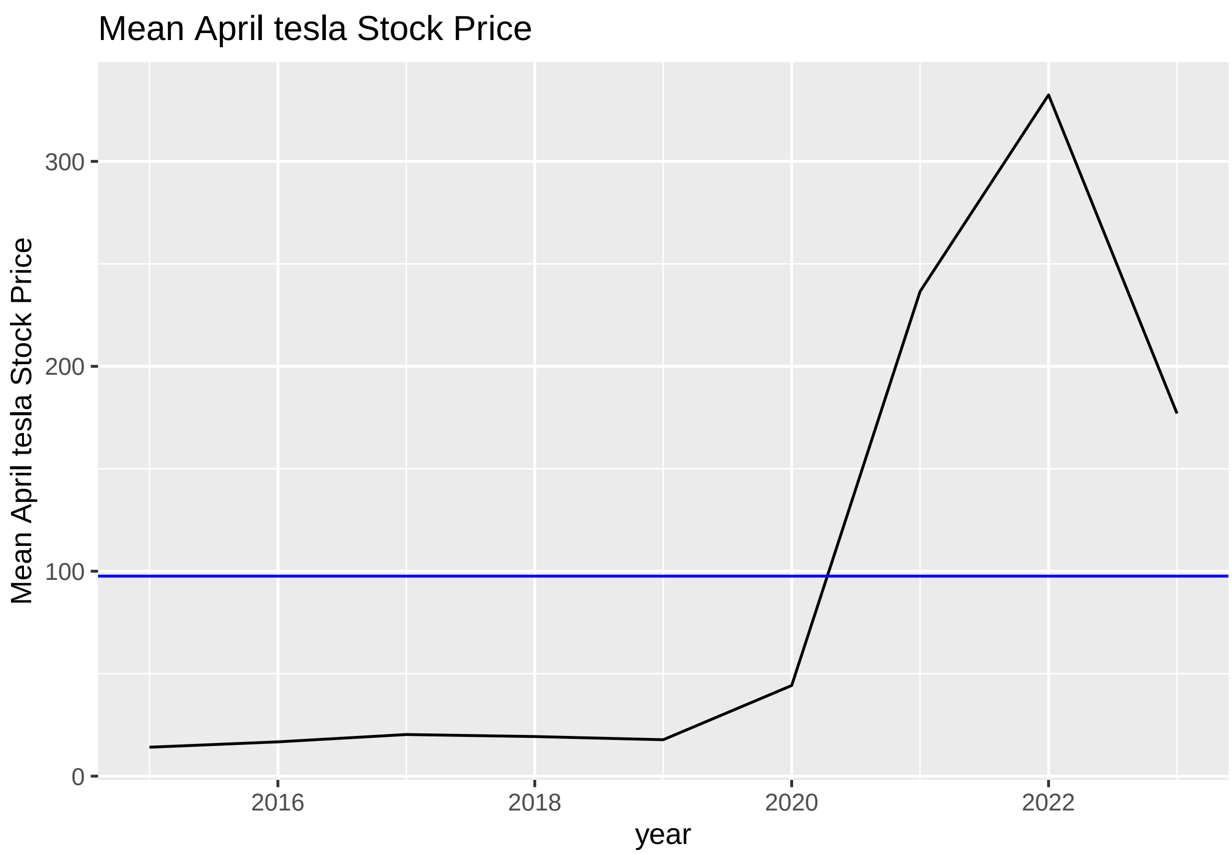

```
feb_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
feb_tesla_price_plot <- feb_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean February tesla Stock Price") +
  ggtitle('Mean February tesla Stock Price') + geom_hline(yintercept = mean(feb_monthly_price$mean_pric
feb_tesla_price_plot
```



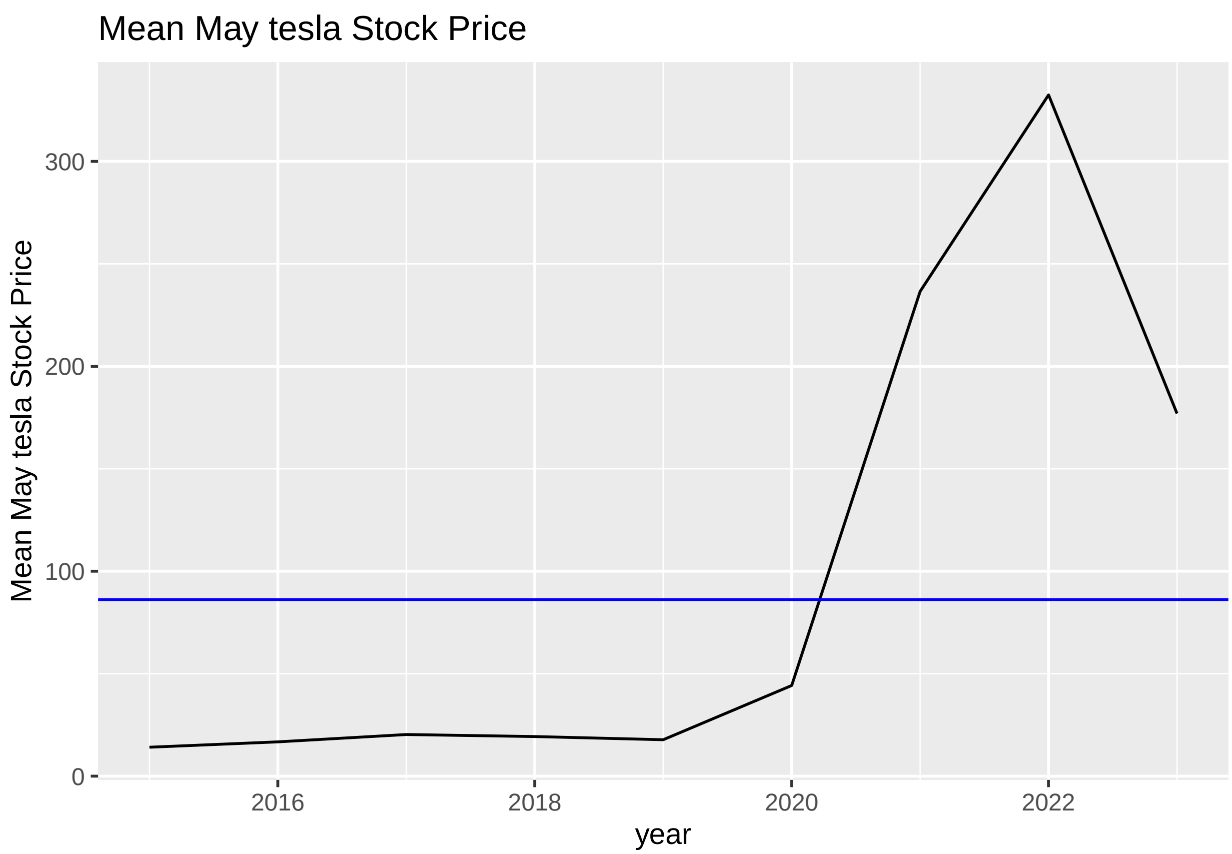

```
mar_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
mar_tesla_price_plot <- mar_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean March tesla Stock Price") +
  ggtitle('Mean March tesla Stock Price') + geom_hline(yintercept = mean(mar_monthly_price$mean_price),
mar_tesla_price_plot
```



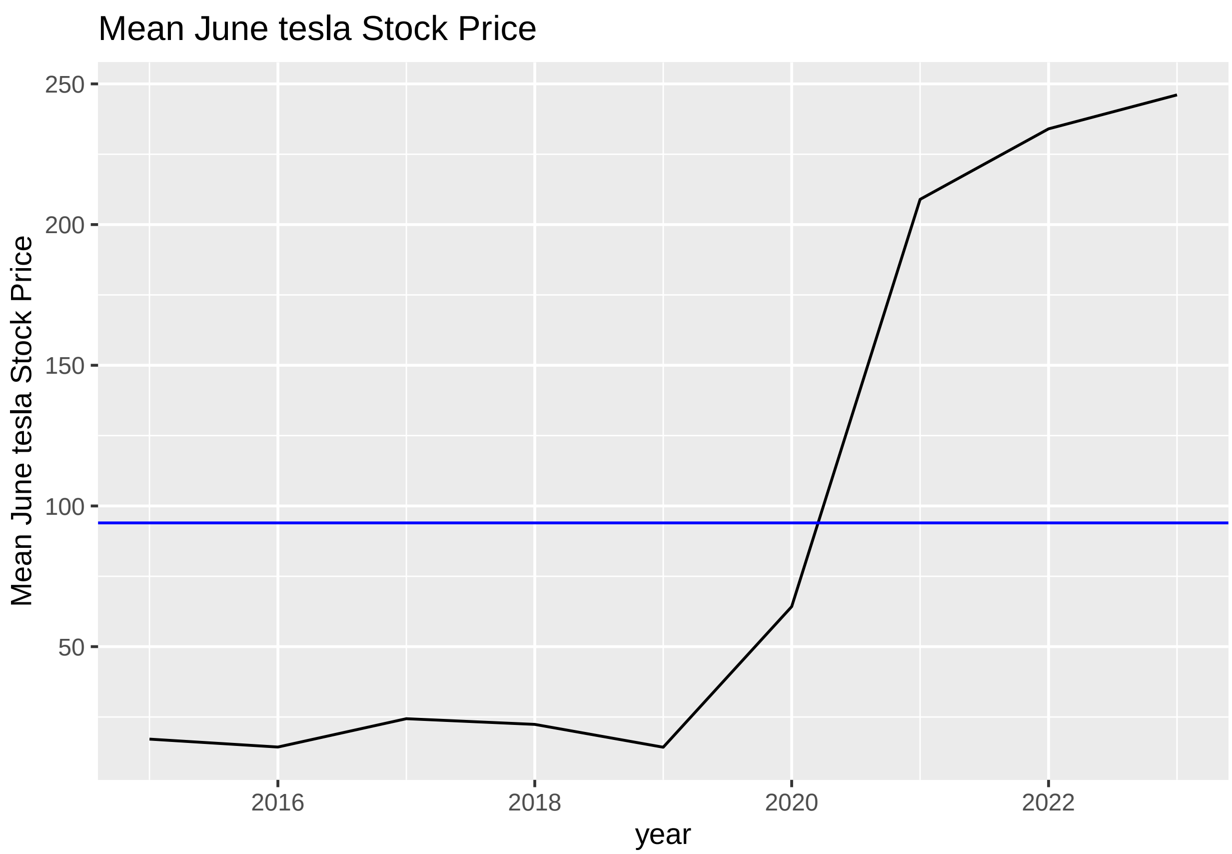

```
apr_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
apr_tesla_price_plot <- apr_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean April tesla Stock Price") +
  ggtitle('Mean April tesla Stock Price') + geom_hline(yintercept = mean(apr_monthly_price$mean_price),
apr_tesla_price_plot
```



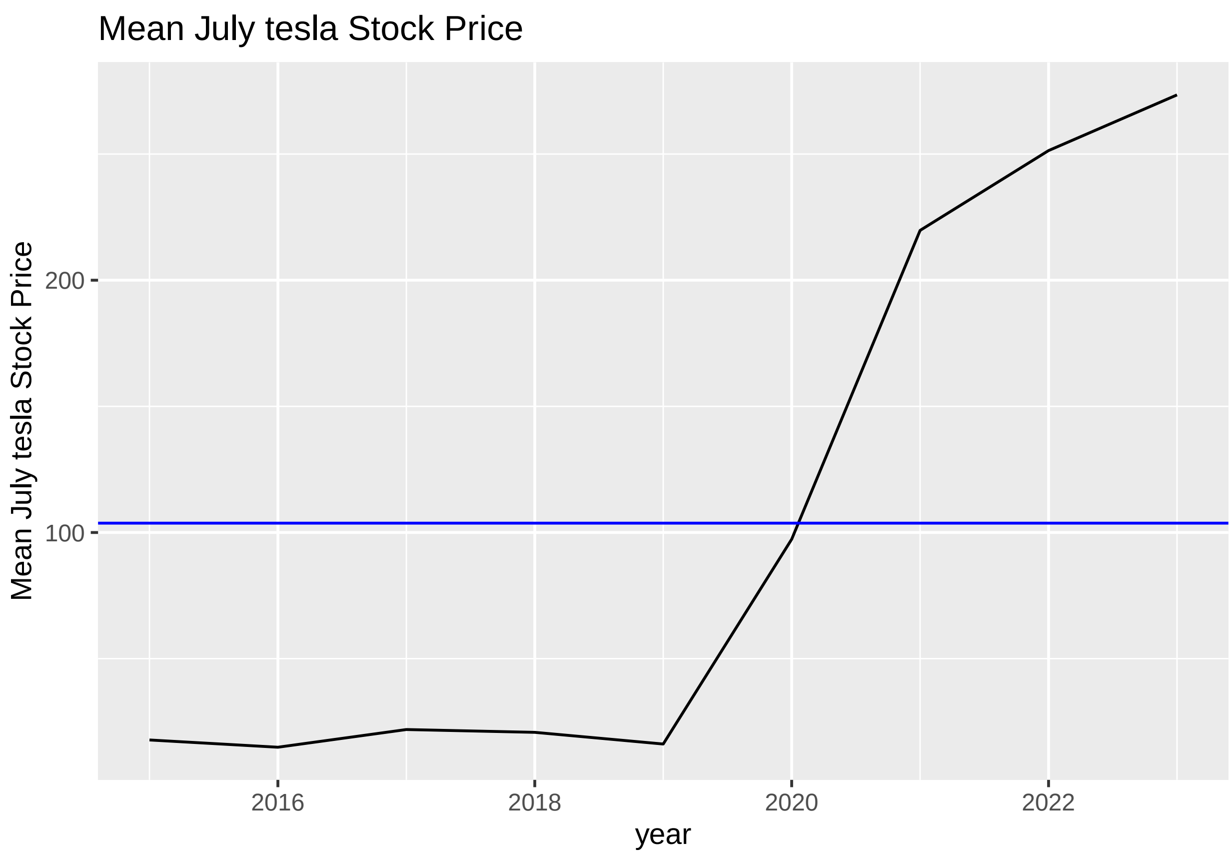

```
may_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
may_tesla_price_plot <- apr_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean May tesla Stock Price") +
  ggtitle('Mean May tesla Stock Price') + geom_hline(yintercept = mean(may_monthly_price$mean_price), co
may_tesla_price_plot
```



