6510_final

Zijun Ma_T00711782, Shaomeng Yin – T00708655

2023-11-30

```r
# load the packages
library(fpp3)
```

```
## -- Attaching packages -------------------------------------------- fpp3 0.5 --

## v tibble      3.2.1     v tsibble     1.1.3
## v dplyr       1.1.3     v tsibbledata 0.4.1
## v tidyr       1.3.0     v feasts      0.3.1
## v lubridate   1.9.2     v fable       0.3.3
## v ggplot2     3.4.3     v fabletools  0.3.3

## -- Conflicts ------------------------------------------------- fpp3_conflicts --
## x lubridate::date()    masks base::date()
## x dplyr::filter()      masks stats::filter()
## x tsibble::intersect() masks base::intersect()
## x tsibble::interval()  masks lubridate::interval()
## x dplyr::lag()         masks stats::lag()
## x tsibble::setdiff()   masks base::setdiff()
## x tsibble::union()     masks base::union()
```

```r
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following object is masked from 'package:tsibble':
##
##     index

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
library(quantmod) # download data form Yahoo finance
```

```
## Loading required package: xts
```

```
##
## ######################### Warning from 'xts' package ##########################
## #                                                                             #
## # The dplyr lag() function breaks how base R's lag() function is supposed to  #
## # work, which breaks lag(my_xts). Calls to lag(my_xts) that you type or       #
## # source() into this session won't work correctly.                            #
## #                                                                             #
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can add #
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop           #
## # dplyr from breaking base R's lag() function.                                #
## #                                                                             #
## # Code in packages is not affected. It's protected by R's namespace mechanism #
## # Set `options(xts.warn_dplyr_breaks_lag = FALSE)` to suppress this warning.  #
## #                                                                             #
## ###############################################################################
```

```
##
## Attaching package: 'xts'
```

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

```
## Loading required package: TTR
```

```r
library(moments) # to know summary statistics of data
library(knitr)
library(forecast)
```

```
##
## Attaching package: 'forecast'
```

```
## The following object is masked from 'package:fabletools':
##
##     accuracy
```

```r
library(fabletools)
library(fable.prophet)
```

```
## Loading required package: Rcpp
```

```r
library(prophet)
```

```
## Loading required package: rlang
```

```
##
## Attaching package: 'prophet'

## The following object is masked from 'package:fable.prophet':
##
##     prophet
```

```
library(ggplot2)
library(ggpubr)
```

```
##
## Attaching package: 'ggpubr'

## The following object is masked from 'package:forecast':
##
##     gghistogram
```

Import data as dataframe

```
df <- read.csv("/Users/zijunma/Desktop/6510final/DailyDelhiClimateFull.csv")
head(df)
```

```
##         date  meantemp humidity wind_speed meanpressure
## 1 2013-01-01 10.000000 84.50000   0.000000     1015.667
## 2 2013-01-02  7.400000 92.00000   2.980000     1017.800
## 3 2013-01-03  7.166667 87.00000   4.633333     1018.667
## 4 2013-01-04  8.666667 71.33333   1.233333     1017.167
## 5 2013-01-05  6.000000 86.83333   3.700000     1016.500
## 6 2013-01-06  7.000000 82.80000   1.480000     1018.000
```

EDA

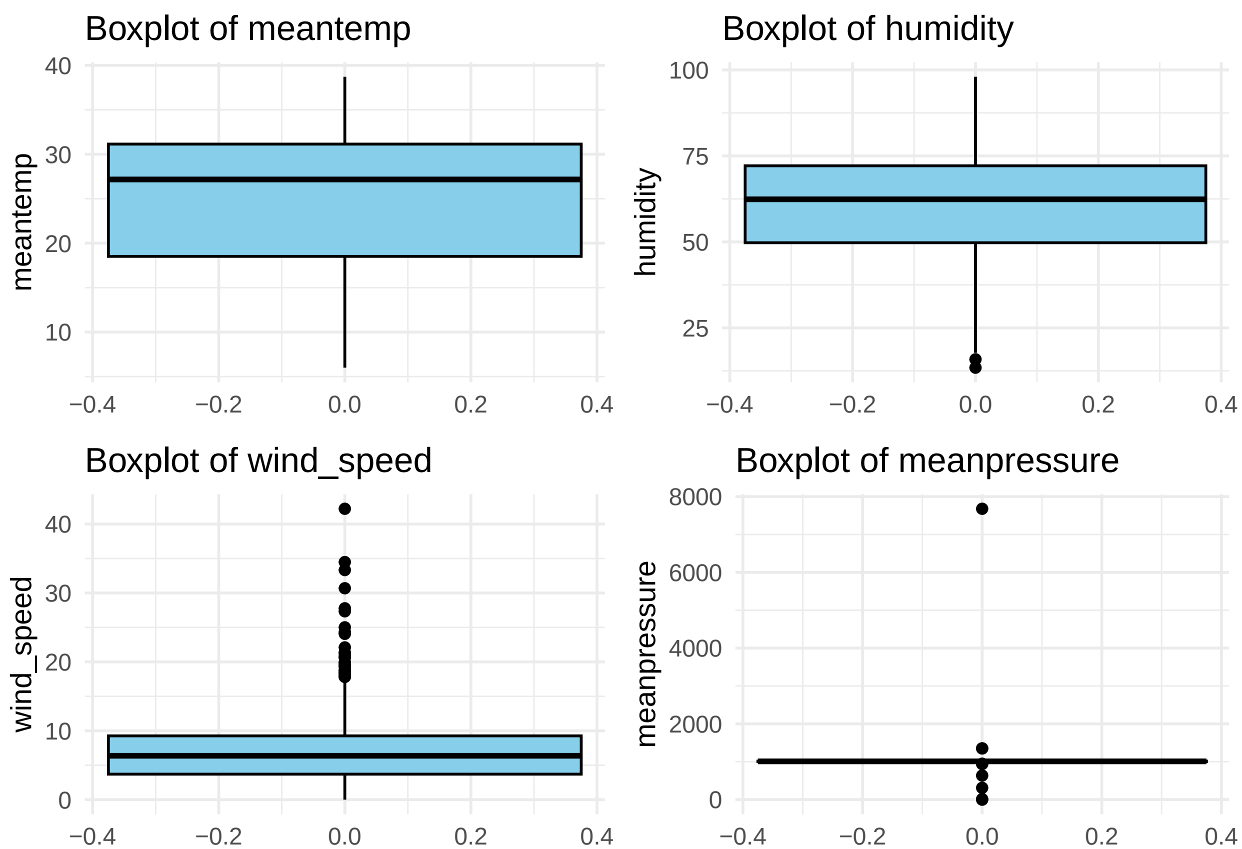
Scatter plot and boxplot

```
# box plot
box_plot_meantemp <- ggplot(df, aes_string(y = "meantemp")) +
  geom_boxplot(fill = "skyblue", color = "black") +
  labs(title = paste("Boxplot of meantemp")) +
  theme_minimal()
```

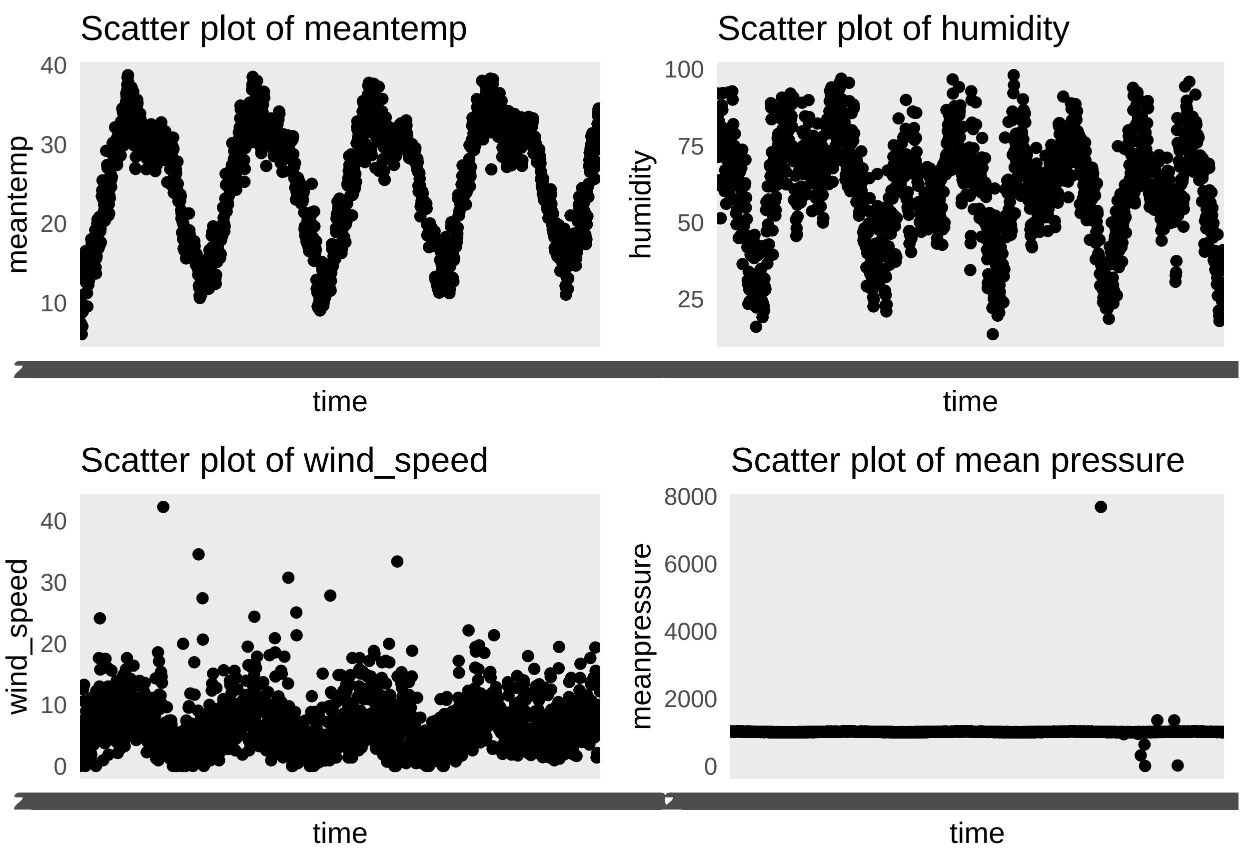
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
box_plot_humidity <- ggplot(df, aes_string(y = "humidity")) +
  geom_boxplot(fill = "skyblue", color = "black") +
  labs(title = paste("Boxplot of humidity")) +
  theme_minimal()

box_plot_wind_speed <- ggplot(df, aes_string(y = "wind_speed")) +
  geom_boxplot(fill = "skyblue", color = "black") +
  labs(title = paste("Boxplot of wind_speed")) +
  theme_minimal()

box_plot_meanpressure <- ggplot(df, aes_string(y = "meanpressure")) +
  geom_boxplot(fill = "skyblue", color = "black") +
  labs(title = paste("Boxplot of meanpressure")) +
  theme_minimal()

# scatter plot
scatter_plot_meantemp <- ggplot(df, aes(x = date, y = meantemp)) +
  geom_point() +
  labs(title = "Scatter plot of meantemp",
       x = "time",
       y = "meantemp") +
  theme_minimal()

# scatter plot
scatter_plot_humidity <- ggplot(df, aes(x = date, y = humidity)) +
  geom_point() +
  labs(title = "Scatter plot of humidity",
       x = "time",
       y = "humidity") +
  theme_minimal()

# scatter plot
scatter_plot_wind_speed <- ggplot(df, aes(x = date, y = wind_speed)) +
  geom_point() +
  labs(title = "Scatter plot of wind_speed",
       x = "time",
       y = "wind_speed") +
  theme_minimal()

# scatter plot
scatter_plot_meanpressure <- ggplot(df, aes(x = date, y = meanpressure)) +
  geom_point() +
  labs(title = "Scatter plot of mean pressure",
       x = "time",
       y = "meanpressure") +
  theme_minimal()
```

```r
figure_1 <- ggarrange(box_plot_meantemp, box_plot_humidity, box_plot_wind_speed, box_plot_meanpressure,
                      ncol = 2, nrow = 2)
figure_1
```




```
figure_2 <- ggarrange(scatter_plot_meantemp, scatter_plot_humidity, scatter_plot_wind_speed, scatter_plo
                    ncol = 2, nrow = 2)
figure_2
```



By checking the scatter plot and box plot, we can find there exist some outlier values for both wind_speed and mean pressure.

```r
# check missing data
sum(is.na(df))
```

```
## [1] 0
```

```r
tsoutliers(df$meantemp)
```

```
## $index
## integer(0)
##
## $replacements
## numeric(0)
```

```r
tsoutliers(df$humidity)
```

```
## $index
## integer(0)
##
## $replacements
## numeric(0)
```

```r
tsoutliers(df$wind_speed)
```

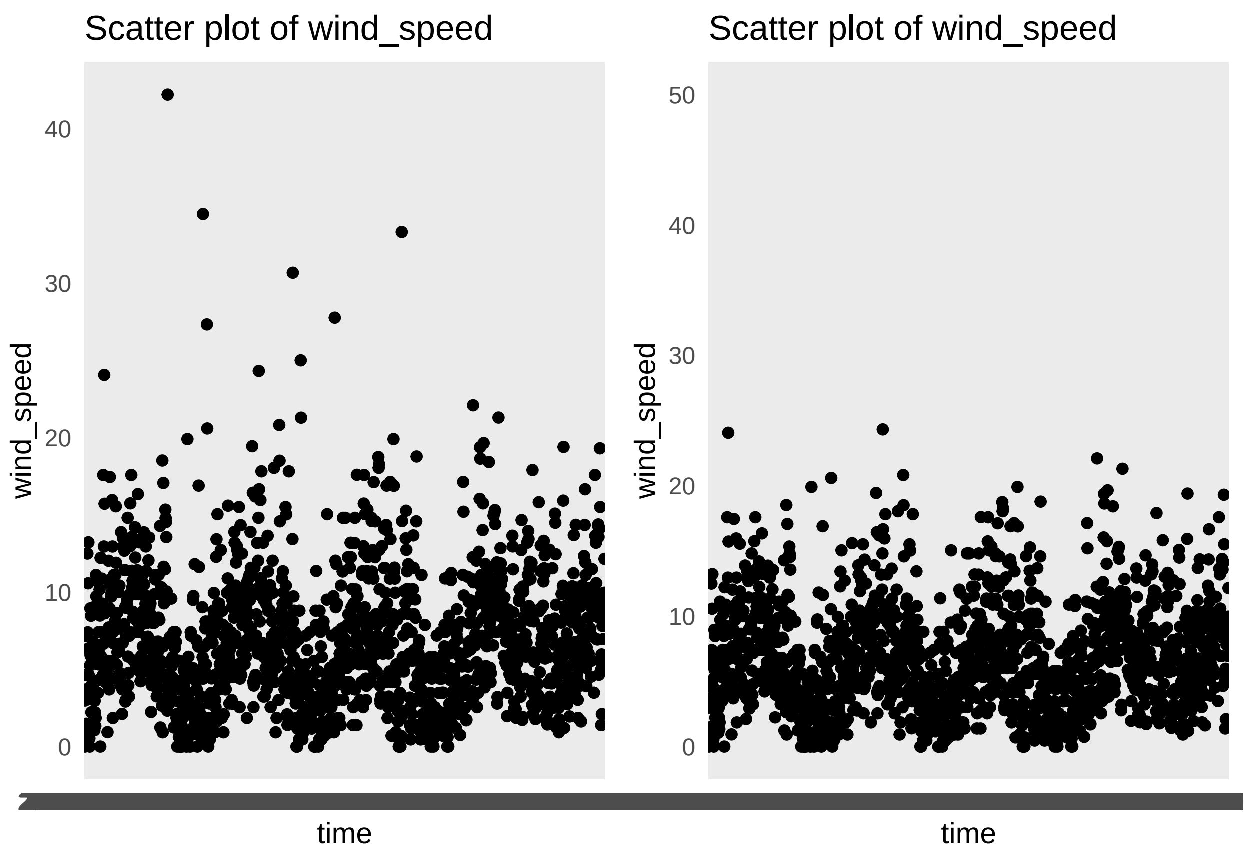
```
## $index
## [1] 252 359 371 631 655 656 758 961
##
## $replacements
## [1]  3.012500  5.093750 10.531250 10.768750  4.629167  4.395833  4.637500
## [8]  9.737500
```

```r
tsoutliers(df$meanpressure)
```

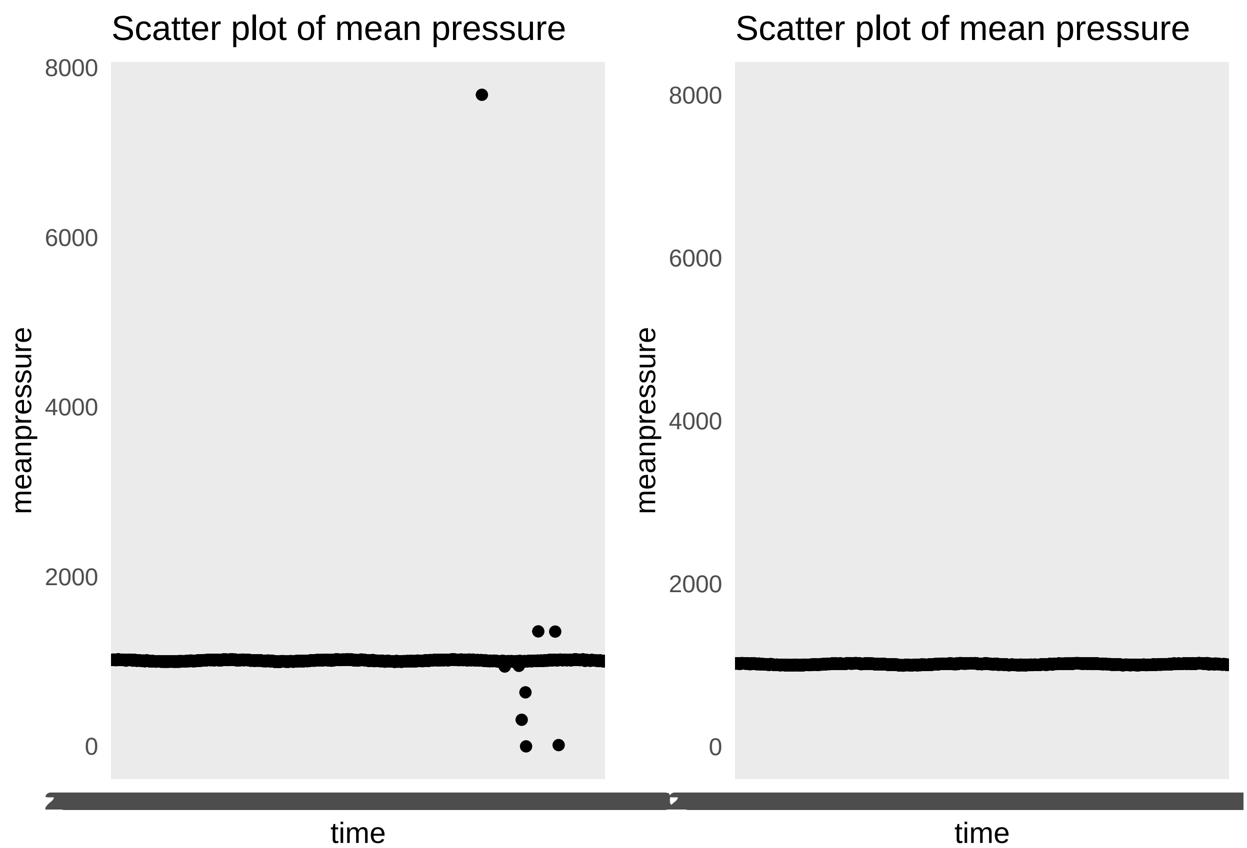
```
## $index
## [1] 1183 1256 1301 1310 1322 1324 1363 1417 1428
##
## $replacements
## [1] 1012.0625  998.7321 1001.7500  998.8125 1000.1786  999.4021 1005.0520
## [8] 1016.1154 1014.2955
```

```r
df_data_cleaned <- df
df_data_cleaned$wind_speed[tsoutliers(df$wind_speed)$index] <- tsoutliers(df$wind_speed)$replacements

df_data_cleaned$meanpressure[tsoutliers(df$meanpressure)$index] <- tsoutliers(df$meanpressure)$replacem
```

```r
# scatter plot for wind_speed without outliers
plot_wind_speed_without_outliers <- ggplot(df_data_cleaned, aes(x = date, y = wind_speed)) +
  geom_point() +
  labs(title = "Scatter plot of wind_speed",
       x = "time",
       y = "wind_speed") +
  ylim(0, 50) +
  theme_minimal()

# scatter plot for meanpressure without outliers
plot_meanpressure_without_outliers <- ggplot(df_data_cleaned, aes(x = date, y = meanpressure)) +
  geom_point() +
  labs(title = "Scatter plot of mean pressure",
       x = "time",
       y = "meanpressure") +
  ylim(0,8000) +
  theme_minimal()

figure_3 <- ggarrange(scatter_plot_wind_speed, plot_wind_speed_without_outliers,
                      ncol = 2, nrow = 1)
figure_4 <- ggarrange(scatter_plot_meanpressure, plot_meanpressure_without_outliers,
                      ncol = 2, nrow = 1)
figure_3
```


Scatter plot of wind_speed
wind_speed
40
30
20
10
0
time
Scatter plot of wind_speed
wind_speed
50
40
30
20
10
0
time

```
figure_4
```



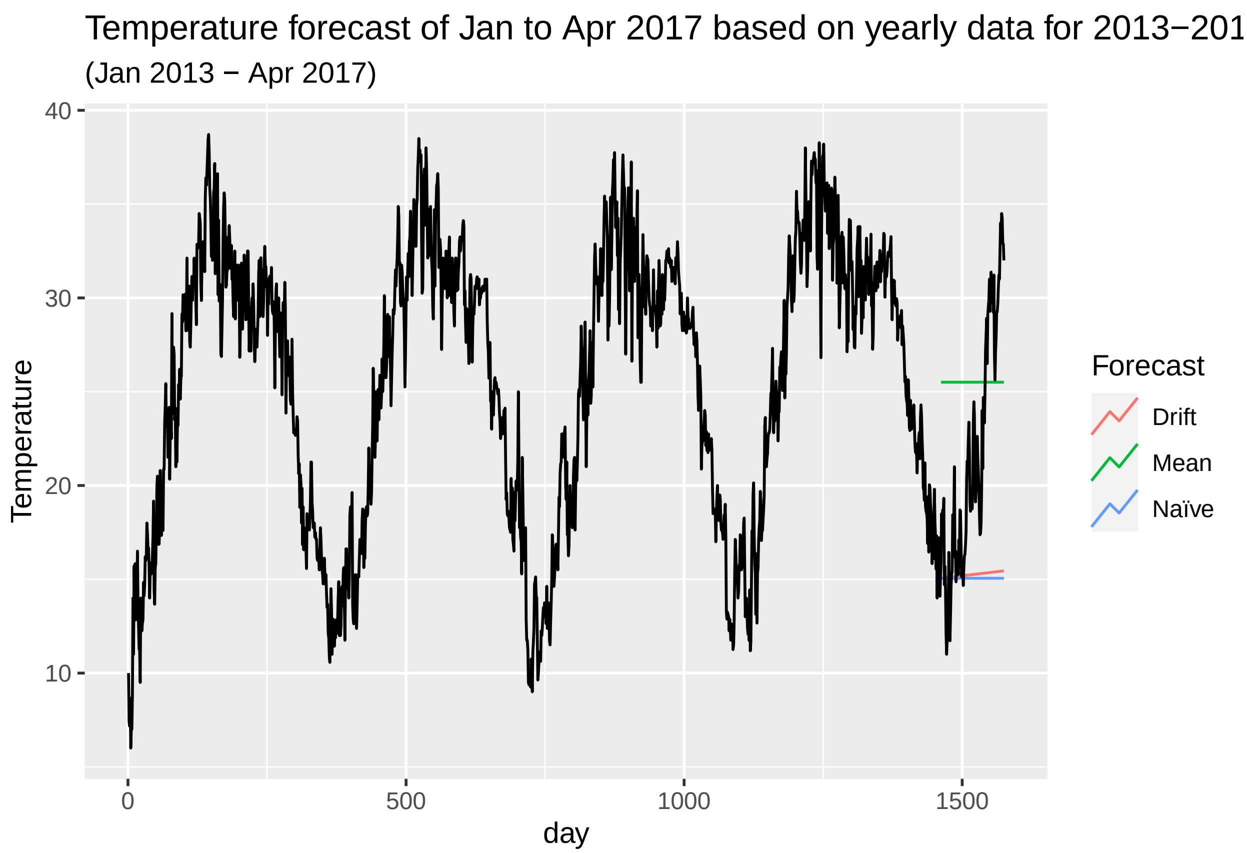

```
full_data_tsibble <- df_data_cleaned |>
  mutate(date = as.Date(date)) |>
  as_tsibble(index = date, regular = TRUE) |>
  mutate(day = row_number()) |>
  update_tsibble(index = day, regular = TRUE)
```