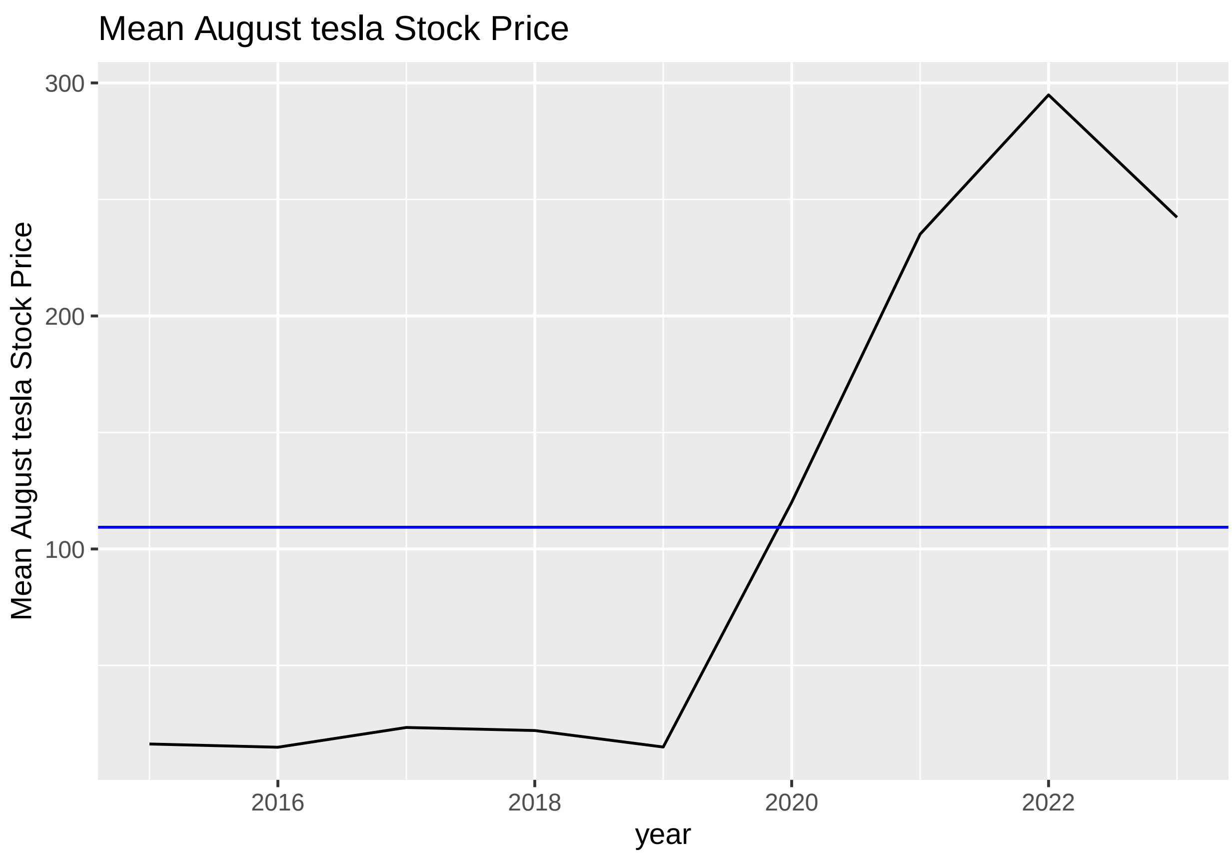

```
jun_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
jun_tesla_price_plot <- jun_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean June tesla Stock Price") +
  ggtitle('Mean June tesla Stock Price') + geom_hline(yintercept = mean(jun_monthly_price$mean_price), 
jun_tesla_price_plot
```



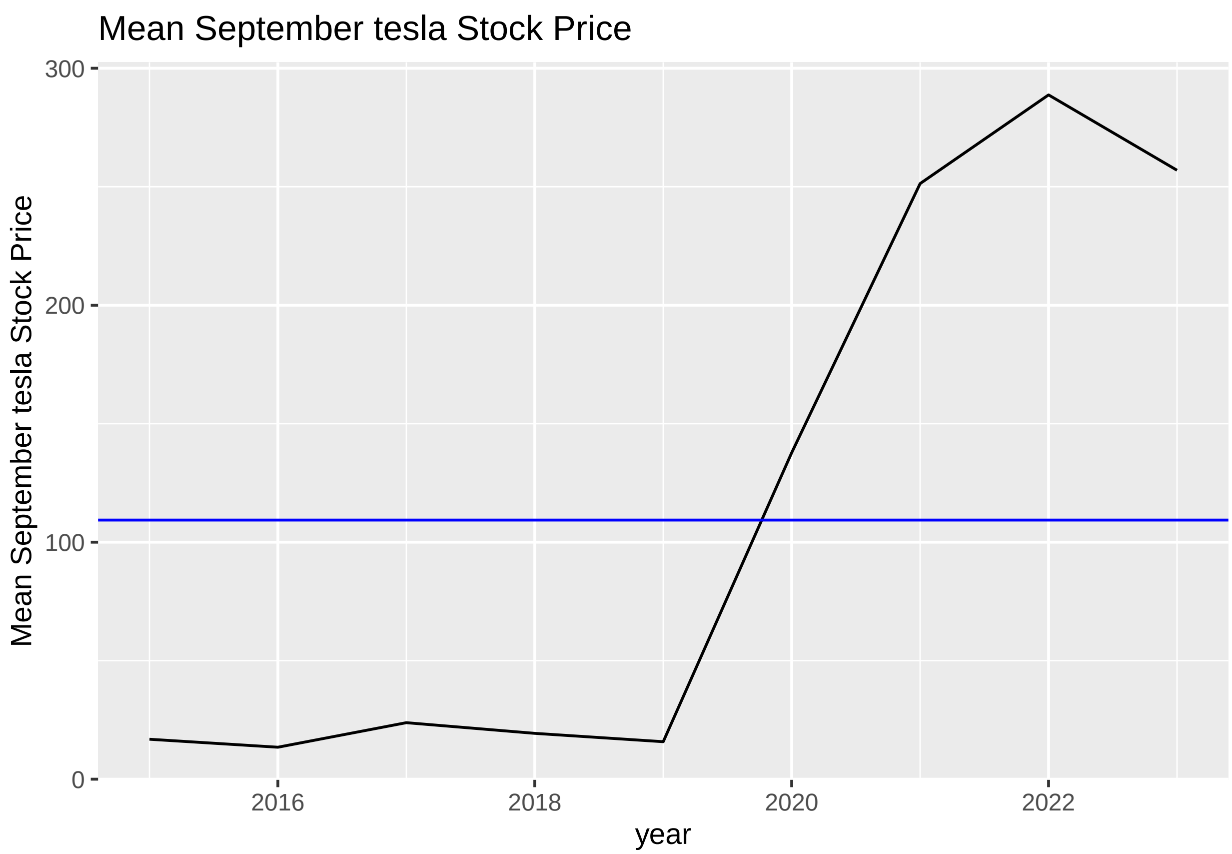

```
jul_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
jul_tesla_price_plot <- jul_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean July tesla Stock Price") +
  ggtitle('Mean July tesla Stock Price') + geom_hline(yintercept = mean(jul_monthly_price$mean_price), c
jul_tesla_price_plot
```



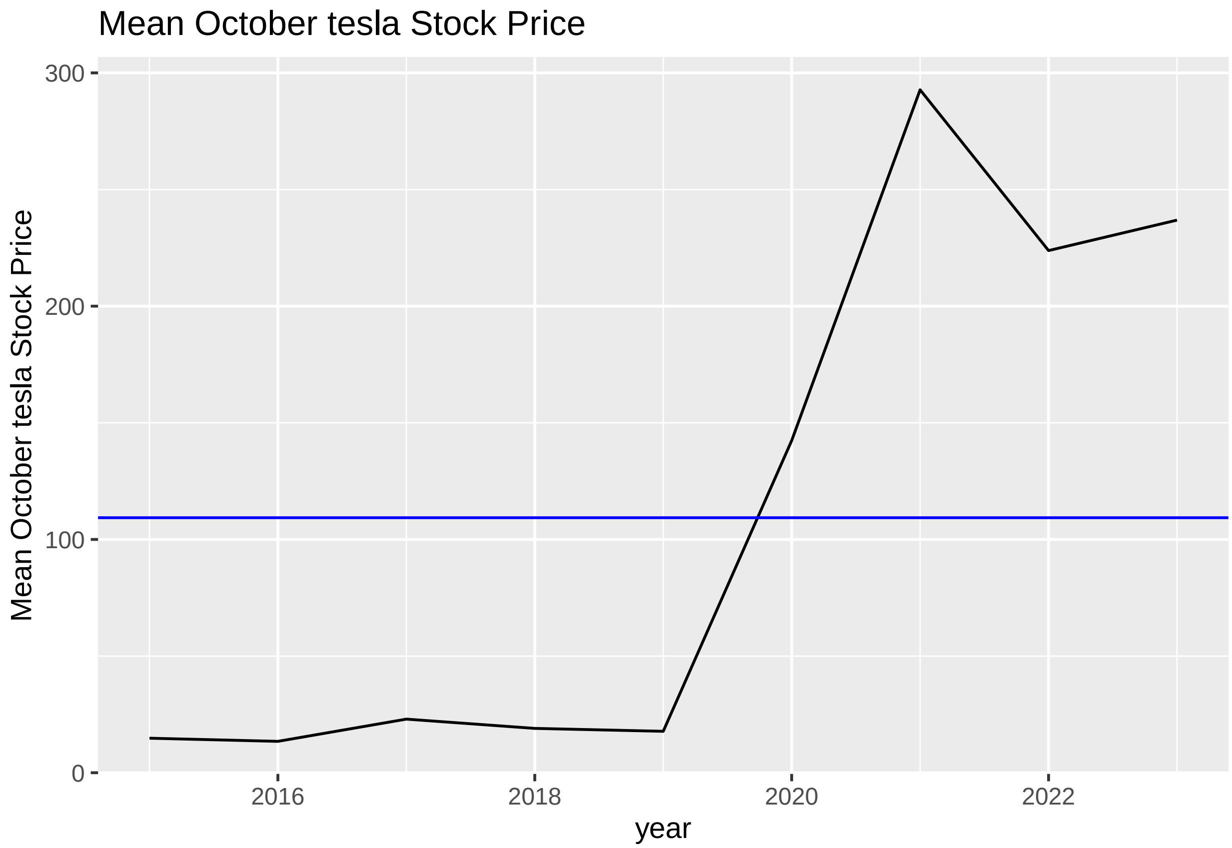

```
aug_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
aug_tesla_price_plot <- aug_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean August tesla Stock Price") +
  ggtitle('Mean August tesla Stock Price') + geom_hline(yintercept = mean(aug_monthly_price$mean_price)
aug_tesla_price_plot
```



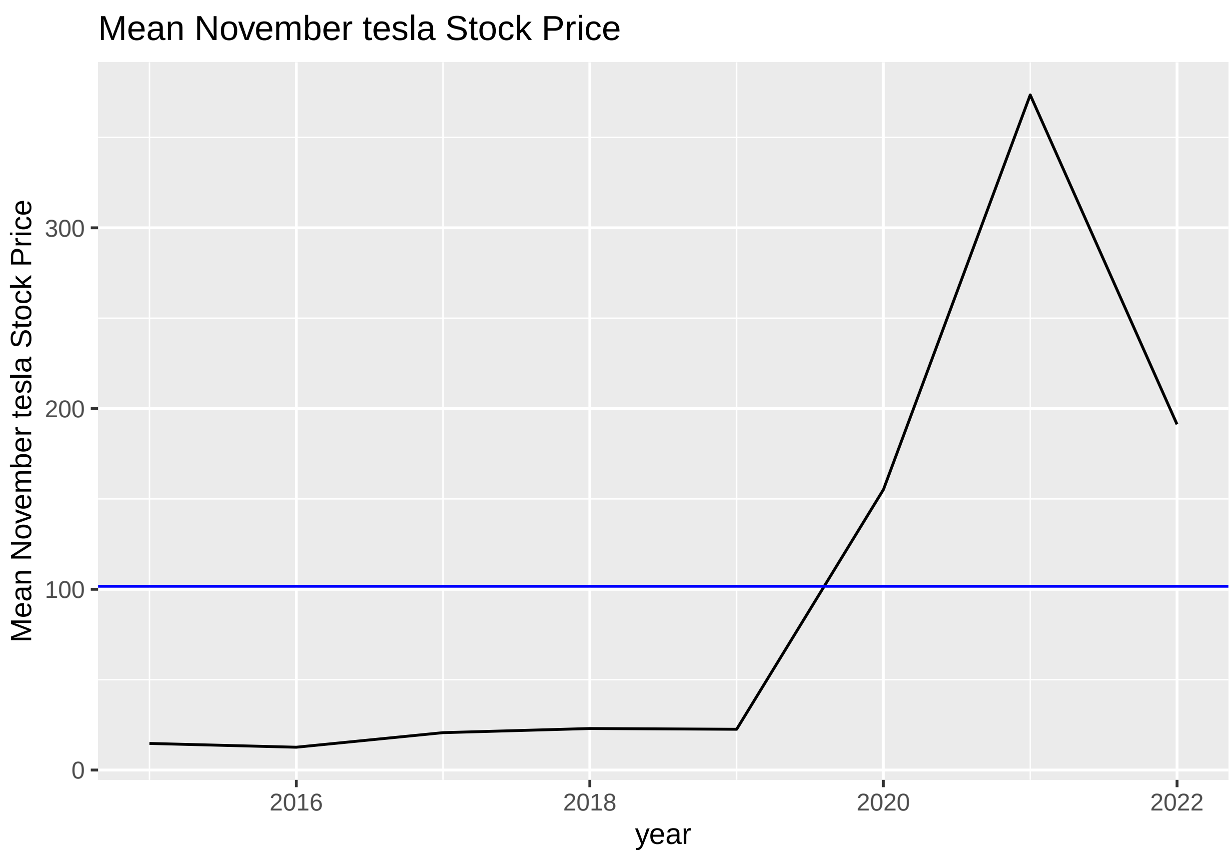

```
sep_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
sep_tesla_price_plot <- sep_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean September tesla Stock Price") +
  ggtitle('Mean September tesla Stock Price') + geom_hline(yintercept = mean(aug_monthly_price$mean_pric
sep_tesla_price_plot
```



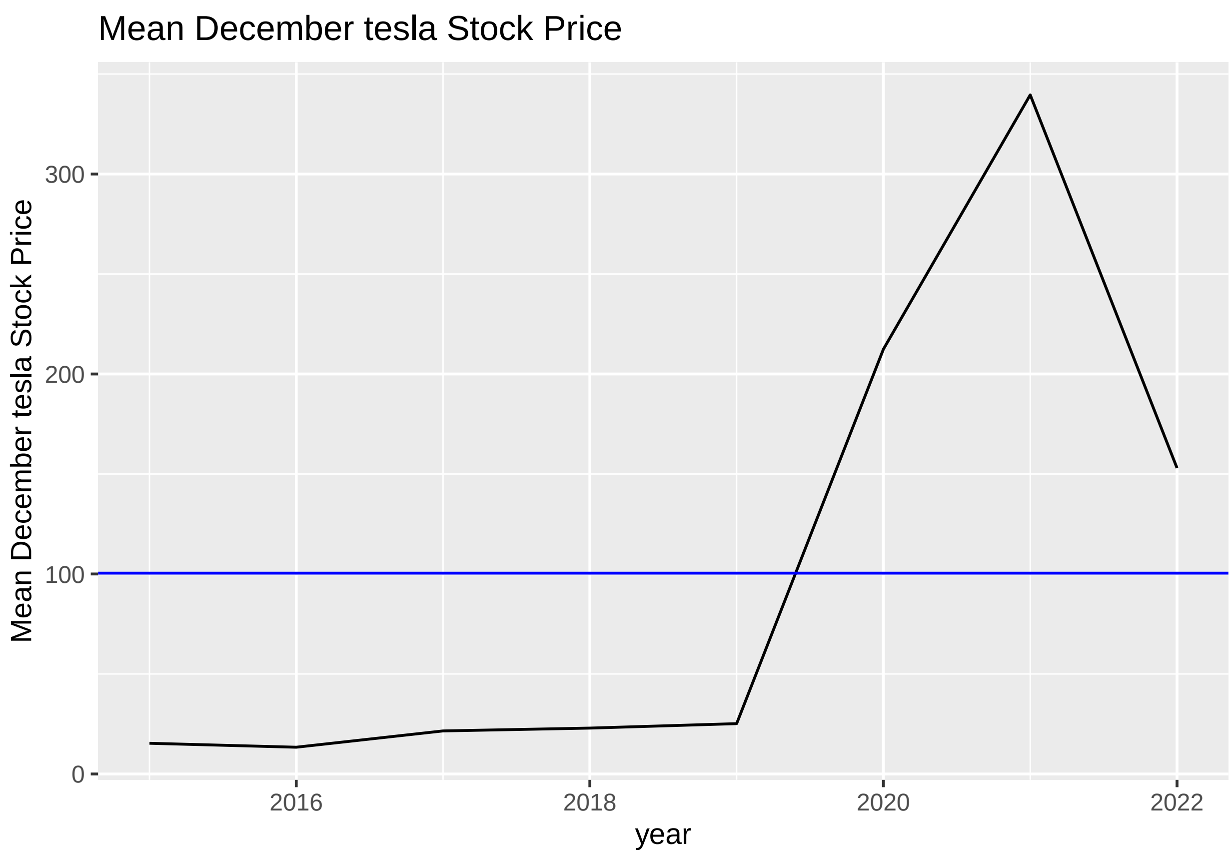

```
oct_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
oct_tesla_price_plot <- oct_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean October tesla Stock Price") +
  ggtitle('Mean October tesla Stock Price') + geom_hline(yintercept = mean(oct_monthly_price$mean_price)
oct_tesla_price_plot
```