```
# split the dataset into training dataset and testing dataset
train_data <- full_data_tsibble |>
  filter(date >= as.Date("2013-01-01") & date <= as.Date("2016-12-31"))

test_data <- full_data_tsibble |>
  filter(date >= as.Date("2017-01-01") & date <= as.Date("2017-04-24"))
```

Three Benchmark models

```
# benchmark models (mean, naive, drift)
bench_fit <- train_data |>
  model(
    Mean = MEAN(meantemp),
    `Naïve` = NAIVE(meantemp),
    Drift = NAIVE(meantemp ~ drift())
  )
```

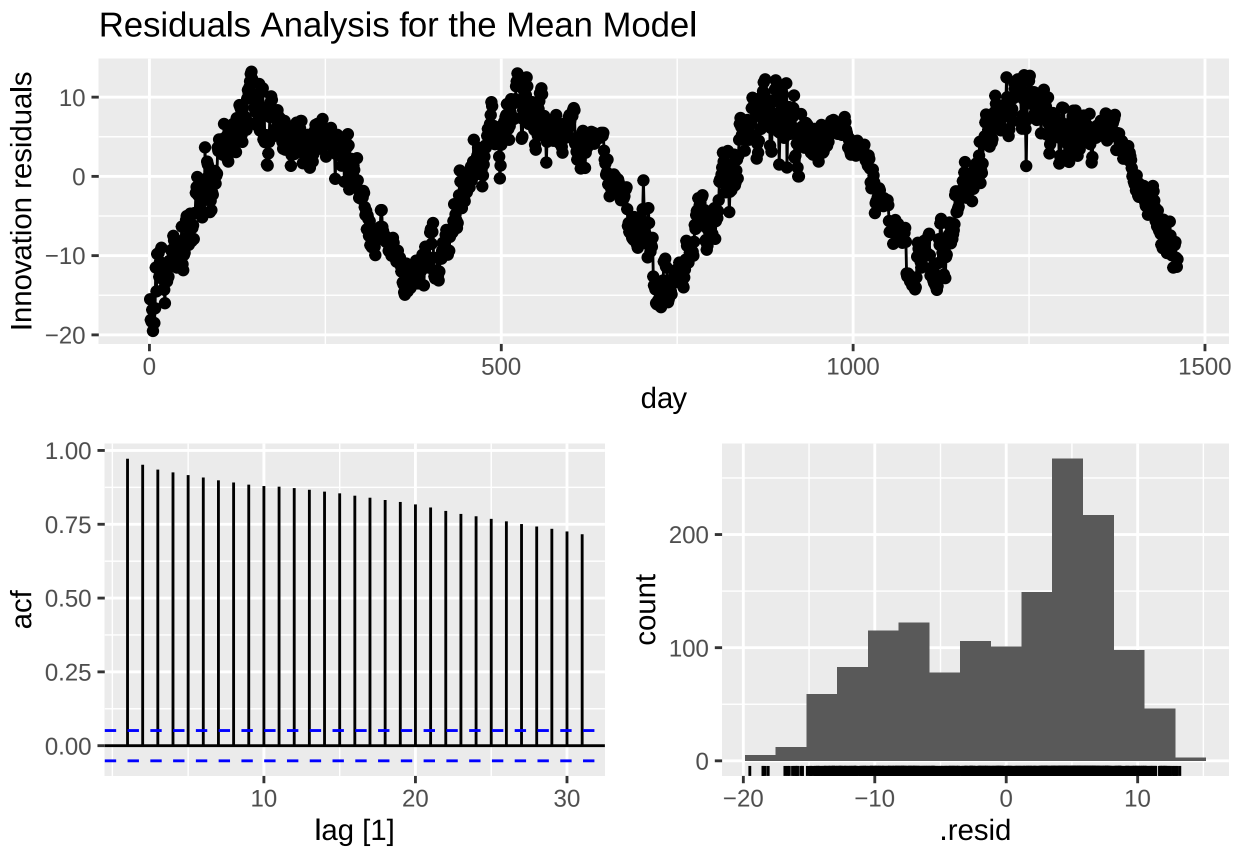
```r
# forecast
bench_fc <- bench_fit |>
  forecast(new_data = test_data)

# Plot the forecasts
bench_fc |>
  autoplot(full_data_tsibble, level = NULL) +
  labs(y = "Temperature",
       title = "Temperature forecast of Jan to Apr 2017 based on yearly data for 2013-2016",
       subtitle = "(Jan 2013 - Apr 2017)") +
  guides(colour = guide_legend(title = "Forecast"))
```




```r
# Forecast accuracy
bench_fc |>
  fabletools::accuracy(test_data)
```

```
## # A tibble: 3 x 10
##   .model .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Drift  Test   6.46  8.98  6.85  23.8  26.9   NaN   NaN 0.948
## 2 Mean   Test  -3.79  7.38  6.62 -27.7  36.8   NaN   NaN 0.949
## 3 Naïve  Test   6.66  9.19  7.04  24.6  27.7   NaN   NaN 0.949
```

```r
# RMSE Drift:8.976, Mean:7.381, Naive:9.190. Mean is the best.

# Check the residuals.
bench_fit |>
select(Mean) |>
gg_tsresiduals()+
  labs(title="Residuals Analysis for the Mean Model")
```



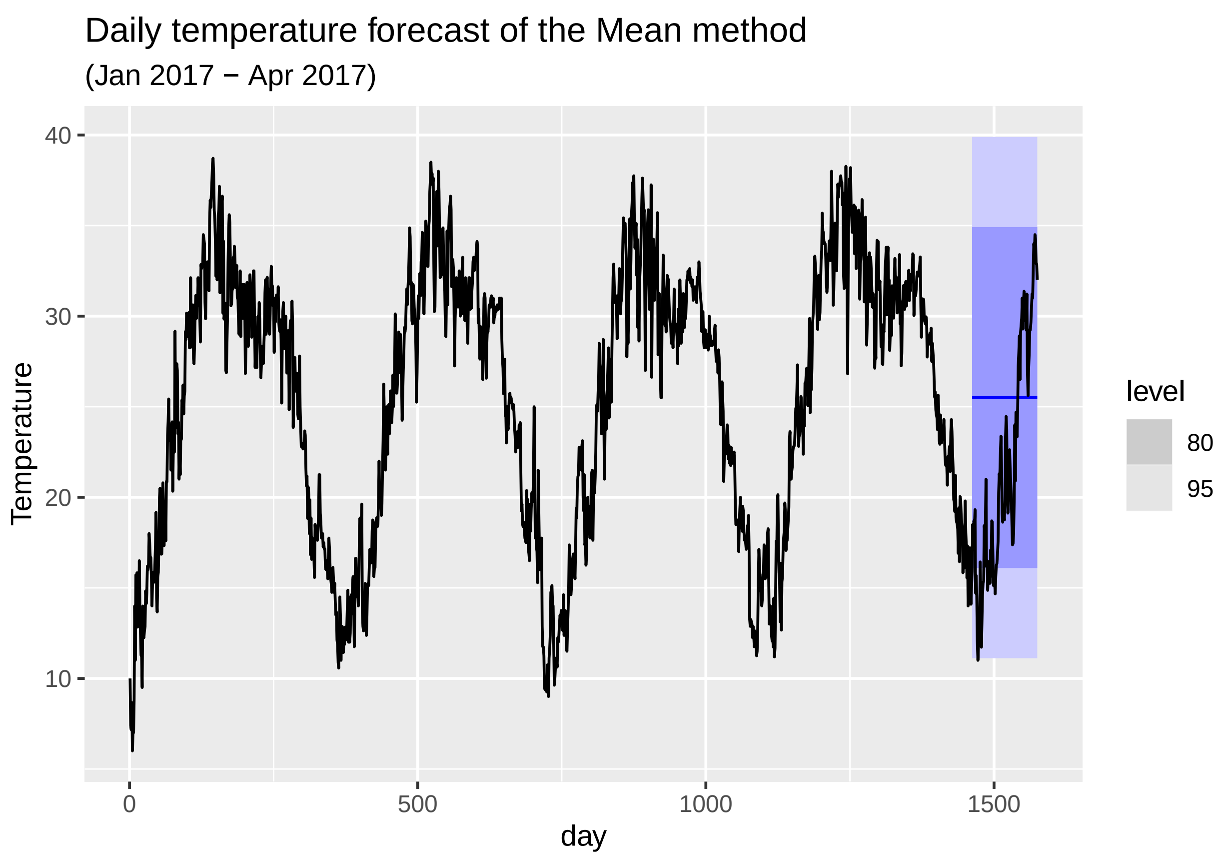

```r
# the residuals from the best method: mean method is not white noise.

#The residuals appear very auto-correlated as many lags exceed the significance threshold. This can als


# Portmanteau tests for autocorrelation
aug <- train_data |>
  model(MEAN(meantemp)) |>
  augment()

aug |> features(.innov, box_pierce, lag = 10)
```

```
## # A tibble: 1 x 3
##   .model         bp_stat bp_pvalue
##   <chr>            <dbl>     <dbl>
## 1 MEAN(meantemp)  12281.         0
```

```r
aug |> features(.innov, ljung_box, lag = 10)
```

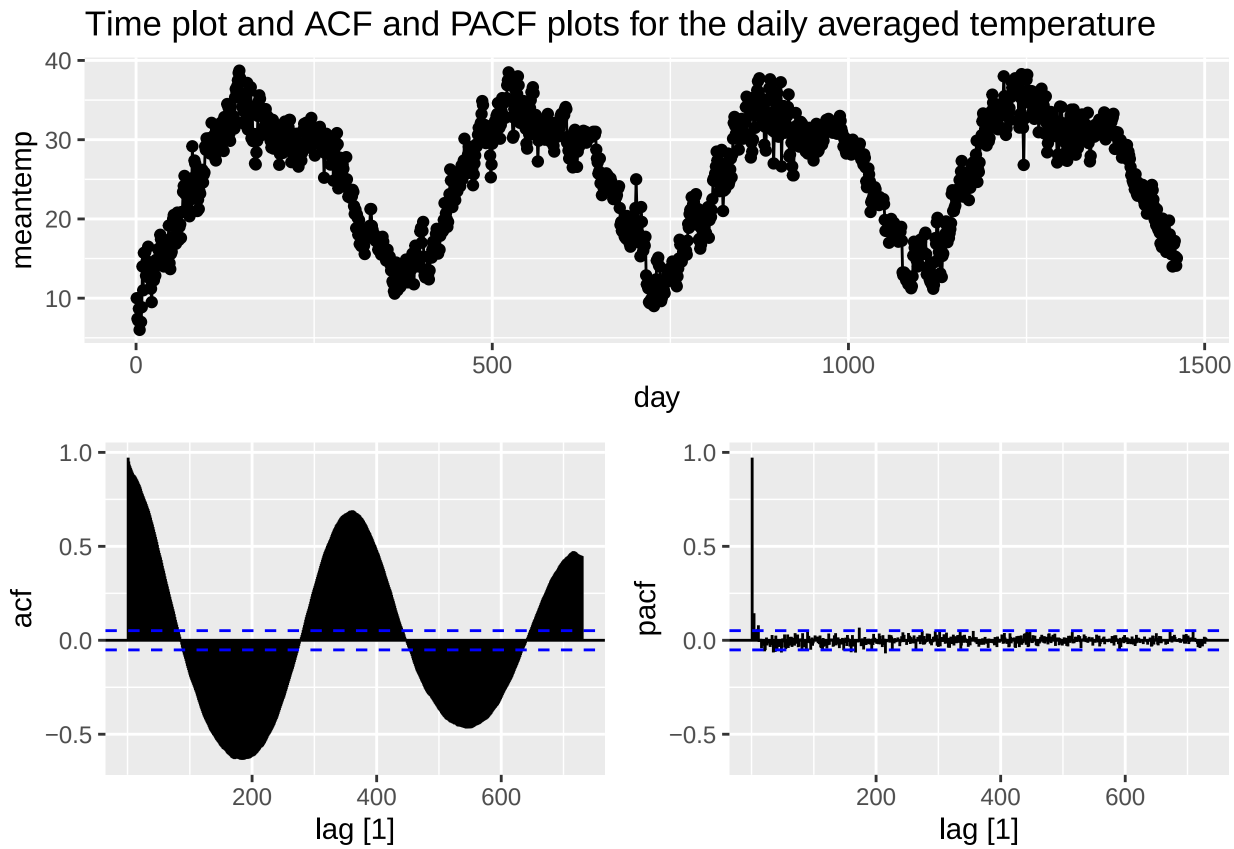
```
## # A tibble: 1 x 3
##   .model          lb_stat lb_pvalue
##   <chr>             <dbl>     <dbl>
## 1 MEAN(meantemp)   12343.         0
```

```r
# Multi-step ahead prediction intervals
#train_data |>
  #model(MEAN(meantemp)) |>
  #forecast(test_data) |>
  #hilo(95)

train_data |>
  model(MEAN(meantemp)) |>
  forecast(test_data) |>
  autoplot(full_data_tsibble) +
  labs(title="Daily temperature forecast of the Mean method",
       subtitle = "(Jan 2017 - Apr 2017)", y="Temperature" )
```



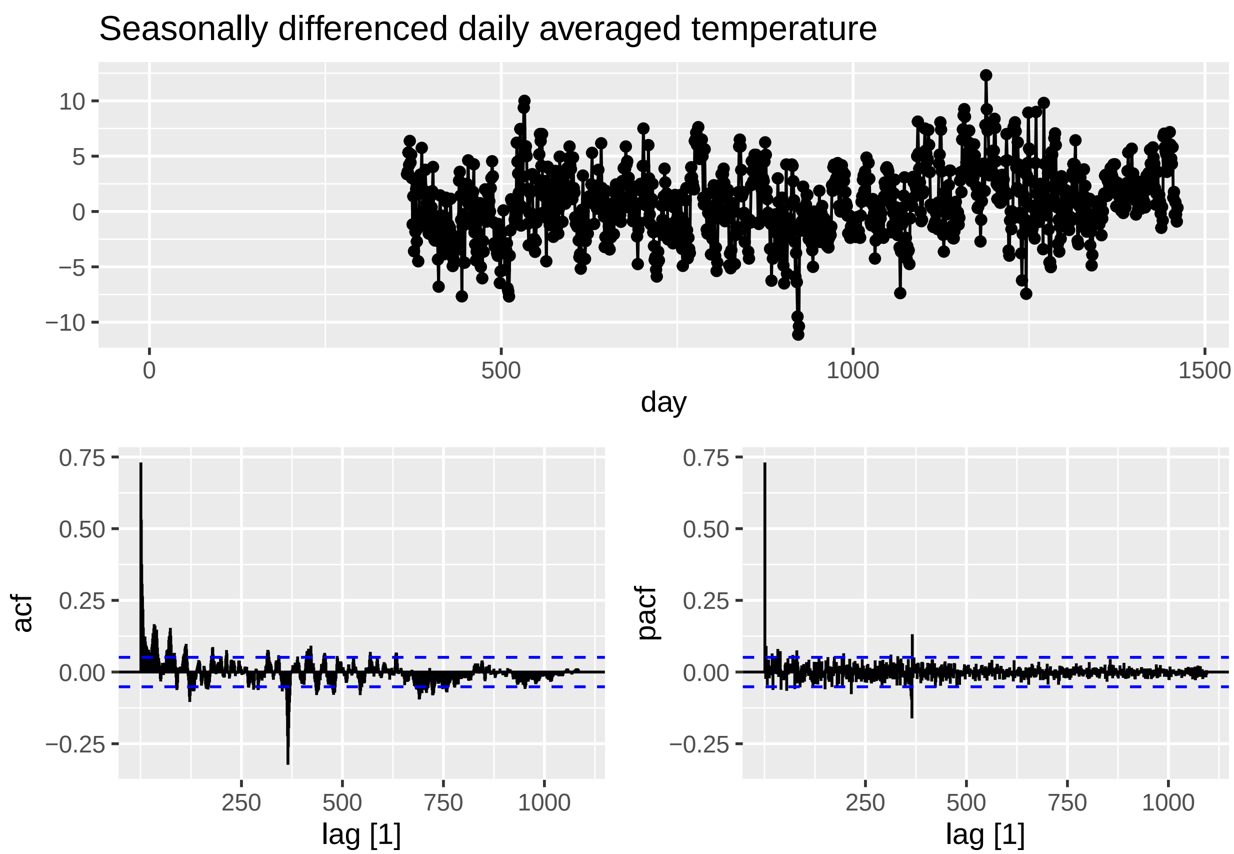
ARIMA or SARIMA models

```r
# No difference
train_data |> gg_tsdisplay(meantemp, plot_type = 'partial', lag=730)+
  labs(title="Time plot and ACF and PACF plots for the daily averaged temperature")
```



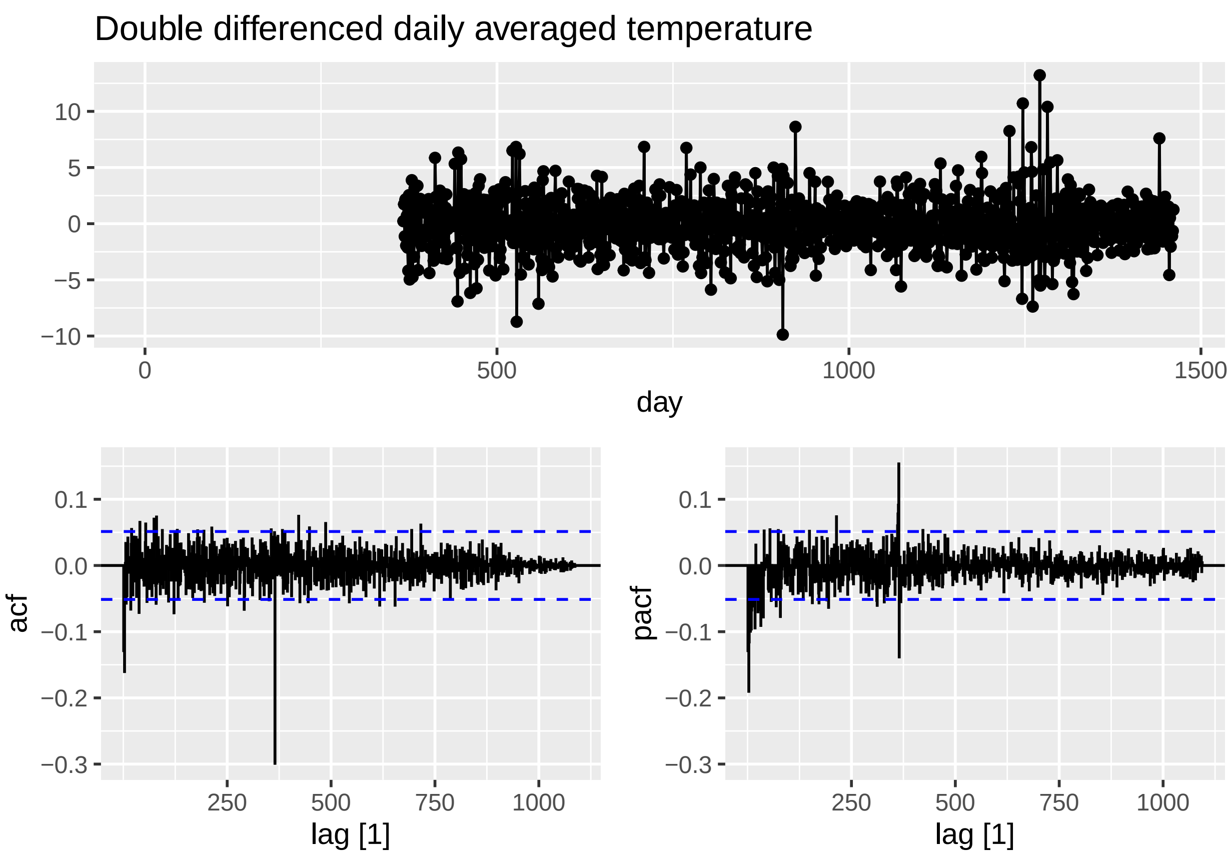

```r
#The ACF of stationary data drops to zero relatively quickly,The ACF of non-stationary data decreases s
#The data are clearly non-stationary, with strong seasonality and a nonlinear trend, so we will first t

# Seasonal difference
train_data |>
  gg_tsdisplay(difference(meantemp, 365),
               plot_type='partial', lag=1095) +
  labs(title="Seasonally differenced daily averaged temperature", y="")
```

```
## Warning: Removed 365 rows containing missing values (`geom_line()`).

## Warning: Removed 365 rows containing missing values (`geom_point()`).
```



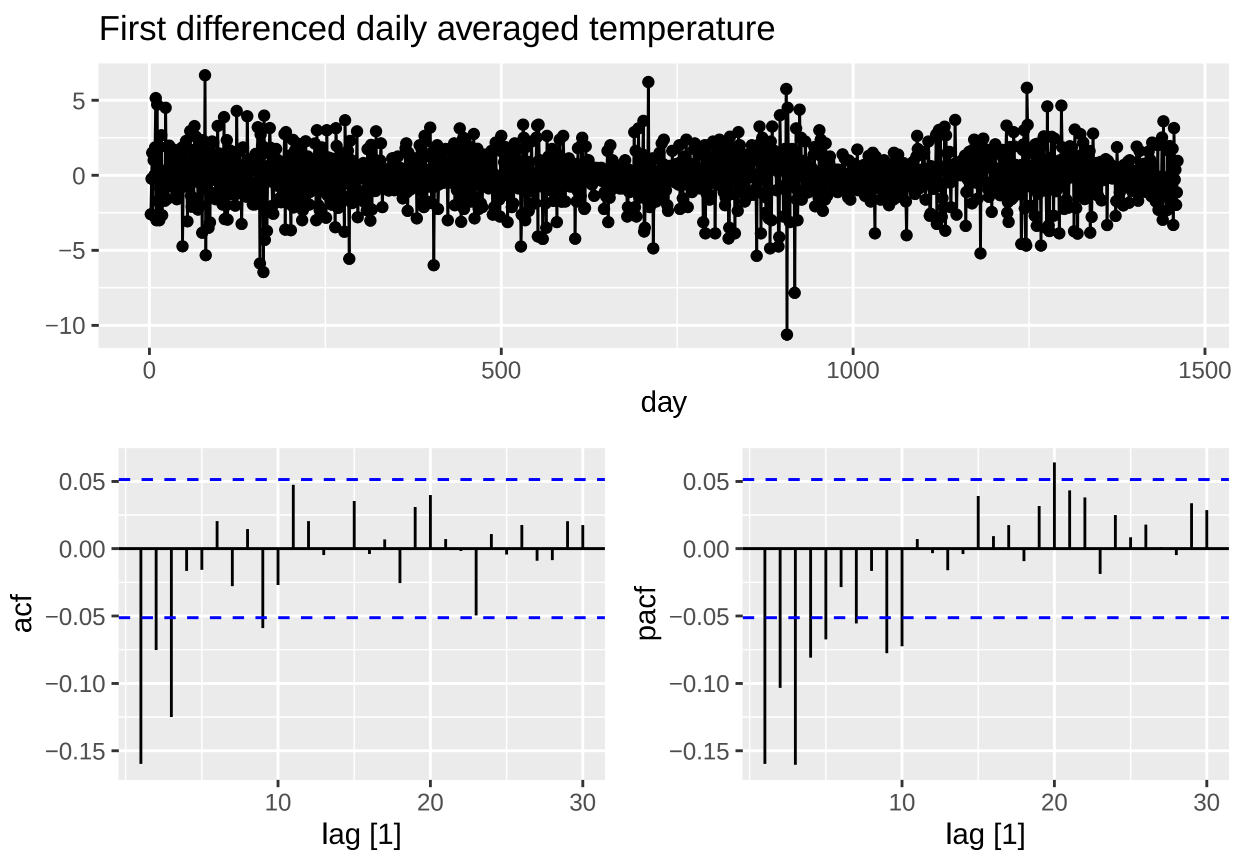

```
#These are also clearly non-stationary, so we take a further first difference

# Seasonal difference+ first difference
train_data |>
  gg_tsdisplay(difference(meantemp, 365)|> difference(),
               plot_type='partial', lag=1095) +
  labs(title="Double differenced daily averaged temperature", y="")
```

```
## Warning: Removed 366 rows containing missing values (`geom_line()`).

## Warning: Removed 366 rows containing missing values (`geom_point()`).
```




```
# First difference is used to stabilize the variance and mean.stationary.
train_data |> gg_tsdisplay(difference(meantemp), plot_type = 'partial', lag=30)+
  labs(title="First differenced daily averaged temperature", y="")
```

```
## Warning: Removed 1 row containing missing values (‘geom_line()‘).