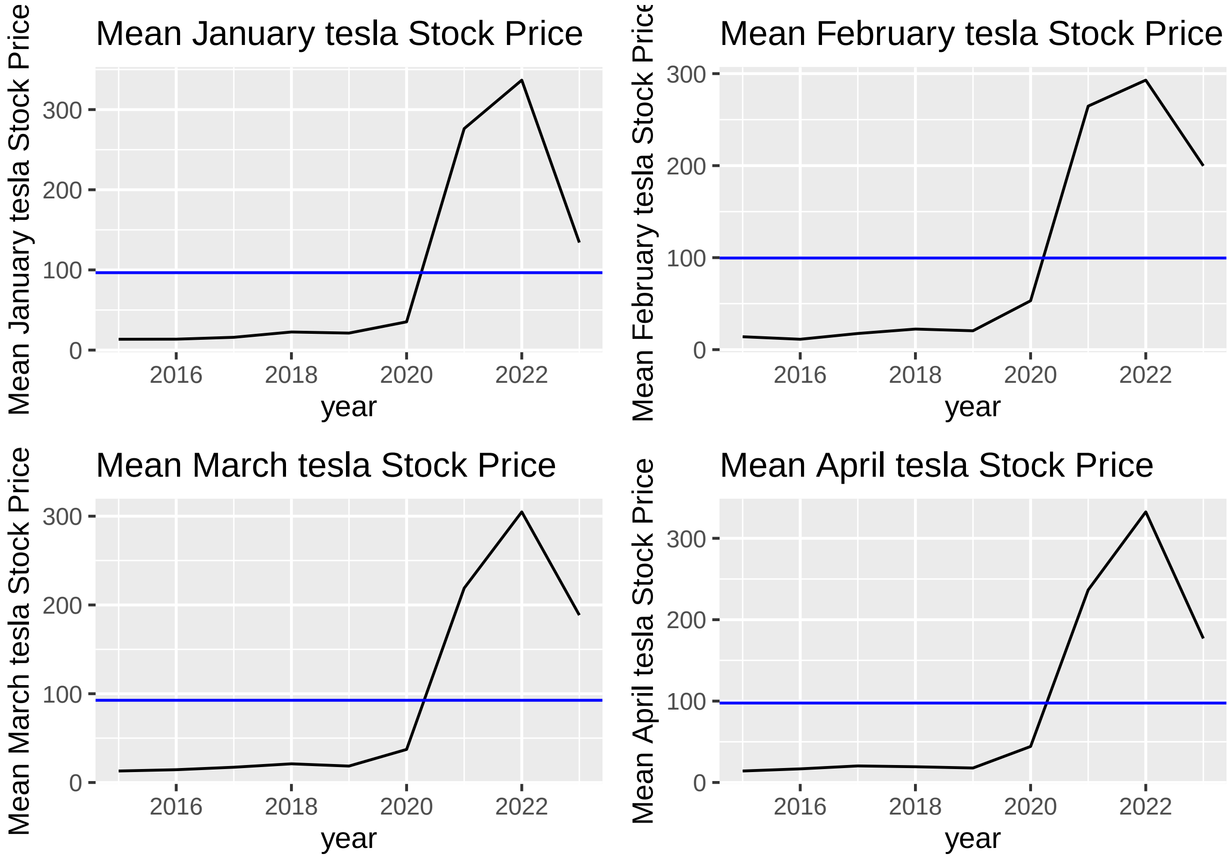

```
nov_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
nov_tesla_price_plot <- nov_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean November tesla Stock Price") +
  ggtitle('Mean November tesla Stock Price') + geom_hline(yintercept = mean(nov_monthly_price$mean_pric
nov_tesla_price_plot
```




```
dec_monthly_price <- monthly_price_avg_tsib %>% as_tibble() %>% select(mean_price, year, month) %>% fil
dec_tesla_price_plot <- dec_monthly_price %>% ggplot(aes(x=year, y = mean_price)) +
  geom_line() + ylab("Mean December tesla Stock Price") +
  ggtitle('Mean December tesla Stock Price') + geom_hline(yintercept = mean(dec_monthly_price$mean_pric
dec_tesla_price_plot
```

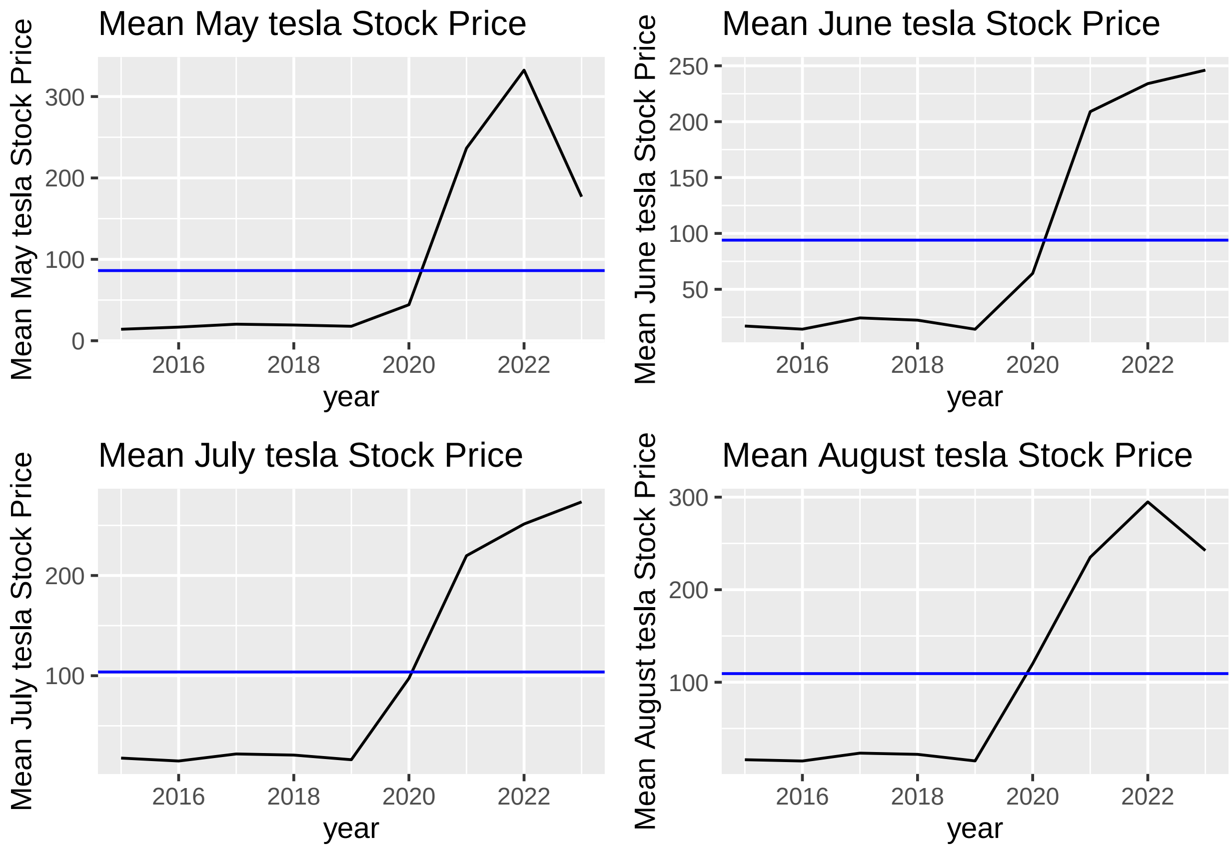



```
grid.arrange(jan_tesla_price_plot, feb_tesla_price_plot, mar_tesla_price_plot, apr_tesla_price_plot, nr
```



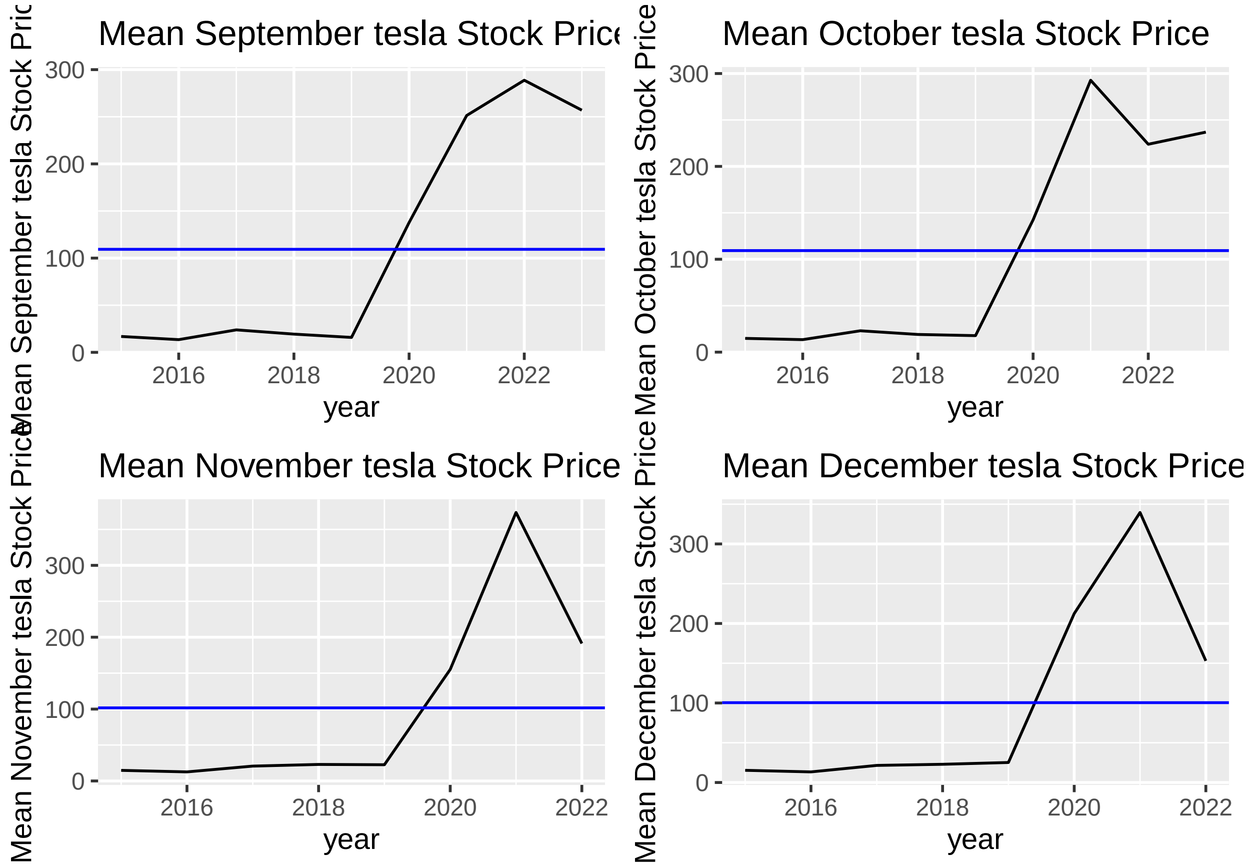
We see that average monthly tesla stock prices rise slightly from January to April, especially in 2022, then drop off afterward.

```
grid.arrange(may_tesla_price_plot, jun_tesla_price_plot, jul_tesla_price_plot, aug_tesla_price_plot, nr
```



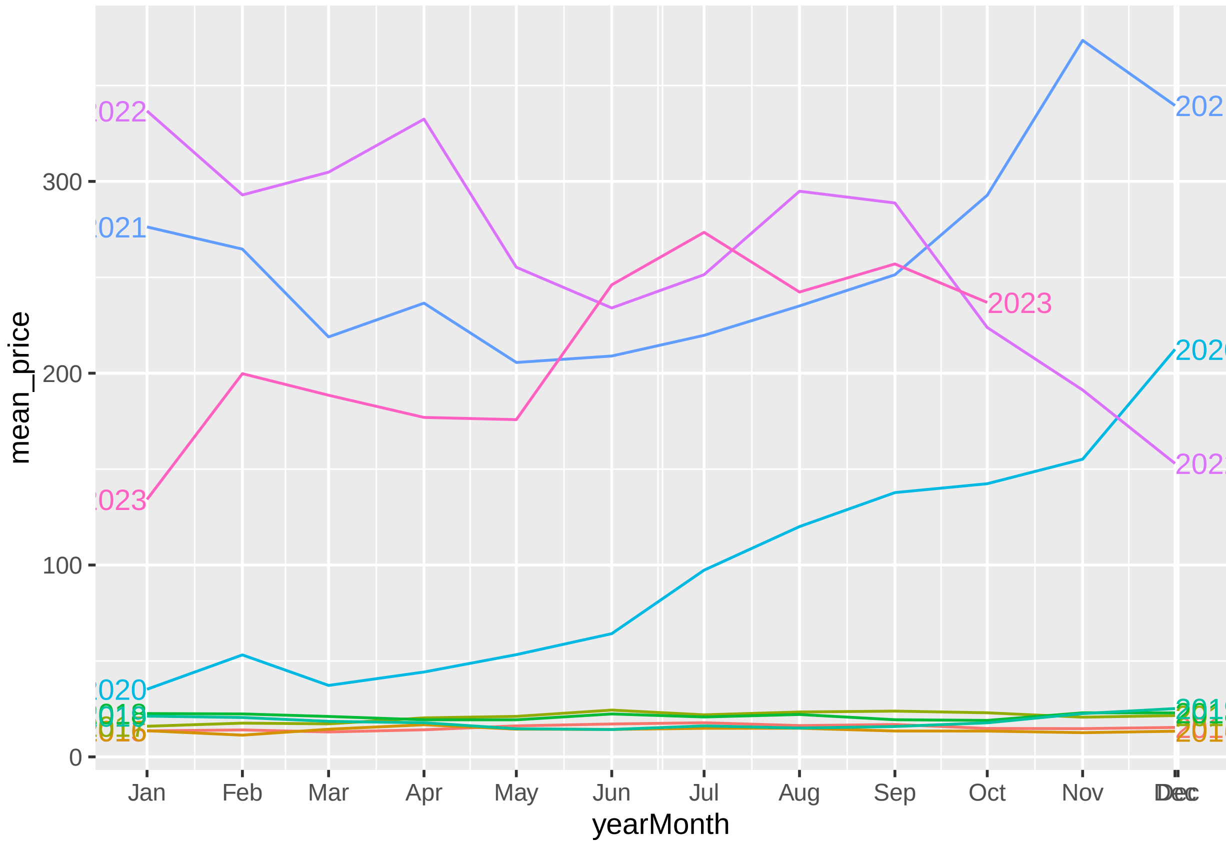
Similar story here.

```
grid.arrange(sep_tesla_price_plot, oct_tesla_price_plot, nov_tesla_price_plot, dec_tesla_price_plot, nr
```

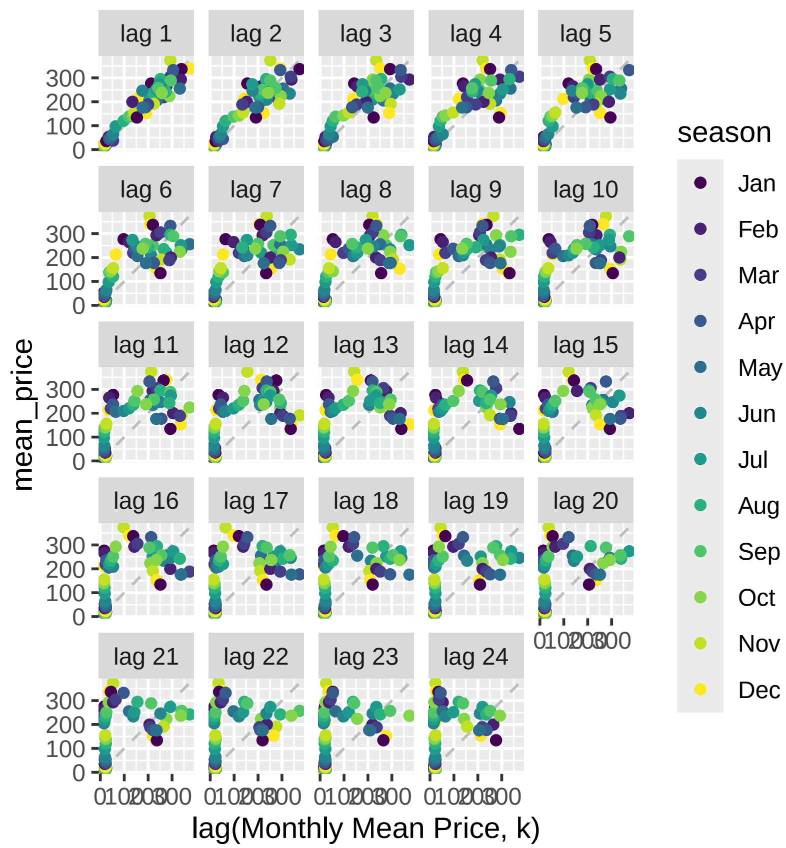


Same pattern but the peak appears in 2021.

```
monthly_price_avg_tsib |>
  gg_season(mean_price, labels = "both")
```