## Warning: Removed 1 rows containing missing values (‘geom_point()‘).
```




```
arima_fit <- train_data |>
  model(arima2010 = ARIMA(meantemp ~ 1+pdq(20,1,0)),
        arima019 = ARIMA(meantemp ~ 1+pdq(0,1,9)),
        stepwise = ARIMA(meantemp))

arima_fit
```

```
## # A mable: 1 x 3
##                  arima2010                  arima019       stepwise
##                    <model>                   <model>        <model>
## 1 <ARIMA(20,1,0) w/ drift> <ARIMA(0,1,9) w/ drift> <ARIMA(2,1,2)>
```

```
glance(arima_fit) |> arrange(AICc) |> select(.model:BIC)
```

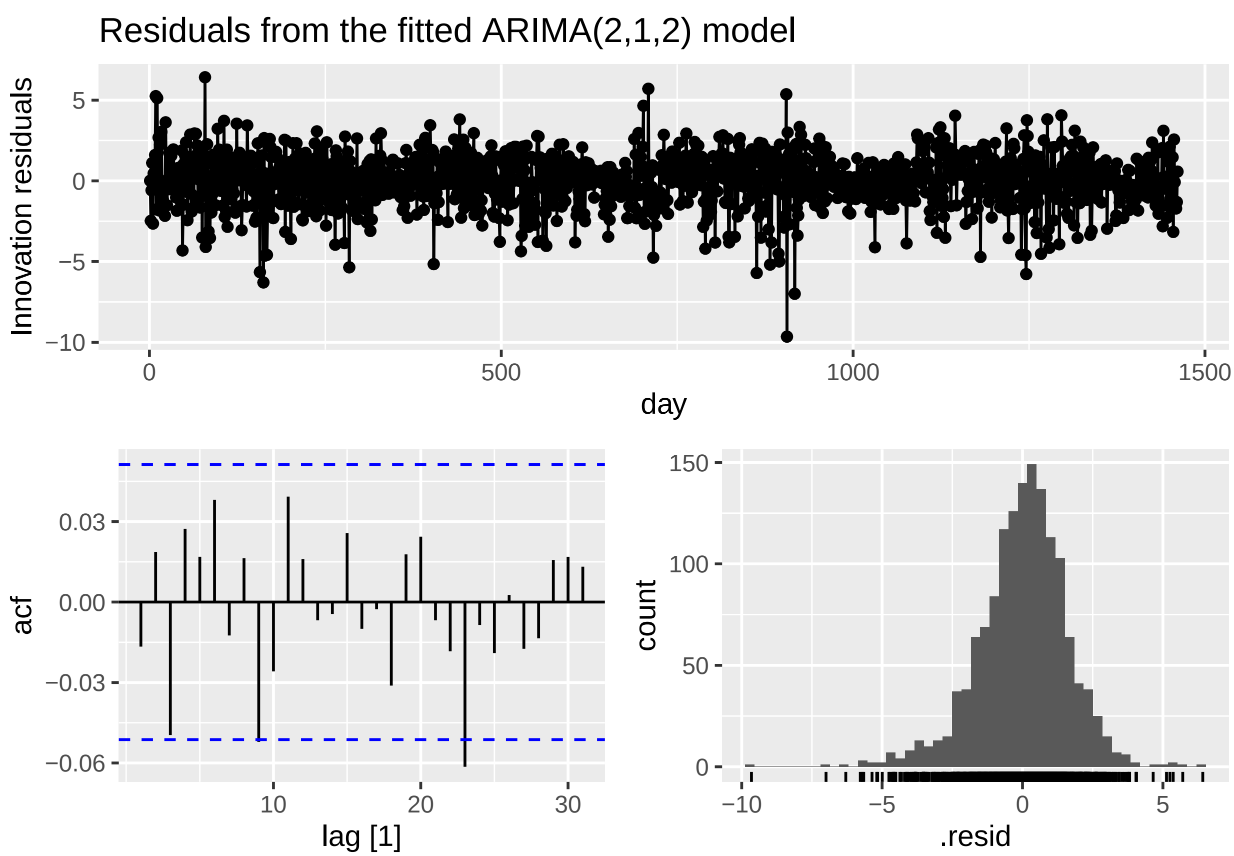
```
## # A tibble: 3 x 6
##   .model    sigma2 log_lik   AIC  AICc   BIC
##   <chr>      <dbl>   <dbl> <dbl> <dbl> <dbl>
## 1 stepwise    2.52  -2745. 5501. 5501. 5527.
## 2 arima019    2.57  -2756. 5534. 5535. 5593.
## 3 arima2010   2.56  -2746. 5537. 5537. 5653.
```

```
# Stepwise gives the best model,ARIMA(2,1,2), with the lowest AIC, AICc and BIC.

# Check the best model
```

```r
arima_fit |>
  select(stepwise) |>
  report()
```

```
## Series: meantemp
## Model: ARIMA(2,1,2)
##
## Coefficients:
##          ar1      ar2      ma1     ma2
##        1.688  -0.6950  -1.9155  0.9221
## s.e.   0.033   0.0328   0.0191  0.0189
##
## sigma^2 estimated as 2.522:  log likelihood=-2745.32
## AIC=5500.65   AICc=5500.69   BIC=5527.08
```

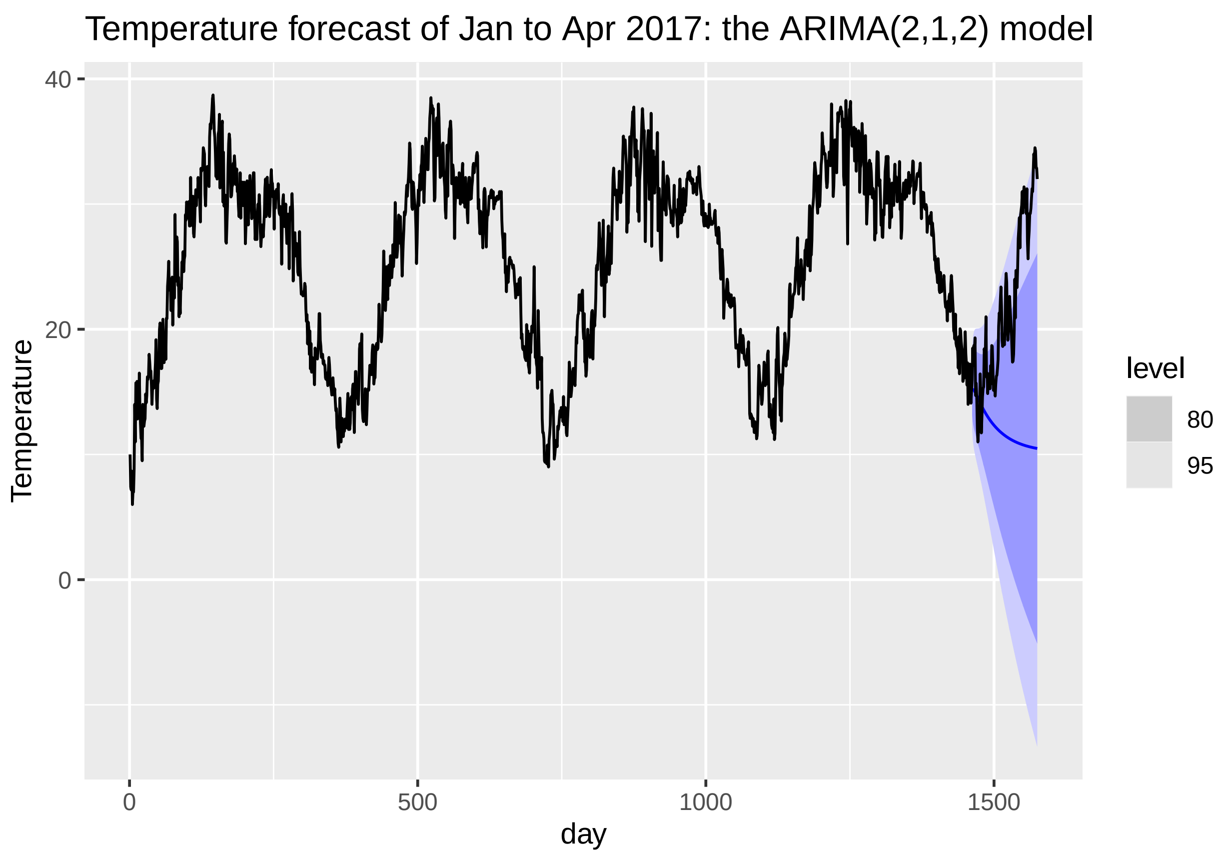
```r
# Check residuals
arima_fit |>
  select(stepwise) |>
  gg_tsresiduals()+
  labs(title="Residuals from the fitted ARIMA(2,1,2) model")
```




```r
augment(arima_fit) |>
  filter(.model=='stepwise') |>
  features(.innov, ljung_box, lag = 10, dof = 4)
```

```
## # A tibble: 1 x 3
##   .model   lb_stat lb_pvalue
##   <chr>      <dbl>     <dbl>
## 1 stepwise    13.8    0.0323
```

```
# P value is 0.03.The residuals does not pass the Ljung-Box test, and the histogram looks like left-ske


# Forecast
arima_fc<-arima_fit |>
  forecast(test_data) |>
  filter(.model=='stepwise')

arima_fc|>
  autoplot(full_data_tsibble) +
  labs(title="Temperature forecast of Jan to Apr 2017: the ARIMA(2,1,2) model", y="Temperature" )
```



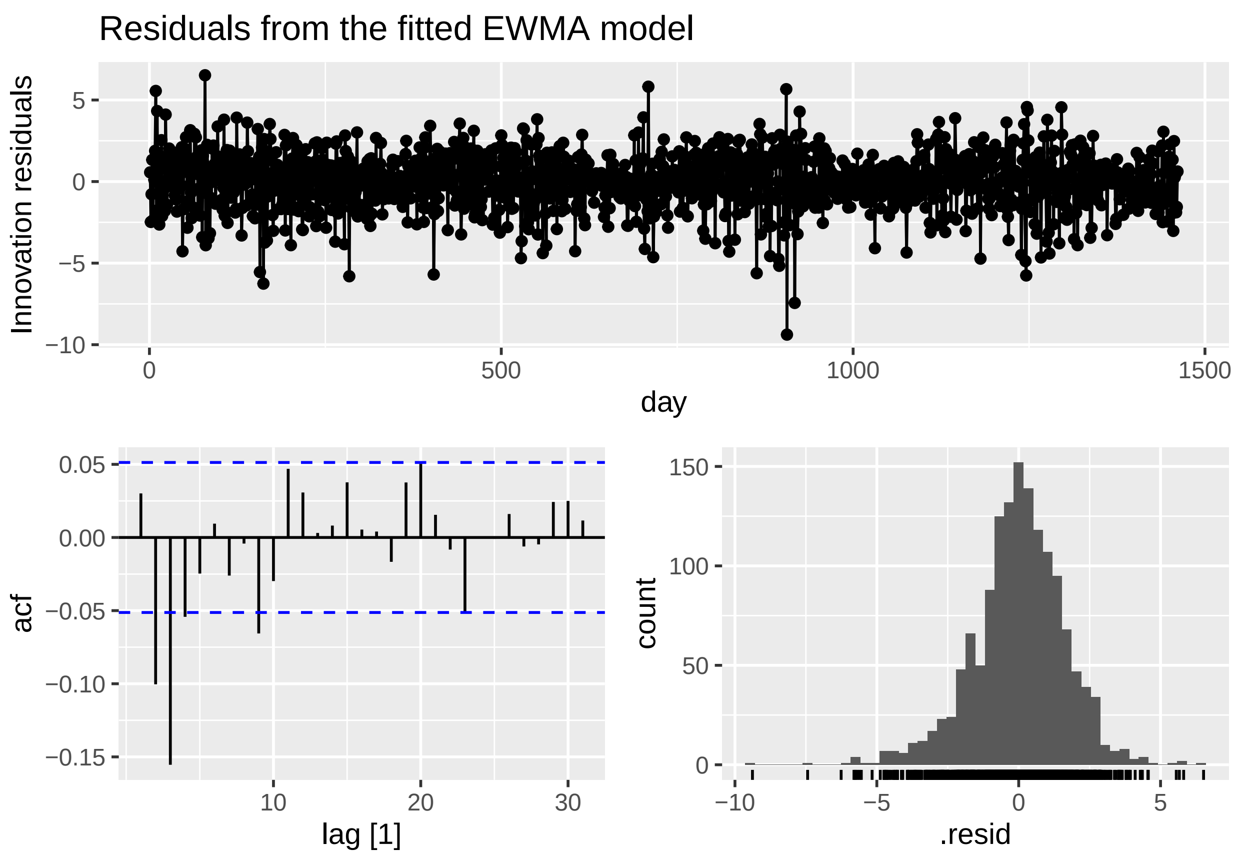

```
arima.acc <- accuracy(arima_fc$.mean,test_data$meantemp)
# RMSE is 12.24, MAE is 9.94
```

EWMA model

```r
# EWMA model
ewma_fit <- train_data %>%
  model(ETS(meantemp))
report(ewma_fit)
```