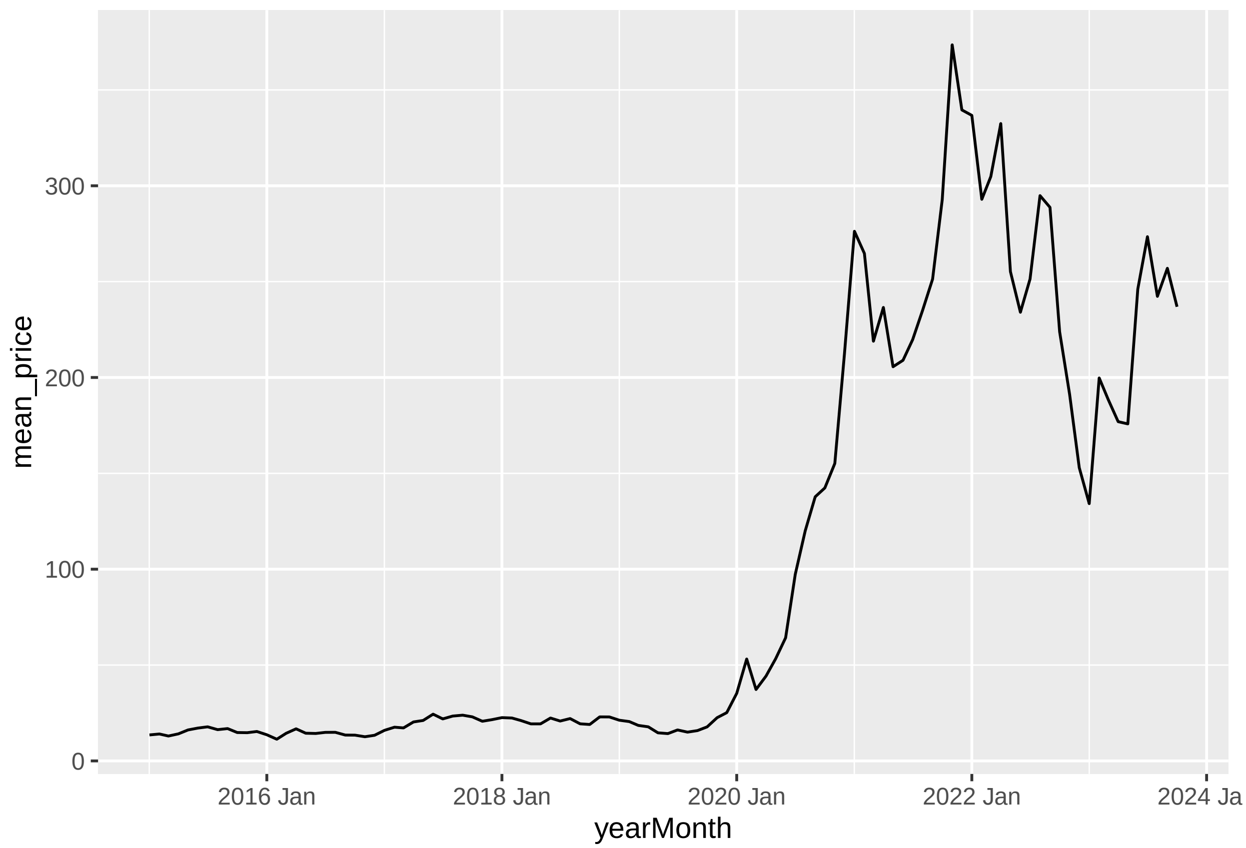

```
monthly_price_avg_tsib |>
  gg_lag(mean_price, lags=1:24, geom = "point") +
  labs(x = "lag(Monthly Mean Price, k)")
```




```
monthly_price_avg_tsib
```

```
## # A tsibble: 106 x 4 [1M]
##    yearMonth mean_price  year month
##        <mth>      <dbl> <dbl> <dbl>
##  1  2015 Jan       13.5  2015     1
##  2  2015 Feb       14.0  2015     2
##  3  2015 Mar       13.0  2015     3
##  4  2015 Apr       14.1  2015     4
##  5  2015 May       16.1  2015     5
##  6  2015 Jun       17.1  2015     6
##  7  2015 Jul       17.8  2015     7
##  8  2015 Aug       16.3  2015     8
##  9  2015 Sep       16.9  2015     9
## 10  2015 Oct       14.8  2015    10
## # i 96 more rows
```

```
monthly_price_avg_tsib |> ggplot(aes(x=yearMonth, y = mean_price)) +
  geom_line()
```




```
tesla_tsib
```

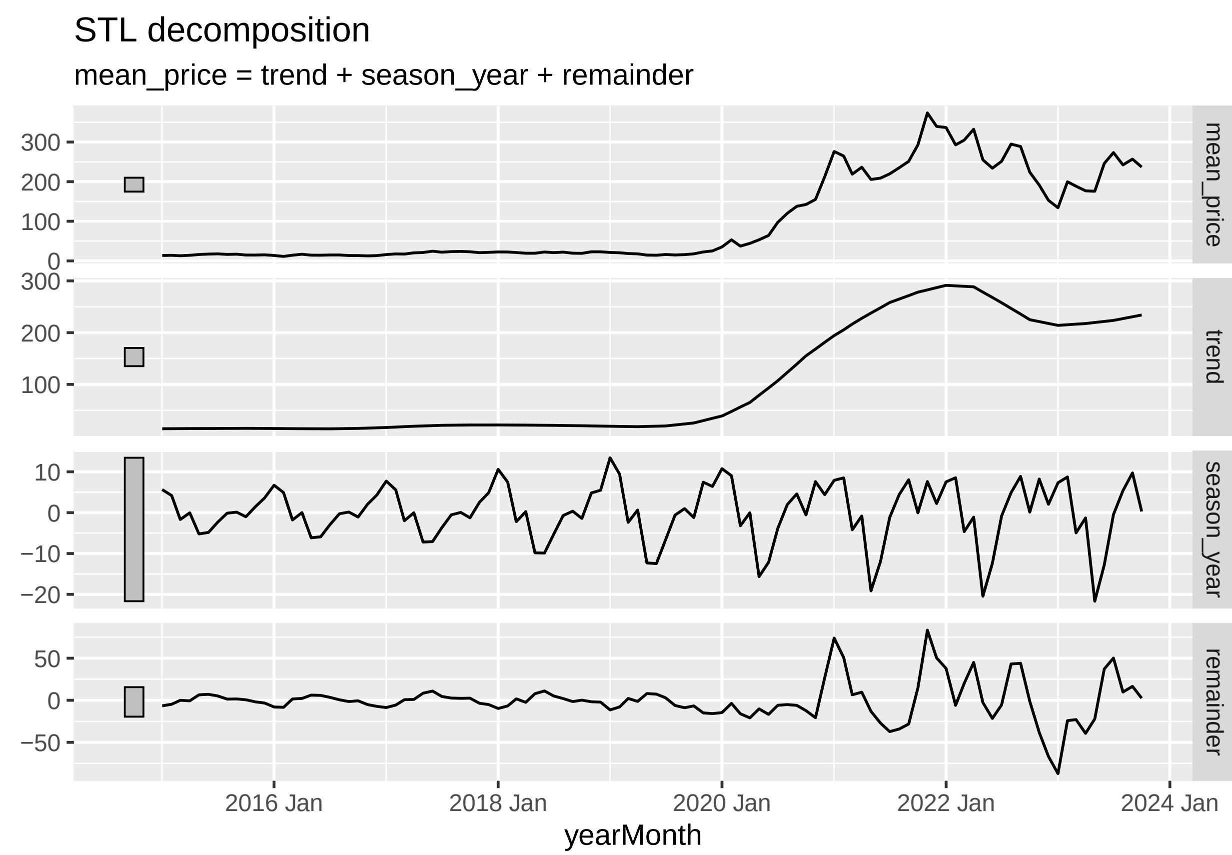
```
## # A tsibble: 2,223 x 10 [1D]
##    Date       close  open lowest highest total_vol mean_vol std_vol news   is_up
##    <date>     <dbl> <dbl>  <dbl>   <dbl>     <dbl>    <dbl>   <dbl> <chr>  <dbl>
##  1 2015-01-02  14.6  14.9   14.2    14.9  59157390  151686. 155761.  <NA>      0
##  2 2015-01-05  14.0  14.4   13.8    14.4  68662800  176058. 168291.  <NA>      1
##  3 2015-01-06  14.1  14.0   13.6    14.3  80752635  207058. 152662.  <NA>      0
##  4 2015-01-07  14.1  14.2   14.0    14.3  38728110   99558. 100907. "['BM~     0
##  5 2015-01-08  14.0  14.2   14.0    14.3  43839960  112699. 130931. "['Ho~     0
##  6 2015-01-09  13.8  13.9   13.7    14.0  59398215  152695. 153615. "['Ho~     0
##  7 2015-01-12  13.5  13.6   13.3    13.6  72115905  184913. 210097. "['Ho~     1
##  8 2015-01-13  13.6  13.6   13.4    13.8  49602285  127512. 127221. "['Ho~     0
##  9 2015-01-14  12.8  12.7   12.3    13.0 137698725  353074. 550815. "['Ho~     0
## 10 2015-01-15  12.8  13.0   12.7    13.0  67520070  173128. 164068. "['Ho~     1
## # i 2,213 more rows
```

```
dcmp <- monthly_price_avg_tsib |>
  model(stl = STL(mean_price))
components(dcmp)
```

```
## # A dable: 106 x 7 [1M]
## # Key:      .model [1]
## # :         mean_price = trend + season_year + remainder
##    .model yearMonth mean_price trend season_year remainder season_adjust
```

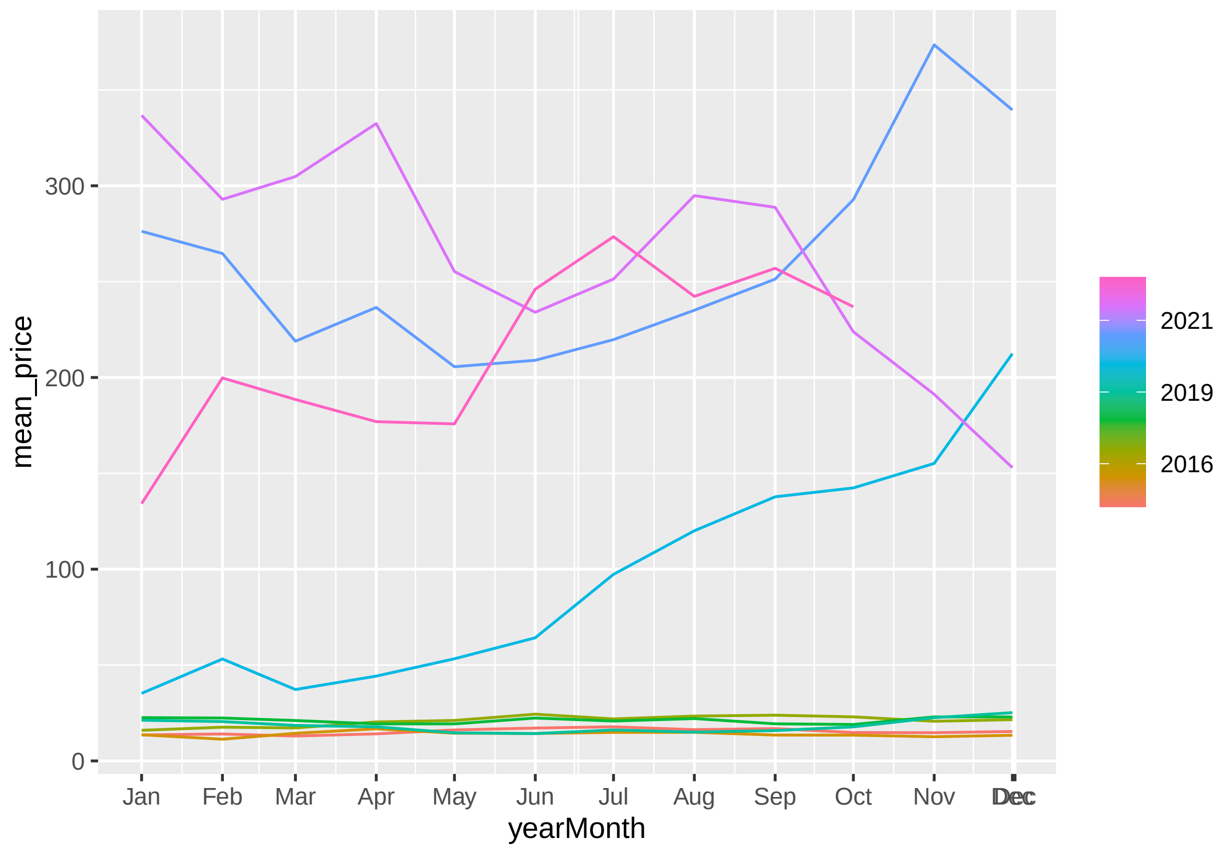
```
##     <chr>       <mth>       <dbl> <dbl>      <dbl>     <dbl>          <dbl>
##   1 stl      2015 Jan     13.5  14.5       5.64     -6.59            7.90
##   2 stl      2015 Feb     14.0  14.6       4.17     -4.72            9.88
##   3 stl      2015 Mar     13.0  14.7      -1.69     -0.0220         14.7
##   4 stl      2015 Apr     14.1  14.8      -0.0440   -0.636          14.2
##   5 stl      2015 May     16.1  14.8      -5.21      6.51           21.4
##   6 stl      2015 Jun     17.1  14.9      -4.84      7.08           22.0
##   7 stl      2015 Jul     17.8  14.9      -2.35      5.18           20.1
##   8 stl      2015 Aug     16.3  15.0      -0.135     1.40           16.4
##   9 stl      2015 Sep     16.9  15.1       0.118     1.61           16.7
##  10 stl      2015 Oct     14.8  15.2      -1.00      0.579          15.8
## # i 96 more rows
```

```
components(dcmp) |> autoplot()
```



There seems to be a peak in monthly average stock prices every starts of every year (around January), with peaks increasing and troughs decreasing (increasing variability) as we move from left to right.

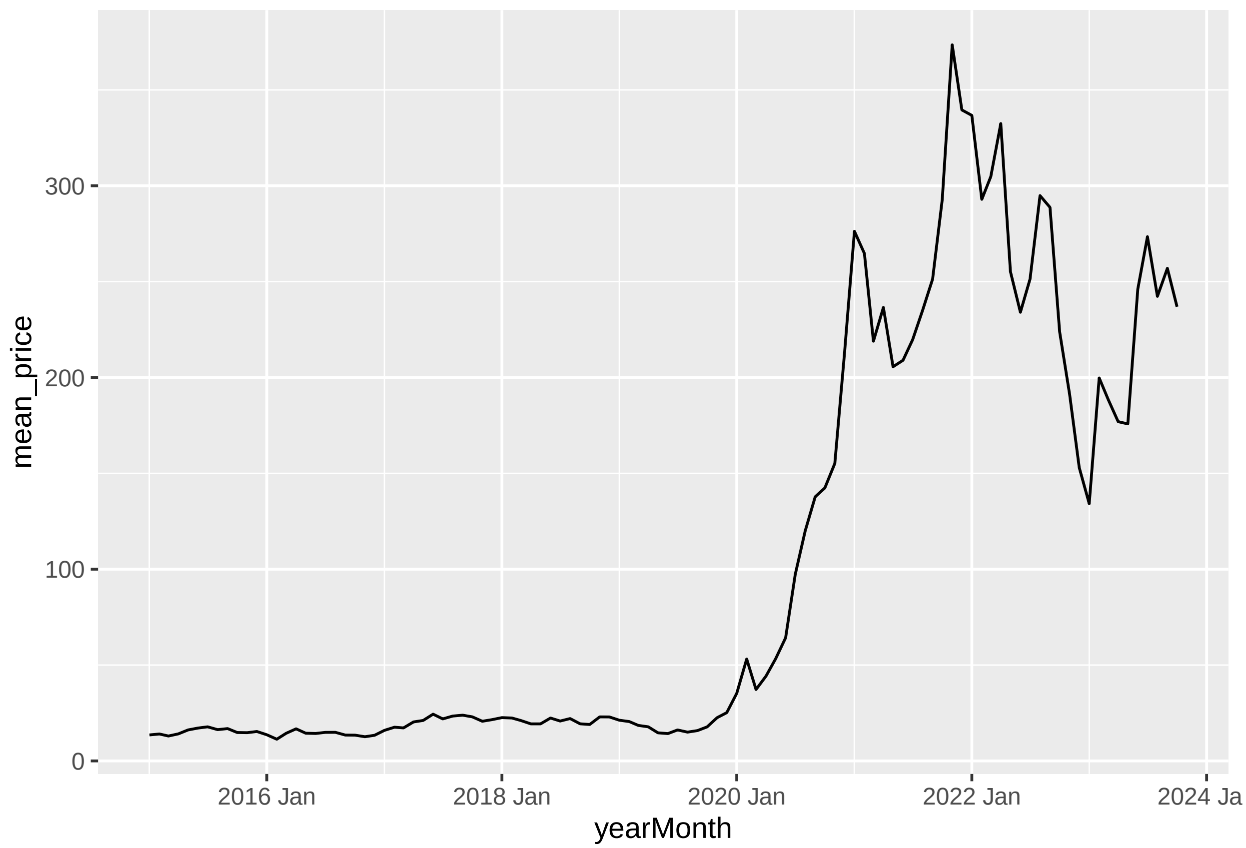
```
monthly_price_avg_tsib |>
  gg_season(mean_price, period = "year")
```




```r
monthly_price_avg_tsib
```

```
## # A tsibble: 106 x 4 [1M]
##    yearMonth mean_price  year month
##        <mth>      <dbl> <dbl> <dbl>
##  1  2015 Jan       13.5  2015     1
##  2  2015 Feb       14.0  2015     2
##  3  2015 Mar       13.0  2015     3
##  4  2015 Apr       14.1  2015     4
##  5  2015 May       16.1  2015     5
##  6  2015 Jun       17.1  2015     6
##  7  2015 Jul       17.8  2015     7
##  8  2015 Aug       16.3  2015     8
##  9  2015 Sep       16.9  2015     9
## 10  2015 Oct       14.8  2015    10
## # i 96 more rows
```

```r
monthly_price_avg_tsib |> ggplot(aes(x=yearMonth, y = mean_price)) +
  geom_line()
```

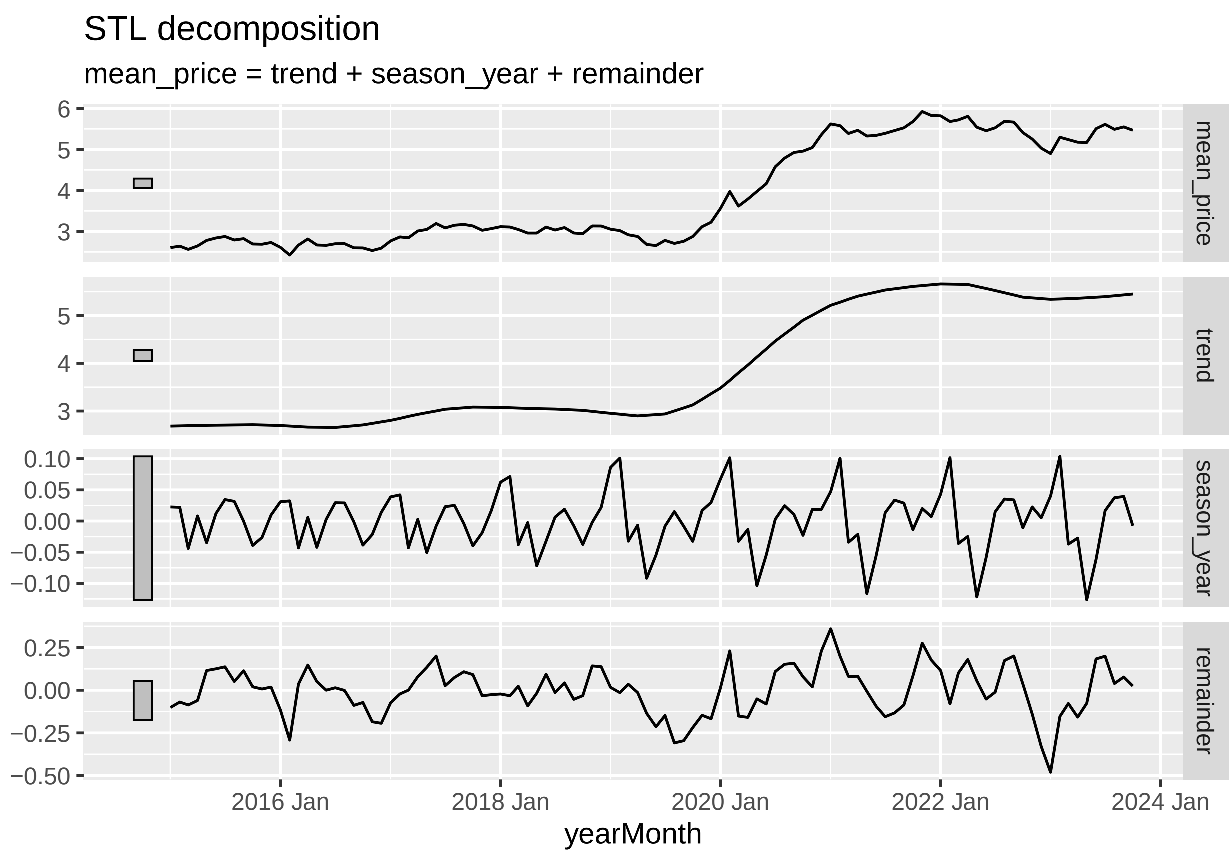


To stabilize the variance, we apply a Box-Cox transformation (log).

```
monthly_price_avg_tsib_transformed <- monthly_price_avg_tsib %>% mutate(mean_price = log(mean_price))
monthly_price_avg_tsib_transformed <- monthly_price_avg_tsib_transformed %>% select(mean_price)

dcmp <- monthly_price_avg_tsib_transformed |>
  model(stl = STL(mean_price))
components(dcmp)
```

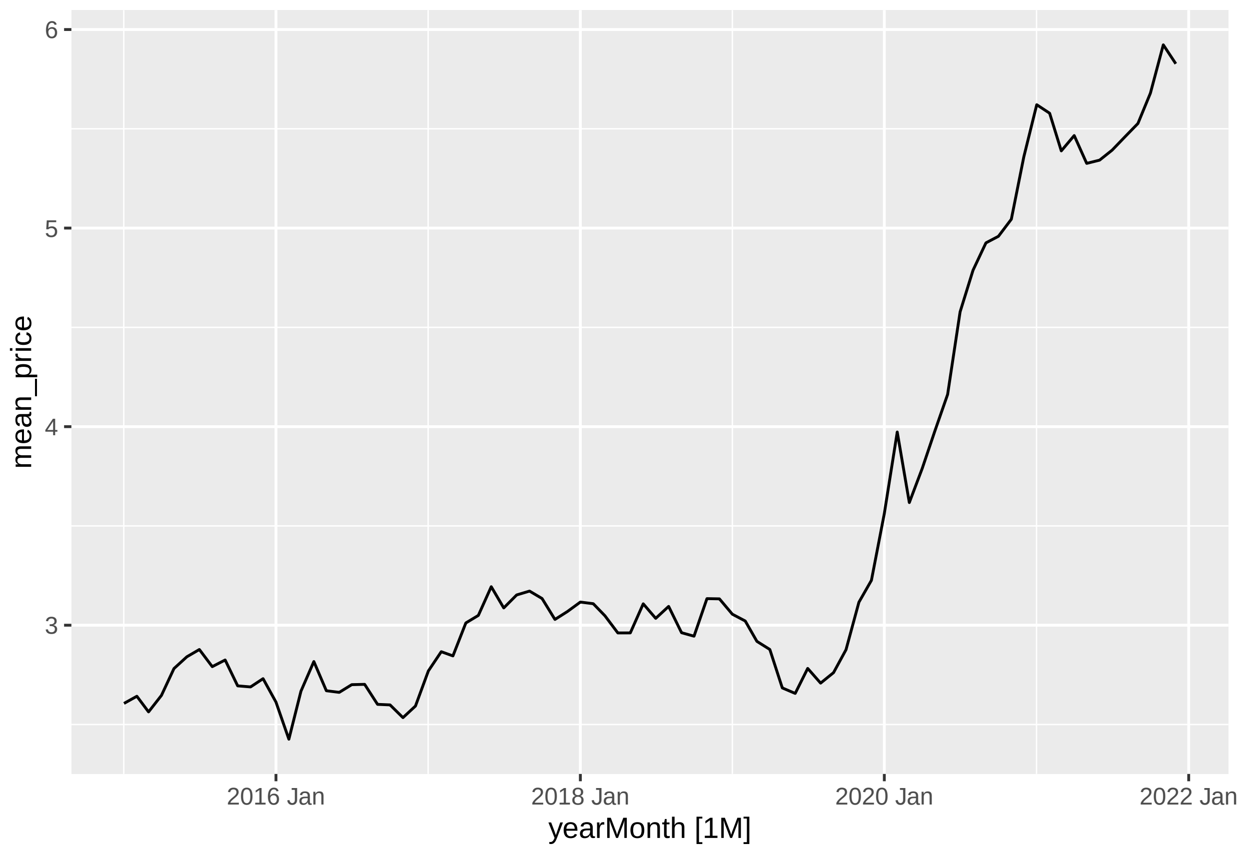
```
## # A dable: 106 x 7 [1M]
## # Key:     .model [1]
## # :        mean_price = trend + season_year + remainder
##    .model yearMonth mean_price trend season_year remainder season_adjust
##    <chr>      <mth>      <dbl> <dbl>       <dbl>     <dbl>         <dbl>
##  1 stl     2015 Jan       2.61  2.68    0.0226     -0.101           2.58
##  2 stl     2015 Feb       2.64  2.69    0.0220     -0.0690          2.62
##  3 stl     2015 Mar       2.56  2.69   -0.0441     -0.0864          2.61
##  4 stl     2015 Apr       2.65  2.70    0.00820    -0.0601          2.64
##  5 stl     2015 May       2.78  2.70   -0.0348      0.116           2.82
##  6 stl     2015 Jun       2.84  2.70    0.0121      0.125           2.83
##  7 stl     2015 Jul       2.88  2.71    0.0345      0.137           2.84
##  8 stl     2015 Aug       2.79  2.71    0.0316      0.0513          2.76
##  9 stl     2015 Sep       2.82  2.71   -0.000405    0.114           2.83
## 10 stl     2015 Oct       2.69  2.71   -0.0392      0.0200          2.73
## # i 96 more rows
```

```r
components(dcmp) |> autoplot()
```




```r
monthly_price_avg_tsib_train <- subset(monthly_price_avg_tsib_transformed, yearMonth < yearmonth(as.Date
monthly_price_avg_tsib_test <- subset(monthly_price_avg_tsib_transformed, yearMonth >= yearmonth(as.Date
```

```r
monthly_price_avg_tsib_train %>% autoplot(mean_price)
```