```
## Series: meantemp
## Model: ETS(A,N,N)
##   Smoothing parameters:
##     alpha = 0.7807523
##
##   Initial states:
##      l[0]
##  9.436231
##
##   sigma^2:  2.6849
##
##      AIC     AICc      BIC
## 12093.06 12093.08 12108.92
```

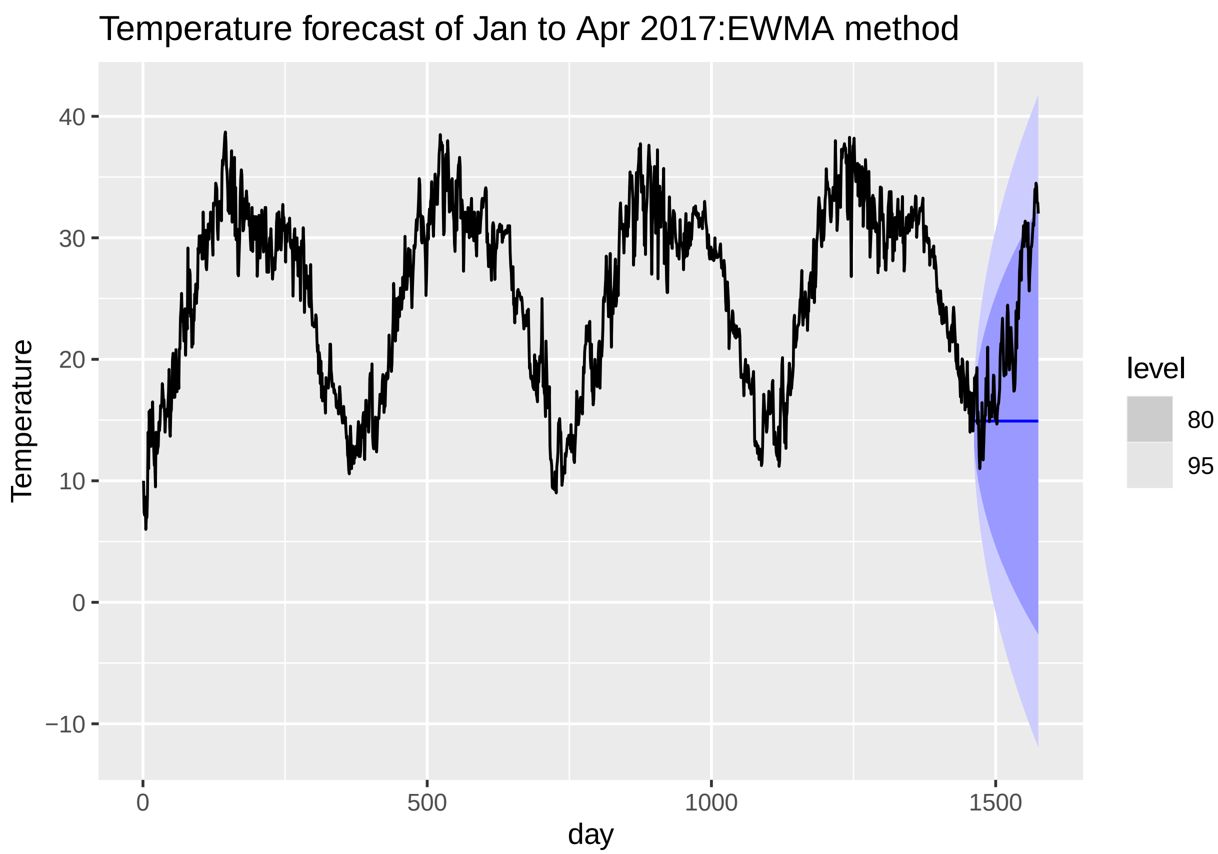
```r
#EWMA Forecast
ewma_fc<-ewma_fit %>%
  forecast(test_data)

# Check residuals
ewma_fit |>
  gg_tsresiduals()+
  labs(title="Residuals from the fitted EWMA model")
```




```
augment(ewma_fit) %>%
features(.resid, ljung_box, lag=10)
```

```
## # A tibble: 1 x 3
##   .model         lb_stat lb_pvalue
##   <chr>            <dbl>     <dbl>
## 1 ETS(meantemp)     65.5  3.27e-10
```

```
# P value is extremely small.The residuals does not pass the Ljung-Box test, and the histogram looks li

ewma_fc|>
  autoplot(full_data_tsibble) +
  labs(title="Temperature forecast of Jan to Apr 2017:EWMA method", y="Temperature" )
```




```r
ewma.acc <- accuracy(ewma_fc$.mean,test_data$meantemp)
# Comment: The RMSE of EWMA model is 9.288.
```

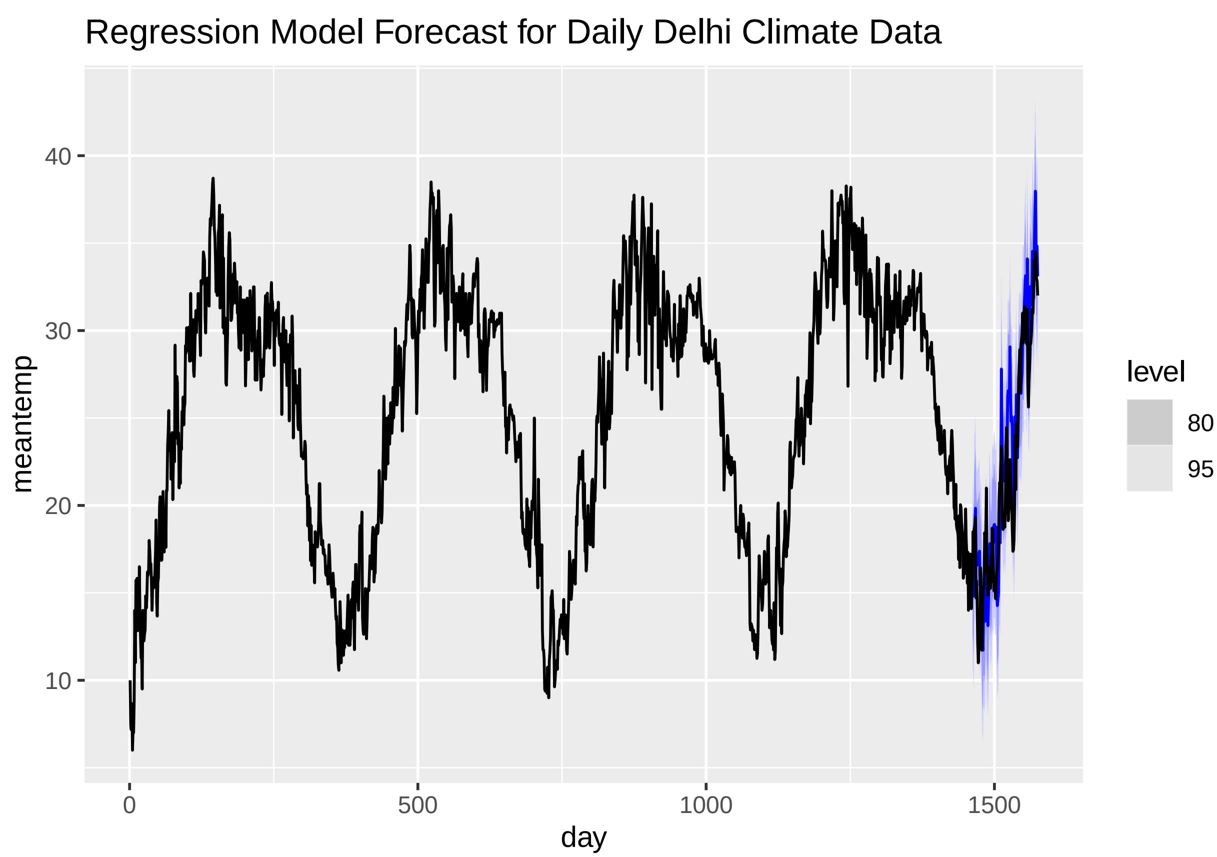
standard regression model

```r
# Fit a regression model with standard time series regression model
regression_fit_model <- train_data |>
  model(
    TSLM(meantemp ~ humidity + wind_speed + meanpressure)
  )

report(regression_fit_model)
```

```
## Series: meantemp
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -9.71261 -1.80066  0.08741  1.84391  7.89276
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  818.942022  10.399093   78.75  < 2e-16 ***
```

```
## humidity      -0.146824   0.004736  -31.00  < 2e-16 ***
## wind_speed    -0.096572   0.018937   -5.10 3.85e-07 ***
## meanpressure  -0.777462   0.010343  -75.17  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.702 on 1457 degrees of freedom
## Multiple R-squared: 0.8648,  Adjusted R-squared: 0.8645
## F-statistic:  3106 on 3 and 1457 DF, p-value: < 2.22e-16
```

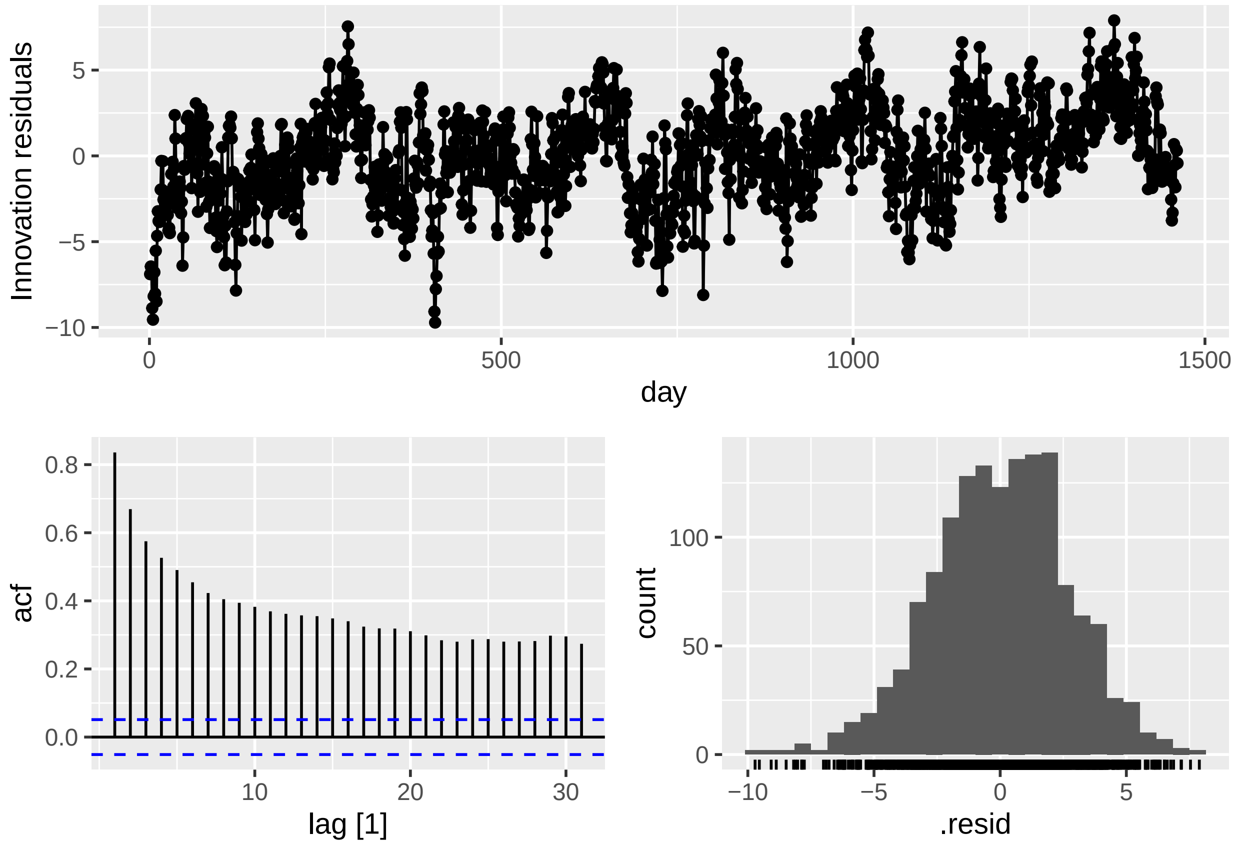
```
# Forecast using regression model on the test set
forecast_regression_model <- regression_fit_model |>
  forecast(new_data = test_data)

# Plot regression forecasts
forecast_regression_model |>
  autoplot(full_data_tsibble) +
  labs(title = "Regression Model Forecast for Daily Delhi Climate Data")
```




```
# accuracy
forecast_regression_model.acc <- accuracy(forecast_regression_model$.mean, test_data$meantemp)
forecast_regression_model.acc
```

```
##                  ME     RMSE      MAE       MPE     MAPE
## Test set -1.056238 2.852384 2.326552 -5.682515 12.11451
```