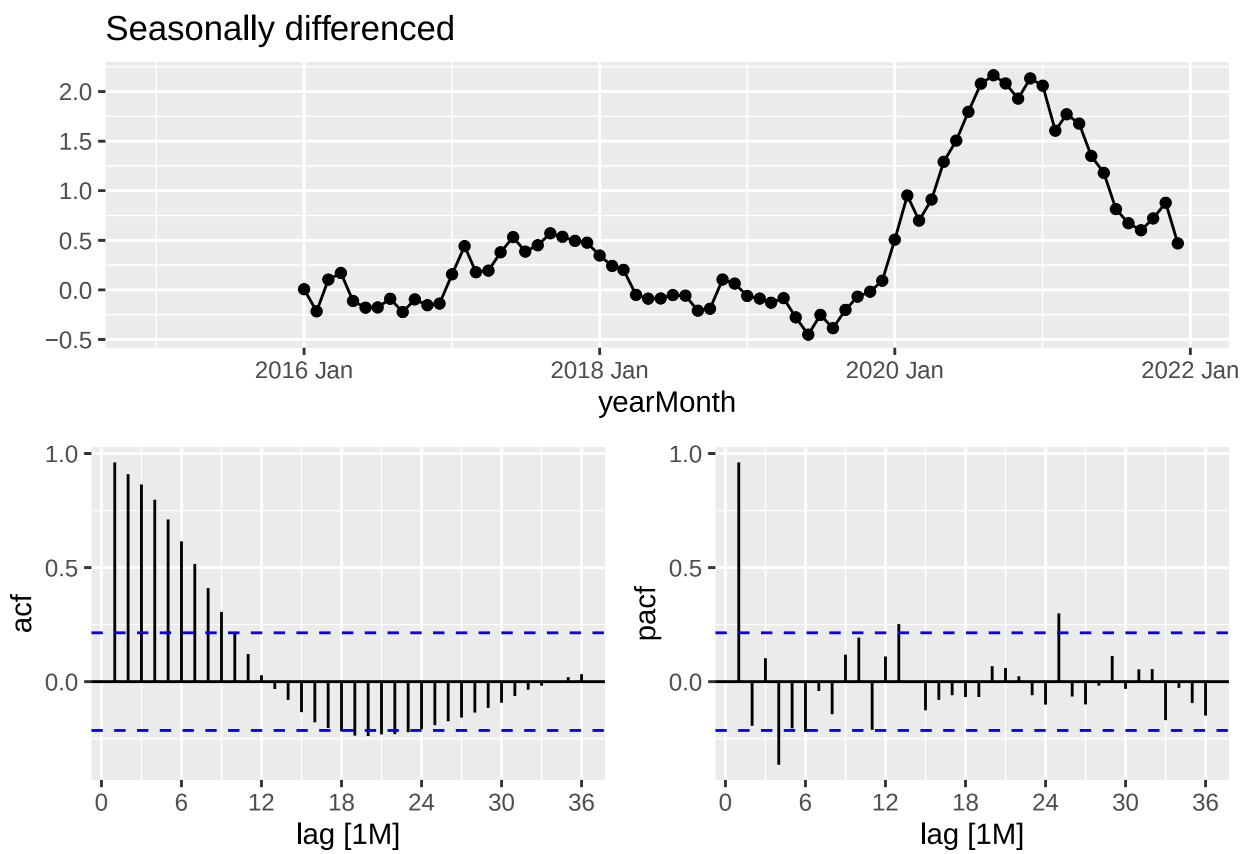


Due to the observed non-stationarity and seasonality with period 12, we take a seasonal difference.

```
monthly_price_avg_tsib_train |>
  gg_tsdisplay(difference(mean_price, 12),
               plot_type='partial', lag=36) +
  labs(title="Seasonally differenced", y="")
```

```
## Warning: Removed 12 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 12 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

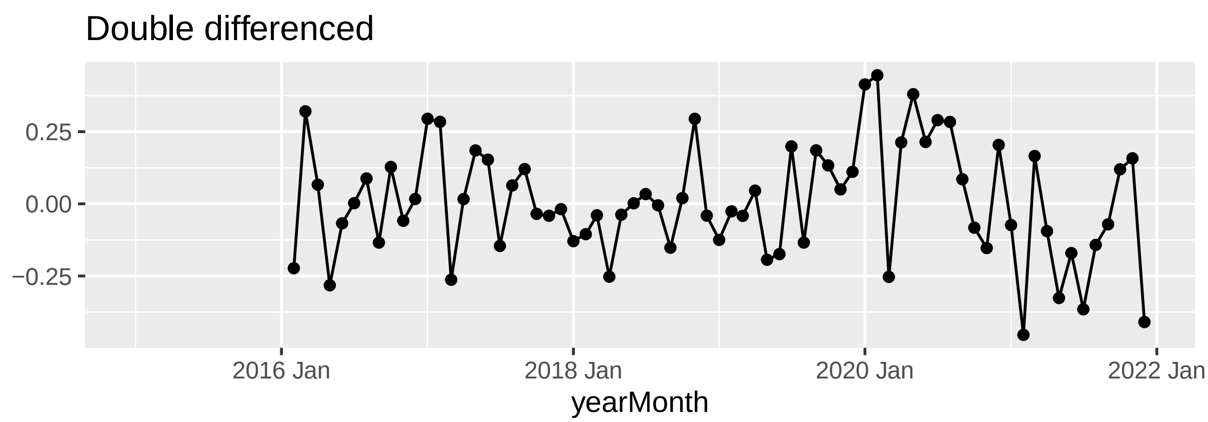


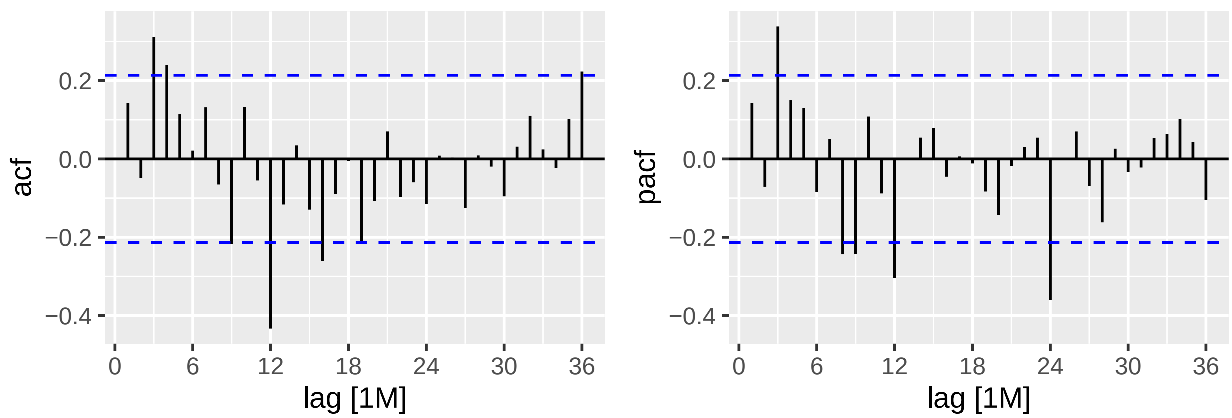
The series is still non-stationary, so we take a further first difference.

```
monthly_price_avg_tsib_train |>
  gg_tsdisplay(difference(mean_price, 12) |> difference(),
               plot_type='partial', lag=36) +
  labs(title = "Double differenced", y="")
```

```
## Warning: Removed 13 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 13 rows containing missing values or values outside the scale range
## (`geom_point()`).
```





We can see now that the data are closer to stationary, despite a one or two significant lags.

Models to use: NAIVE, AR, SARIMA

We'll start off with a simple forecasting method - the NAIVE method. Using this method, we set the forecasts to be the value of the last observation, a method that works surprisingly well in economics and finance.

Then, we compare them against a pure AR and a SARIMA model chosen by best AIC.

```
install.packages('forecast')
```

```
## Installing package into '/srv/r'
## (as 'lib' is unspecified)
```

```
library(forecast)
library(fable)

model_comp <- monthly_price_avg_tsib_train %>%
  model(model_1 = ARIMA(mean_price ~ 0 + pdq(3, 0, 0) + PDQ(0, 0, 0)),
        model_2 = ARIMA(mean_price ~ 0 + pdq(0,1,0) + PDQ(1, 1, 0)),
        auto_aic_mod = ARIMA(mean_price ~ 0 + pdq(1:10, 1:2, 1:10) +
                                                    PDQ(1:2,0:1,0), ic="aic",
                                                    stepwise=F, greedy=F),
        auto_bic_mod = ARIMA(mean_price ~ pdq(0:10, 1:2, 0:10) +
                                                    PDQ(0:2,0:1,0), ic="bic",
                                                    stepwise=F, greedy=F),
        random_walk_mod = NAIVE(mean_price)
        )
```

```
## Warning: 1 error encountered for model_1
## [1] non-stationary AR part from CSS
```

```
model_comp
```

```
## # A mable: 1 x 5
##        model_1                  model_2              auto_aic_mod
##        <model>                  <model>                   <model>
## 1 <NULL model> <ARIMA(0,1,0)(1,1,0)[12]> <ARIMA(1,1,3)(1,0,0)[12]>
## # i 2 more variables: auto_bic_mod <model>, random_walk_mod <model>
```

```
model_comp %>%
  augment() %>%
  filter(.model == "model_1") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = NA, df = 10, p-value = NA
```

```
model_comp %>%
  augment() %>%
  filter(.model == "model_2") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 15.511, df = 10, p-value = 0.1145
```

```
model_comp %>%
  augment() %>%
  filter(.model == "auto_aic_mod") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 4.5076, df = 10, p-value = 0.9216
```

```
model_comp %>%
  augment() %>%
  filter(.model == "auto_bic_mod") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 11.649, df = 10, p-value = 0.3092
```

```
model_comp %>%
  augment() %>%
  filter(.model == "random_walk_mod") %>%
  select(.resid) %>%
  as.ts() %>%
  Box.test(., lag=10, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  .
## X-squared = 11.52, df = 10, p-value = 0.3185
```

```
new_tesla_stock <- new_data(monthly_price_avg_tsib_train, 4)
new_tesla_stock
```

```
## # A tsibble: 4 x 1 [1M]
##   yearMonth
##       <mth>
## 1  2022 Jan
## 2  2022 Feb
## 3  2022 Mar
## 4  2022 Apr
```

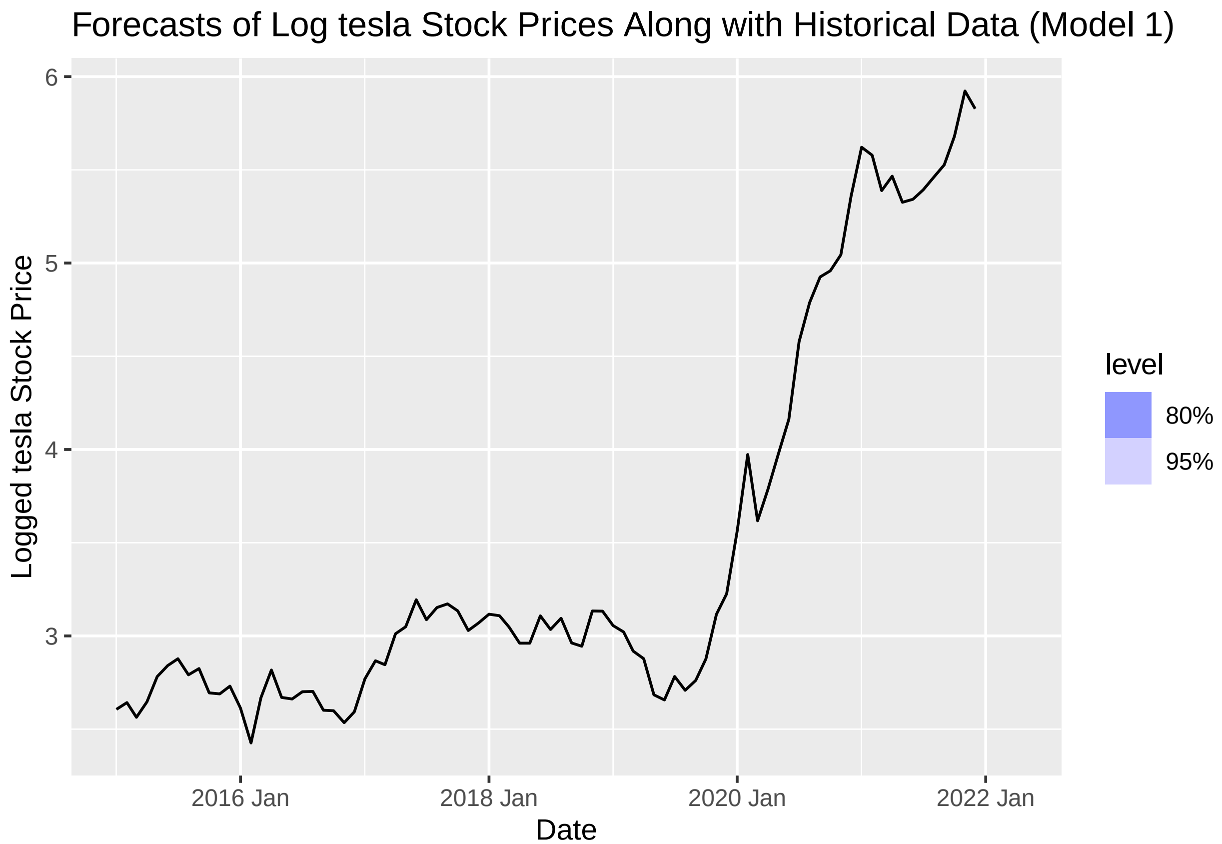
```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "model_1") %>% autoplot(monthly_price_avg_tsib_train) + labs(x = "Date", y = "Logged
      title = "Forecasts of Log tesla Stock Prices Along with Historical Data (Model 1)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

```
## Warning: Removed 4 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



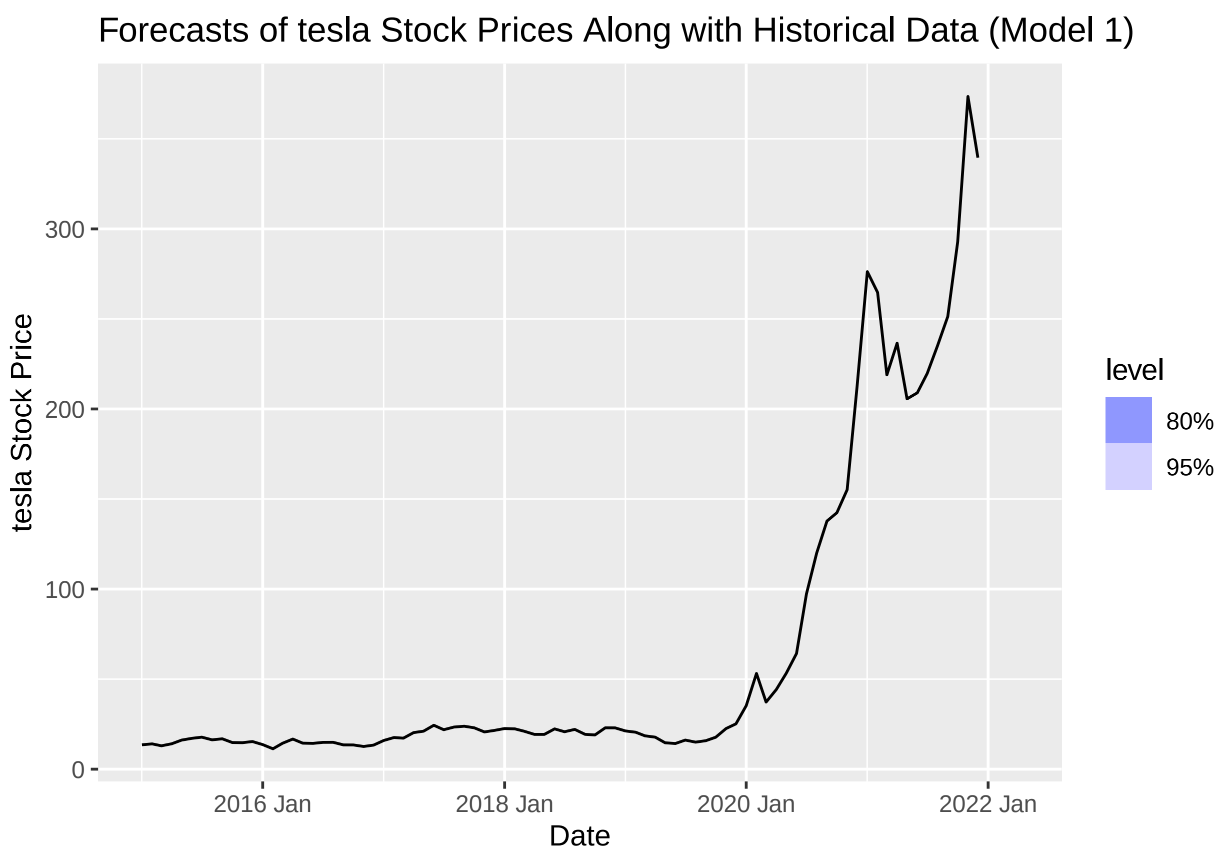

```
reference_data <- monthly_price_avg_tsib_train %>% mutate(mean_price = exp(mean_price))
reference_data
```

```
## # A tsibble: 84 x 2 [1M]
##    mean_price yearMonth
##         <dbl>     <mth>
##  1       13.5  2015 Jan
##  2       14.0  2015 Feb
##  3       13.0  2015 Mar
##  4       14.1  2015 Apr
##  5       16.1  2015 May
##  6       17.1  2015 Jun
##  7       17.8  2015 Jul
##  8       16.3  2015 Aug
##  9       16.9  2015 Sep
## 10       14.8  2015 Oct
## # i 74 more rows
```

We need to use fable object before autoplot when making a plots of forecasts along with historical data. When we transform log values back to their original values, we must also back-transform the probability distribution that is created in the fable object, as that probability distribution is centered around the log transformed forecast value.

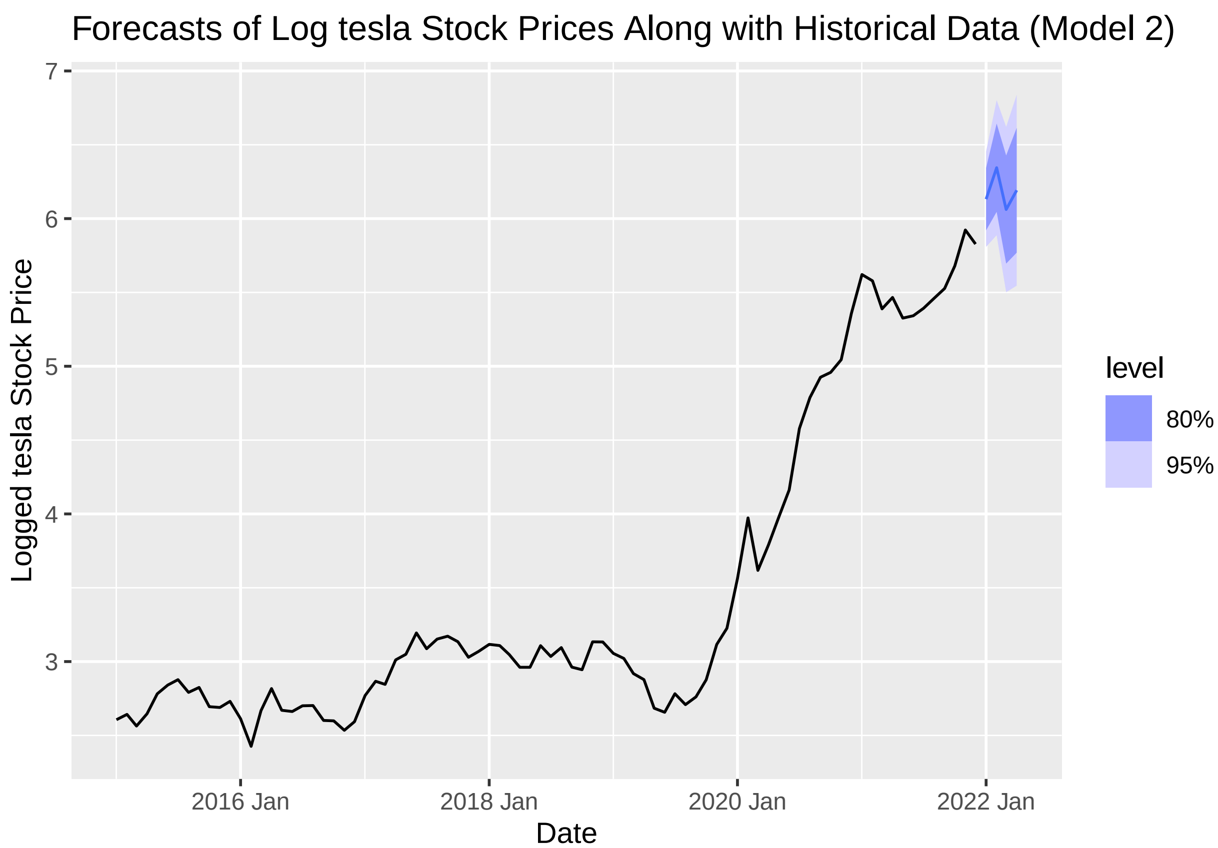
```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "model_1") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "tesla Stock Price",
       title = "Forecasts of tesla Stock Prices Along with Historical Data (Model 1)")
```

```
## Warning: Input forecast horizon ‘h‘ will be ignored as ‘new_data‘ has been
## provided.