```r
# check residual plot
regression_fit_model |> gg_tsresiduals()
```




```r
# check error term

augment(regression_fit_model) |>
  features(.innov, ljung_box, lag = 10)
```

```
## # A tibble: 1 x 3
##   .model                                              lb_stat lb_pvalue
##   <chr>                                                 <dbl>     <dbl>
## 1 TSLM(meantemp ~ humidity + wind_speed + meanpressure)   4175.         0
```

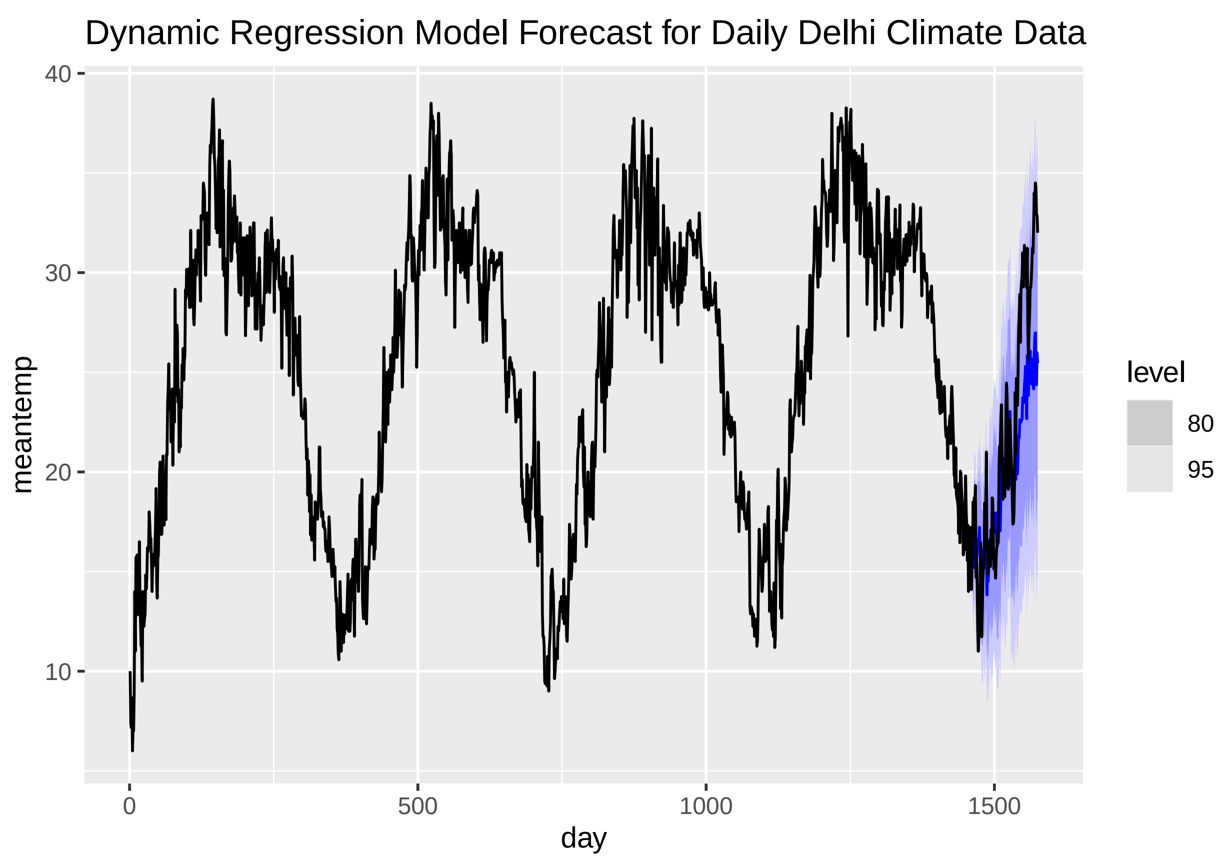
p-value is 0, can reject the null hypothesis, it shows the error term does not follow the white noise behavior.

dynamic regression model

```r
# Fit a regression model with the SARIMA errors process
dynamic_reg_fit_model <- train_data |>
  model(ARIMA(meantemp ~ humidity + wind_speed + meanpressure)
        )

report(dynamic_reg_fit_model)
```

```
## Series: meantemp
## Model: LM w/ ARIMA(2,1,1) errors
##
## Coefficients:
##          ar1      ar2      ma1  humidity  wind_speed  meanpressure
##       0.7208  -0.1472  -0.8136   -0.1324     -0.0400       -0.2375
## s.e.  0.0388   0.0286   0.0303    0.0040      0.0079        0.0197
##
## sigma^2 estimated as 1.326:  log likelihood=-2274.55
## AIC=4563.1   AICc=4563.18   BIC=4600.1
```

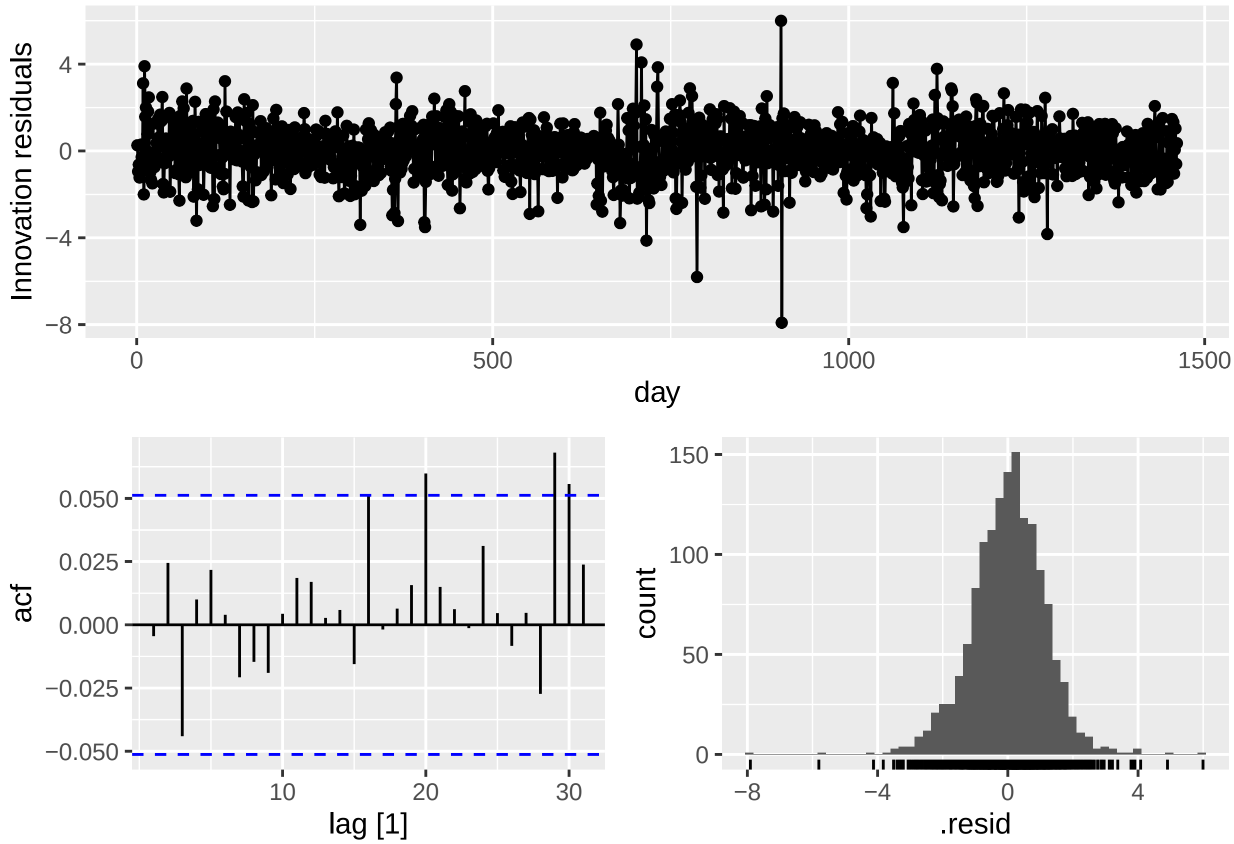
```
# Forecast using dynamic regression model on the test set
forecast_dynamic_reg_model <- dynamic_reg_fit_model |>
  forecast(new_data = test_data)

# Plot dynamic regression forecasts
forecast_dynamic_reg_model |>
  autoplot(full_data_tsibble) +
  labs(title = "Dynamic Regression Model Forecast for Daily Delhi Climate Data")
```




```
forecast_dynamic_reg_model.acc <- accuracy(forecast_dynamic_reg_model$.mean, test_data$meantemp)

forecast_dynamic_reg_model.acc
```

```
##                ME     RMSE      MAE      MPE     MAPE
## Test set 1.914062 3.873715 3.050454 5.341978 13.10199
```

```r
# check residual plot
dynamic_reg_fit_model |> gg_tsresiduals()
```




```r
augment(dynamic_reg_fit_model) |>
  features(.innov, ljung_box, dof = 4, lag = 8)
```

```
## # A tibble: 1 x 3
##   .model                                                  lb_stat lb_pvalue
##   <chr>                                                     <dbl>     <dbl>
## 1 ARIMA(meantemp ~ humidity + wind_speed + meanpressure)     5.57     0.234
```

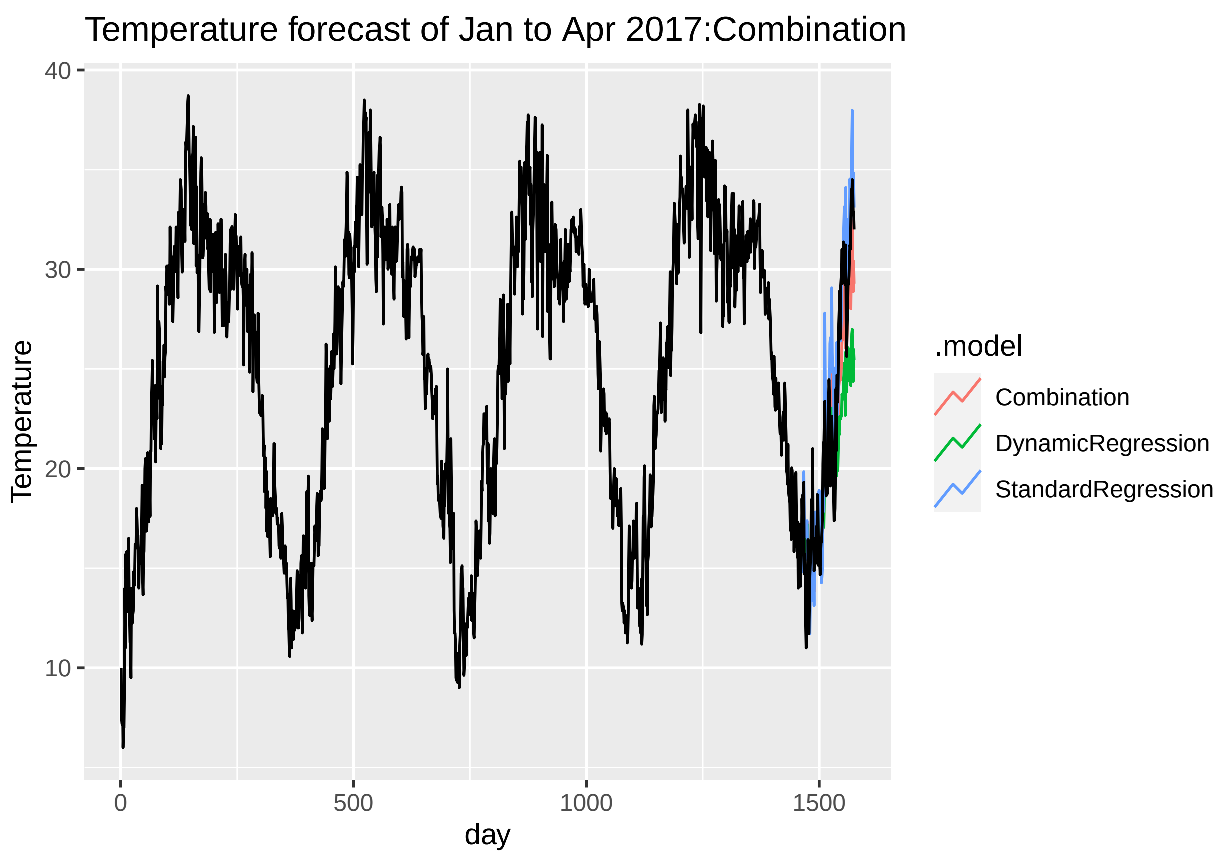
The p-value is 0.885, which is larger than 0.05, thus we cannot reject null hypothesis, and it shows the error term which follow ARIMA(1,1,3) model has white noise behavior.

From the residual plot of the fitted dynamic regression model, we can see there is barely heteroscedasticity in the residuals. The model also has few significant autocorrelation in the residuals, and the histogram of the residuals shows normal distribution. It shows ARIMA errors follow the white noise behavior very closely.

Thus, we can indicate that dynamic regression model somehow adequately addressed the autocorrelations seen in the standard time series regression model, because the SARIMA error term in dynamic regression model capture these information which does not explain in the standard regression time series model.

Combination of Dynamic regression and standard regression