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf

## Warning: Removed 4 rows containing missing values or values outside the scale range
## (‘geom_line()‘).
```



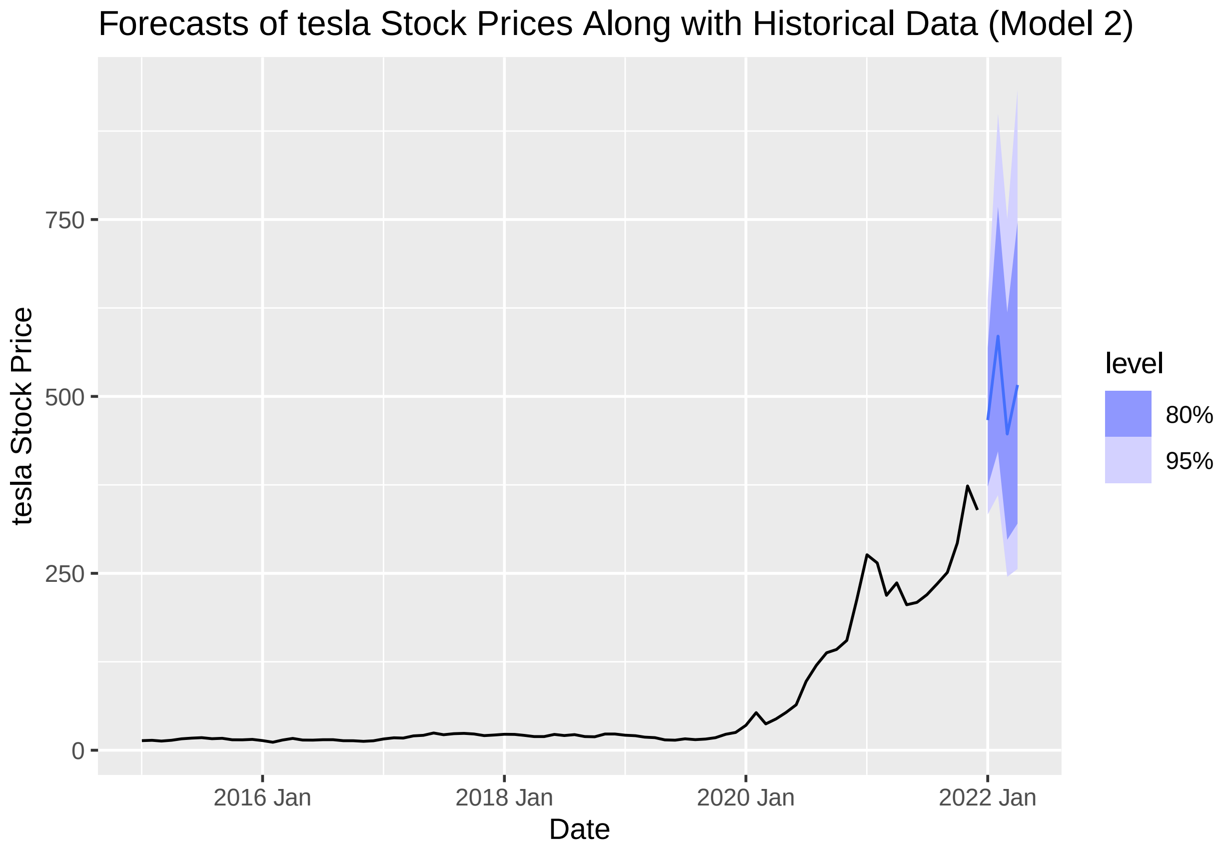

```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "model_2") %>% autoplot(monthly_price_avg_tsib_train) + labs(x = "Date", y = "Logged
       title = "Forecasts of Log tesla Stock Prices Along with Historical Data (Model 2)")
```

```
## Warning: Input forecast horizon ‘h‘ will be ignored as ‘new_data‘ has been
## provided.
```



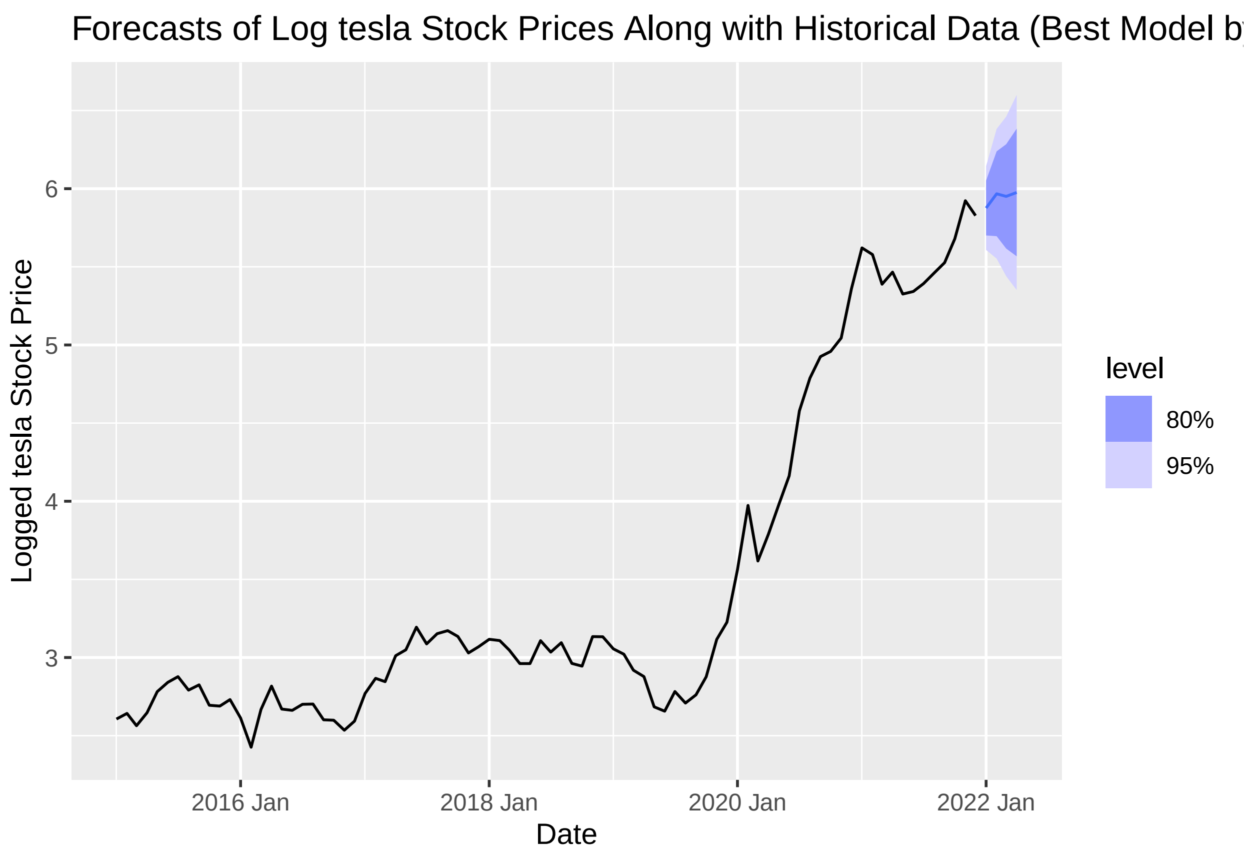

```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "model_2") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "tesla Stock Price",
       title = "Forecasts of tesla Stock Prices Along with Historical Data (Model 2)")
```

```
## Warning: Input forecast horizon ‘h‘ will be ignored as ‘new_data‘ has been
## provided.
```



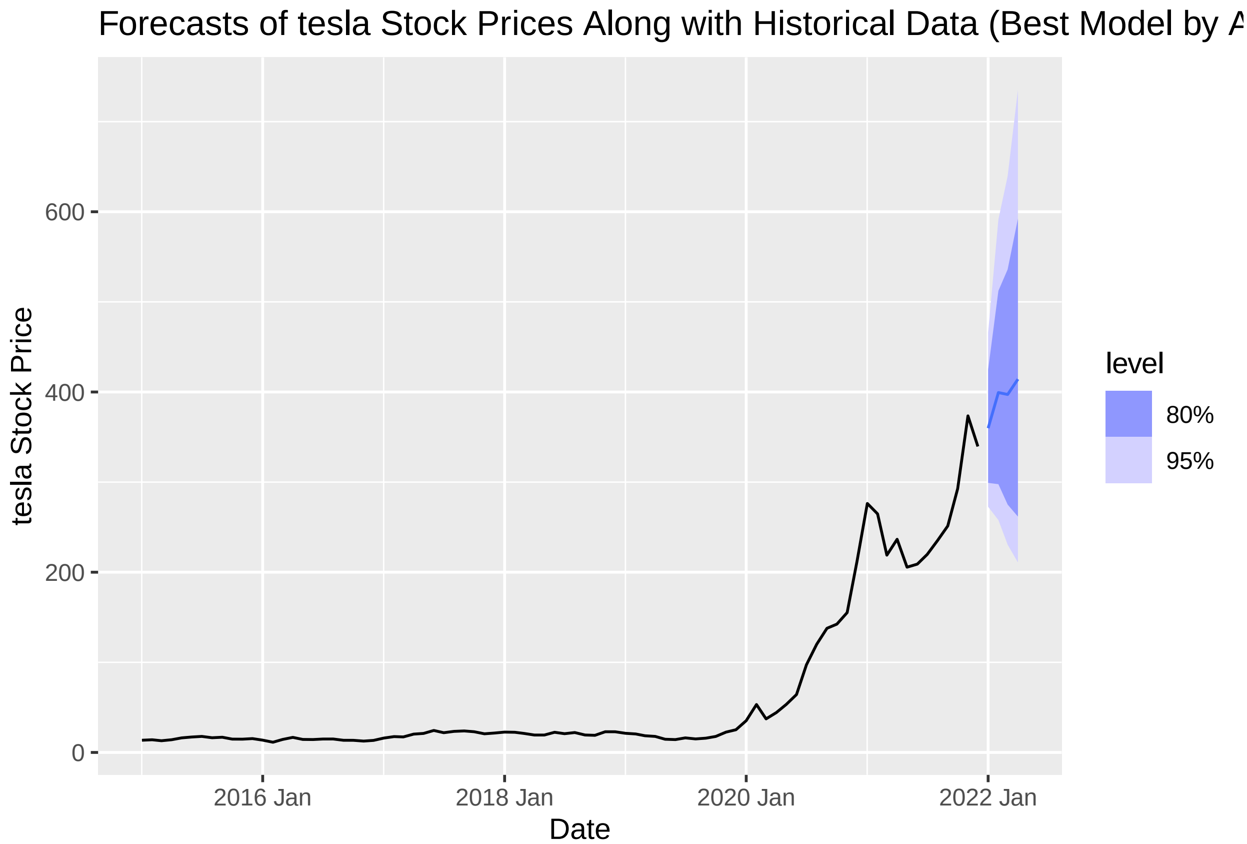

```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "auto_aic_mod") %>% autoplot(monthly_price_avg_tsib_train) + labs(x = "Date", y = "Lo
       title = "Forecasts of Log tesla Stock Prices Along with Historical Data (Best Model by AIC)")
```

```
## Warning: Input forecast horizon 'h' will be ignored as 'new_data' has been
## provided.
```



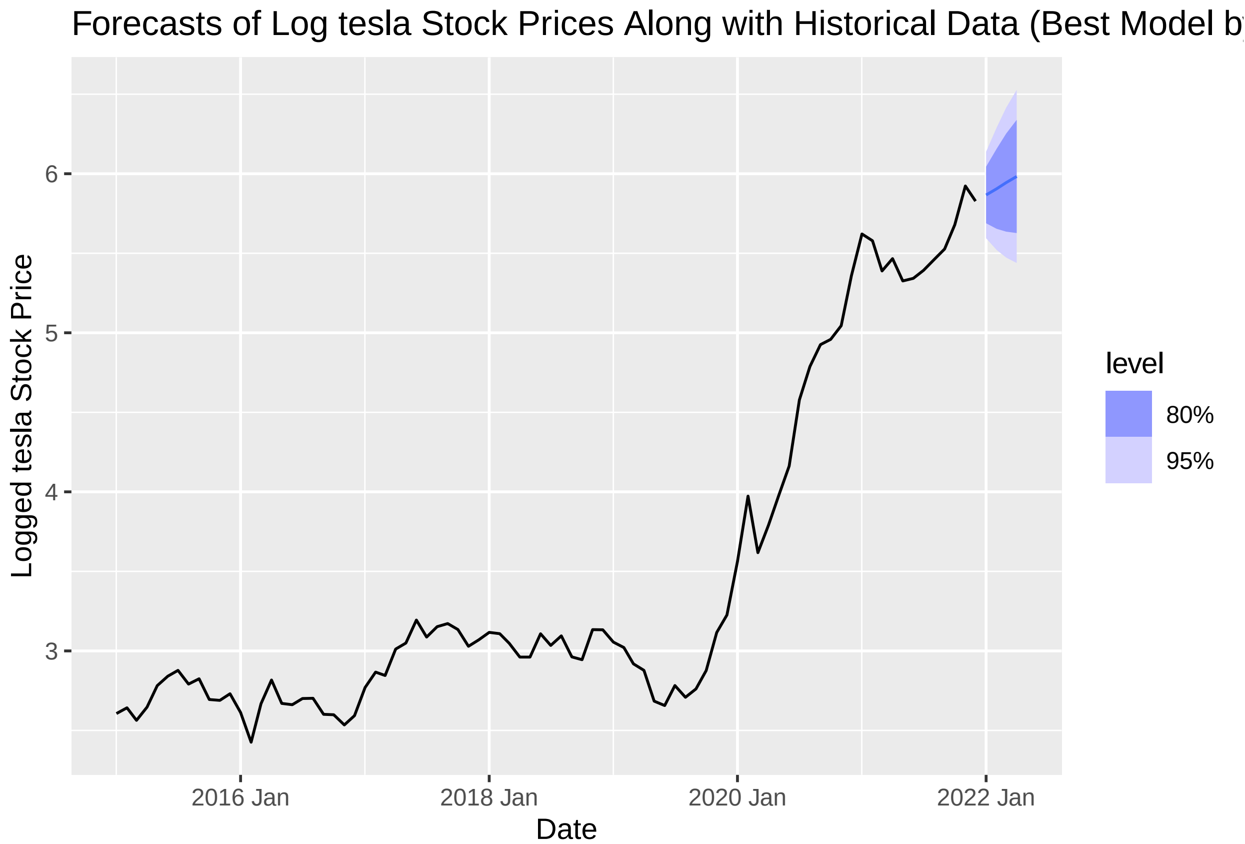

```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "auto_aic_mod") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "tesla Stock Price",
       title = "Forecasts of tesla Stock Prices Along with Historical Data (Best Model by AIC)")
```

```
## Warning: Input forecast horizon ‘h‘ will be ignored as ‘new_data‘ has been
## provided.
```



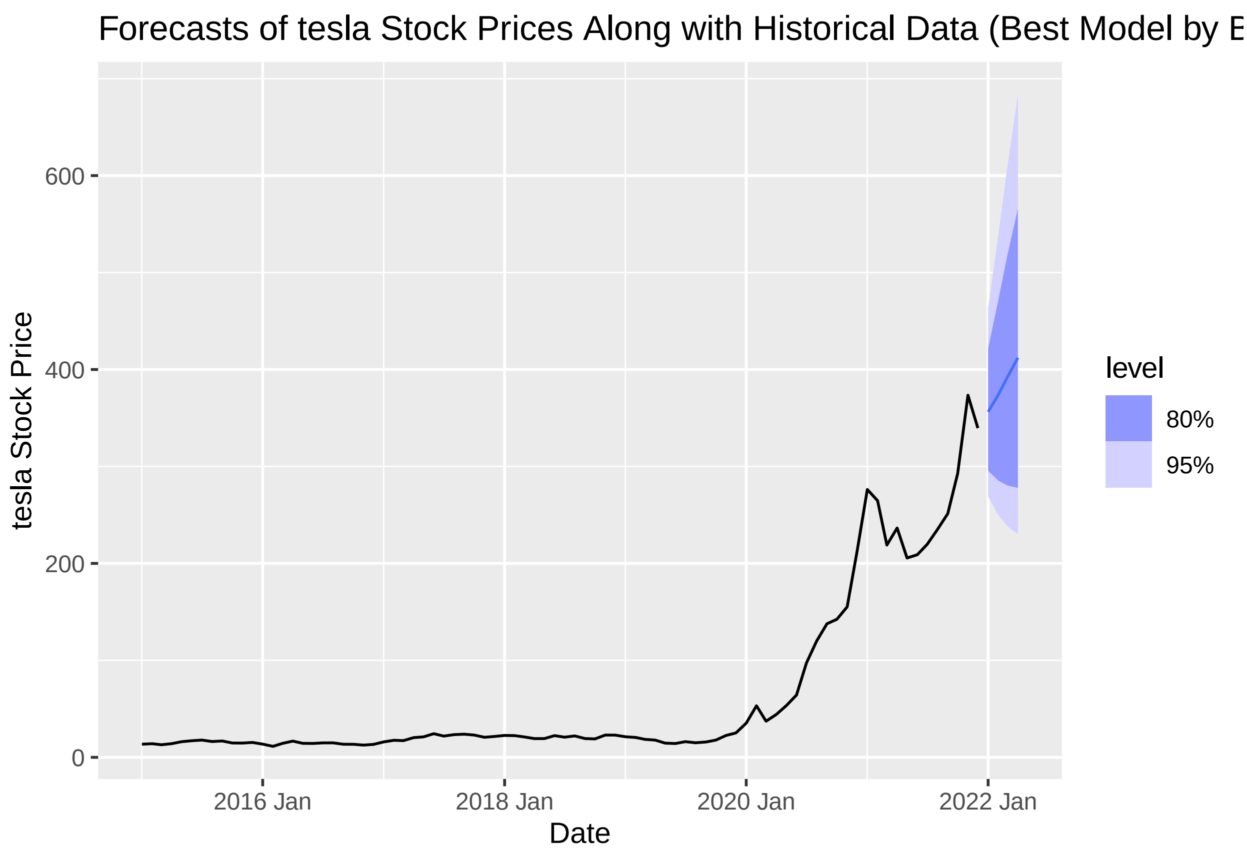

```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "auto_bic_mod") %>% autoplot(monthly_price_avg_tsib_train) + labs(x = "Date", y = "Lo
      title = "Forecasts of Log tesla Stock Prices Along with Historical Data (Best Model by BIC)")
```

```
## Warning: Input forecast horizon `h` will be ignored as `new_data` has been
## provided.
```



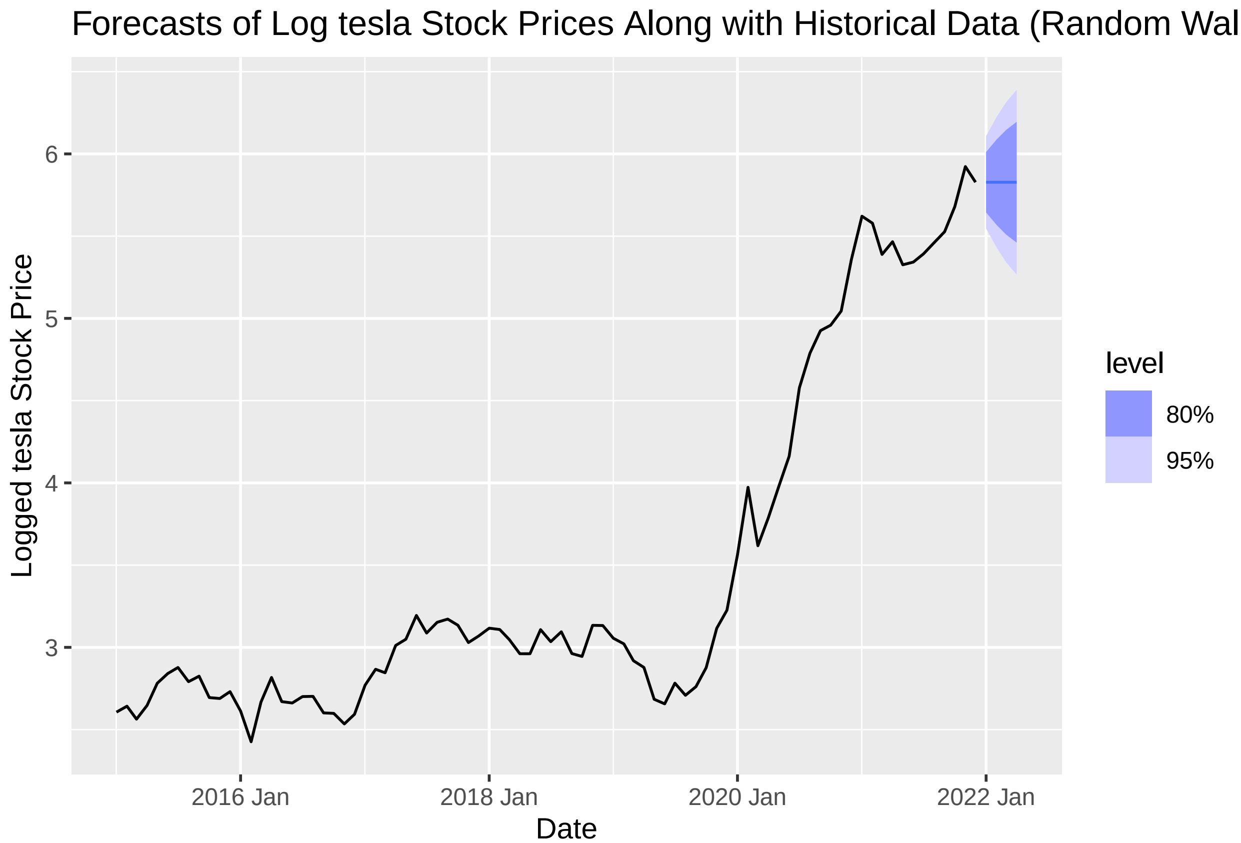

```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "auto_bic_mod") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "tesla Stock Price",
       title = "Forecasts of tesla Stock Prices Along with Historical Data (Best Model by BIC)")
```

```
## Warning: Input forecast horizon ‘h‘ will be ignored as ‘new_data‘ has been
## provided.
```



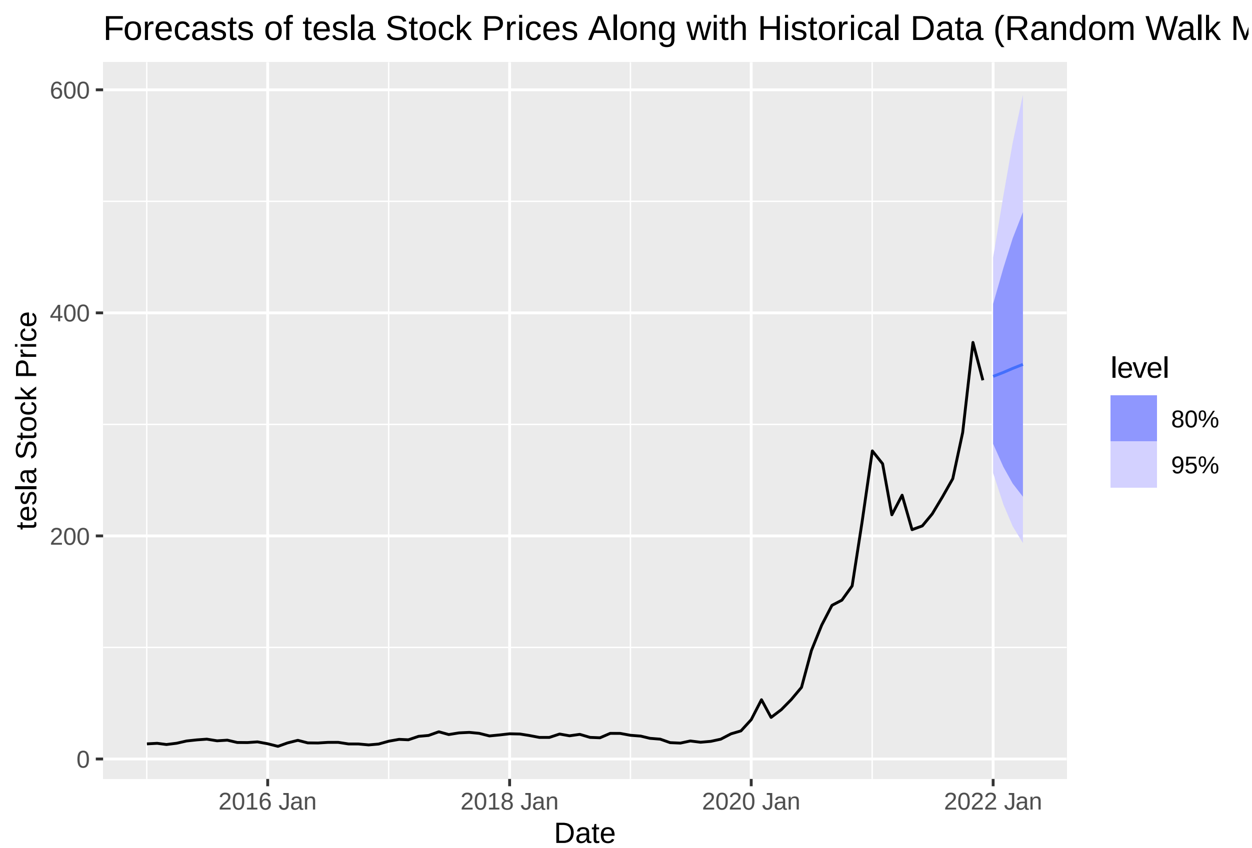

```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "random_walk_mod") %>% autoplot(monthly_price_avg_tsib_train) + labs(x = "Date", y =
      title = "Forecasts of Log tesla Stock Prices Along with Historical Data (Random Walk Model)")
```

```
## Warning: Input forecast horizon ‘h‘ will be ignored as ‘new_data‘ has been
## provided.
```




```
model_comp %>% forecast(new_tesla_stock, h=4) %>%
  filter(.model == "random_walk_mod") %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price)) %>%
  autoplot(reference_data) + labs(x = "Date", y = "tesla Stock Price",
       title = "Forecasts of tesla Stock Prices Along with Historical Data (Random Walk Model)")
```

```
## Warning: Input forecast horizon ‘h‘ will be ignored as ‘new_data‘ has been
## provided.
```




```r
model_forecasts <- model_comp %>% forecast(new_tesla_stock, h=4) %>%
  mutate(.mean = exp(.mean), mean_price = exp(mean_price))
```

```
## Warning: Input forecast horizon ‘h‘ will be ignored as ‘new_data‘ has been
## provided.
```

```r
model_forecasts
```

```
## # A fable: 20 x 4 [1M]
## # Key:     .model [5]
##    .model          yearMonth     mean_price .mean
##    <chr>               <mth>         <dist> <dbl>
##  1 model_1          2022 Jan          t(NA)   NA
##  2 model_1          2022 Feb          t(NA)   NA
##  3 model_1          2022 Mar          t(NA)   NA
##  4 model_1          2022 Apr          t(NA)   NA
##  5 model_2          2022 Jan lN(6.1, 0.027)  460.
##  6 model_2          2022 Feb lN(6.3, 0.054)  569.
##  7 model_2          2022 Mar lN(6.1, 0.082)  429.
##  8 model_2          2022 Apr  lN(6.2, 0.11)  489.
##  9 auto_aic_mod     2022 Jan lN(5.9, 0.019)  356.
## 10 auto_aic_mod     2022 Feb   lN(6, 0.045)  391.
## 11 auto_aic_mod     2022 Mar   lN(6, 0.068)  384.
## 12 auto_aic_mod     2022 Apr     lN(6, 0.1)  394.
## 13 auto_bic_mod     2022 Jan lN(5.9, 0.019)  353.
```

```
## 14 auto_bic_mod     2022 Feb lN(5.9, 0.039)  367.
## 15 auto_bic_mod     2022 Mar lN(5.9, 0.058)  381.
## 16 auto_bic_mod     2022 Apr   lN(6, 0.077)  397.
## 17 random_walk_mod  2022 Jan lN(5.8, 0.021)  340.
## 18 random_walk_mod  2022 Feb lN(5.8, 0.041)  340.
## 19 random_walk_mod  2022 Mar lN(5.8, 0.062)  340.
## 20 random_walk_mod  2022 Apr lN(5.8, 0.082)  340.
```

```
comparison_data <- monthly_price_avg_tsib %>% select(mean_price)
comparison_data
```

```
## # A tsibble: 106 x 2 [1M]
##    mean_price yearMonth
##         <dbl>     <mth>
##  1       13.5  2015 Jan
##  2       14.0  2015 Feb
##  3       13.0  2015 Mar
##  4       14.1  2015 Apr
##  5       16.1  2015 May
##  6       17.1  2015 Jun
##  7       17.8  2015 Jul
##  8       16.3  2015 Aug
##  9       16.9  2015 Sep
## 10       14.8  2015 Oct
## # i 96 more rows
```

```
accuracy(model_forecasts, comparison_data)
```

```
## # A tibble: 5 x 10
##   .model          .type     ME  RMSE   MAE   MPE  MAPE    MASE   RMSSE    ACF1
##   <chr>           <chr>  <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl>   <dbl>
## 1 auto_aic_mod    Test   -75.9  82.3  75.9 -24.5  24.5   1.77    1.04  -0.251
## 2 auto_bic_mod    Test   -67.1  72.5  67.1 -21.6  21.6   1.57    0.915 -0.0374
## 3 model_1         Test   NaN   NaN   NaN   NaN   NaN   NaN     NaN     NA
## 4 model_2         Test  -187.  198.  187.  -60.1  60.1   4.36    2.49  -0.647
## 5 random_walk_mod Test   -31.7  36.9  31.7 -10.4  10.4   0.739   0.465 -0.280
```

```
future_compare <- monthly_price_avg_tsib_test %>% mutate(mean_price = exp(mean_price))
future_compare$is_up <- append(as.numeric(diff(future_compare$mean_price) > 0), 0)
future_compare <- future_compare %>% slice(1:4)
```

```
bic <- model_forecasts %>% filter(.model == "auto_bic_mod") %>% as_tsibble()
bic$is_up <- as.numeric(append(diff(bic$.mean) > 0, 0))
```

```
sum(bic$is_up == future_compare$is_up) / length(bic$is_up)
```

```
## [1] 0.75
```

We get a preliminary accuracy of 0.75 when forecasting 4 months ahead.