```
# Dynamic regression plus standard regression
com_fc <- train_data %>%
model(
DynamicRegression = ARIMA(meantemp ~ humidity + wind_speed + meanpressure),
StandardRegression =TSLM(meantemp ~ humidity + wind_speed + meanpressure)
) %>%
mutate(
Combination = (DynamicRegression + StandardRegression)/2
) %>%
forecast(test_data)

com_fc %>% autoplot(full_data_tsibble, level = NULL) +
labs(y = "Temperature",title = "Temperature forecast of Jan to Apr 2017:Combination")
```




```
combination.acc <- accuracy(com_fc$.mean,test_data$meantemp)
# Comment: The RMSE of Combination model is 3.874.
```

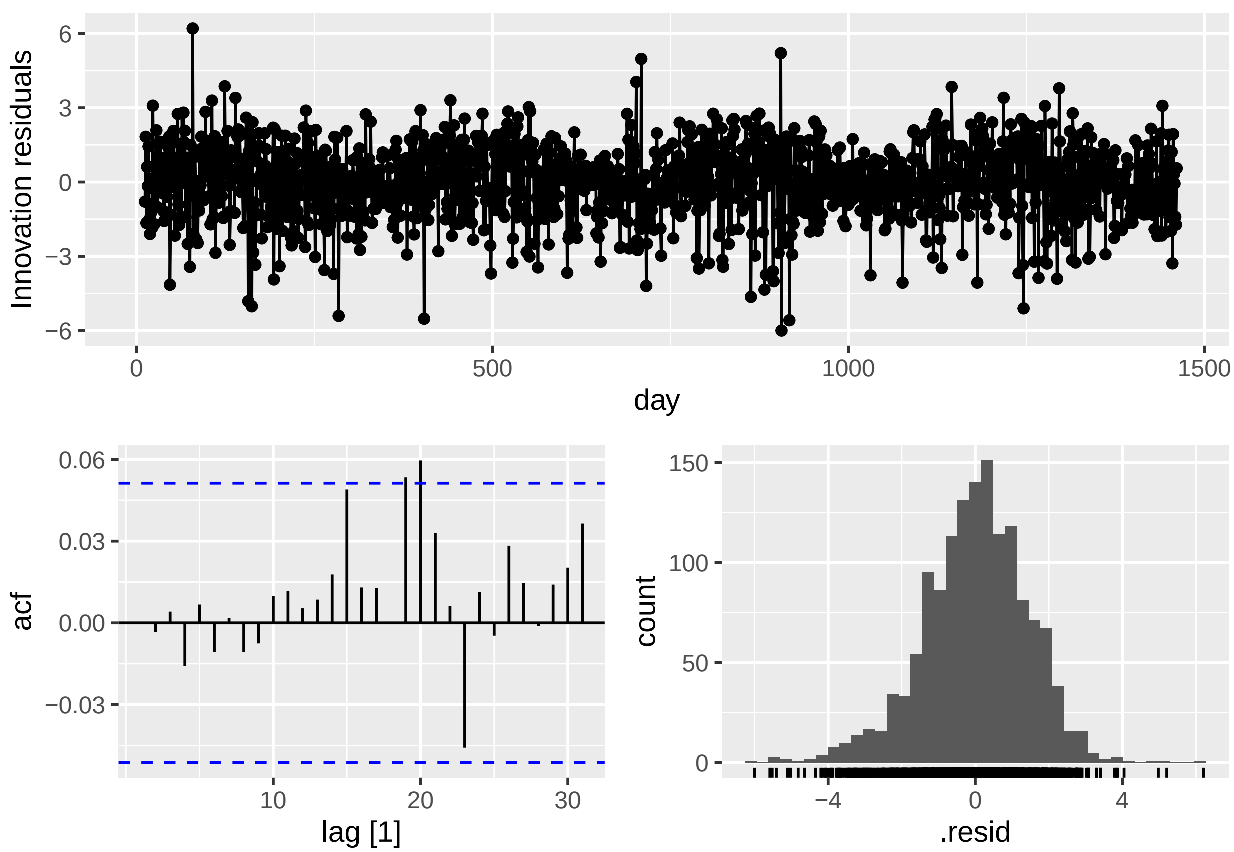
NNAR model

```r
## NNAR model
NNAR_fit <- train_data |>
  model(NNETAR(meantemp))

NNAR_fc <- NNAR_fit |>
  forecast(new_data = test_data)

# View(NNAR_fc)
accuracy_NNAR <- fabletools::accuracy(NNAR_fc, full_data_tsibble)
accuracy_NNAR
```

```
## # A tibble: 1 x 10
##   .model           .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>            <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 NNETAR(meantemp) Test   3.48  6.13  4.58  10.6  18.5  3.70  3.68 0.932
```

```r
NNAR_fit |> gg_tsresiduals()
```

```
## Warning: Removed 11 rows containing missing values (`geom_line()`).

## Warning: Removed 11 rows containing missing values (`geom_point()`).

## Warning: Removed 11 rows containing non-finite values (`stat_bin()`).
```



Prophet model

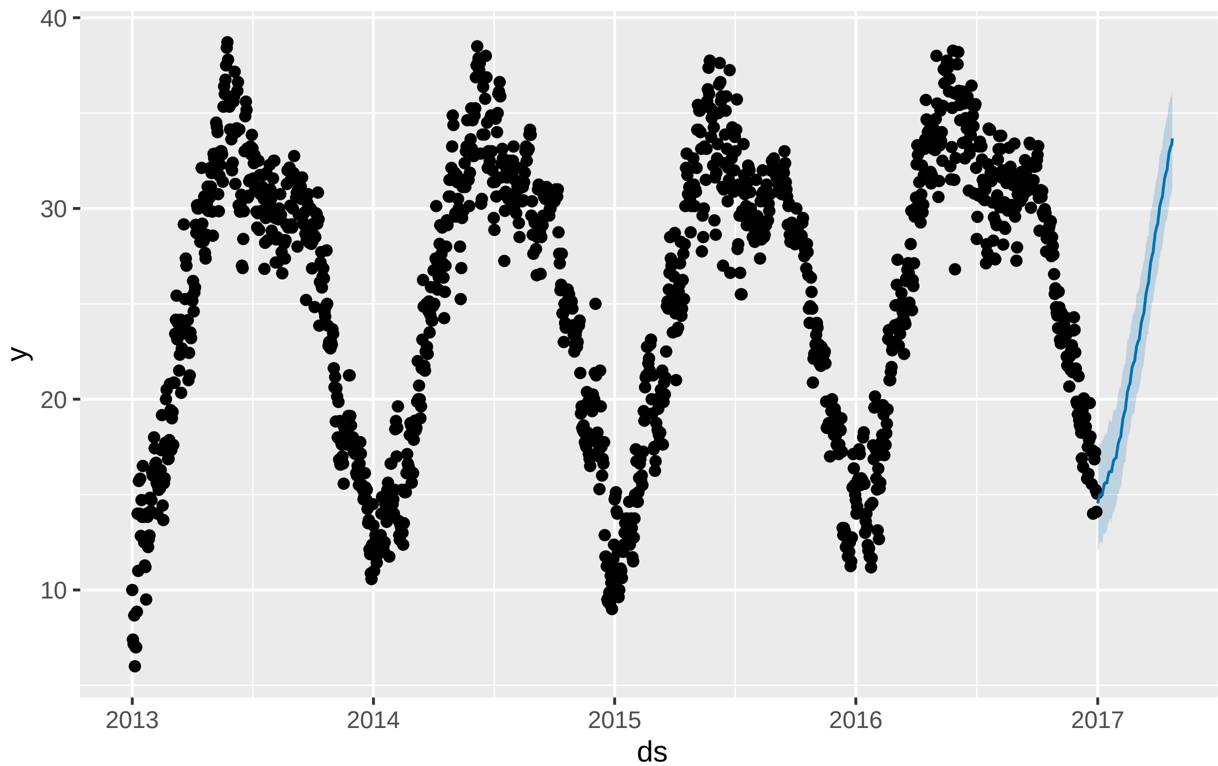
```r
a <- train_data$meantemp

train <- as.data.frame(a)
train <- cbind(ds = train_data$date, train)
rownames(train) <- 1:nrow(train)
colnames(train) <- c ("ds", "y")
head(train)
```

```
##           ds         y
## 1 2013-01-01 10.000000
## 2 2013-01-02  7.400000
## 3 2013-01-03  7.166667
## 4 2013-01-04  8.666667
## 5 2013-01-05  6.000000
## 6 2013-01-06  7.000000
```

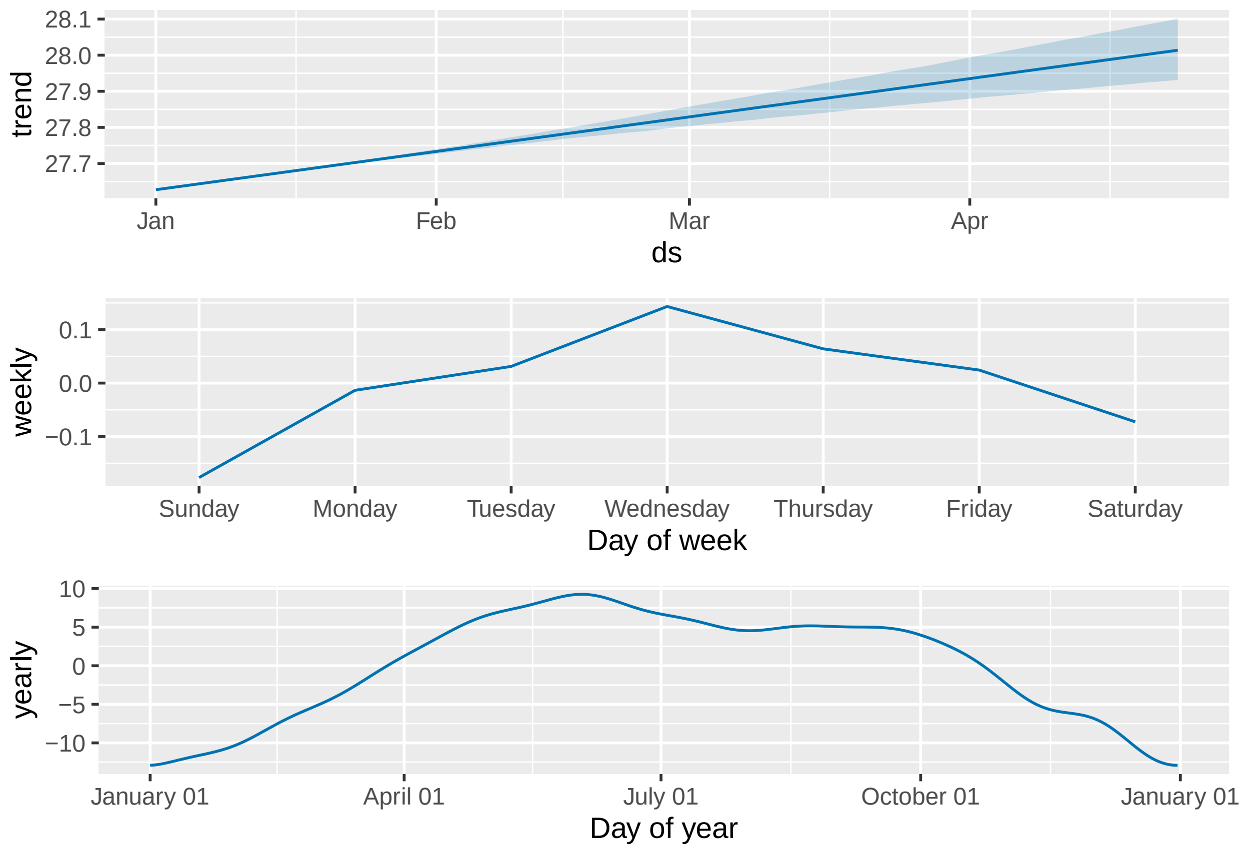
```r
# fit the prophet model
fit.prophet <- prophet(train)
```

```
## Disabling daily seasonality. Run prophet with daily.seasonality=TRUE to override this.
```

```r
future <- data.frame(ds = test_data$date)
fit.prophet_fc <- predict(fit.prophet, future)

plot(fit.prophet, fit.prophet_fc)
```




```r
## prophet decomposition
prophet_plot_components(fit.prophet, fit.prophet_fc)
```




```
# accuracy
accuracy_fit_prophet <- forecast::accuracy(fit.prophet_fc$yhat, test_data$meantemp)
accuracy_fit_prophet
```

```
##                   ME     RMSE      MAE       MPE     MAPE
## Test set -1.151801 2.725525 2.229138 -7.052589 11.90307
```

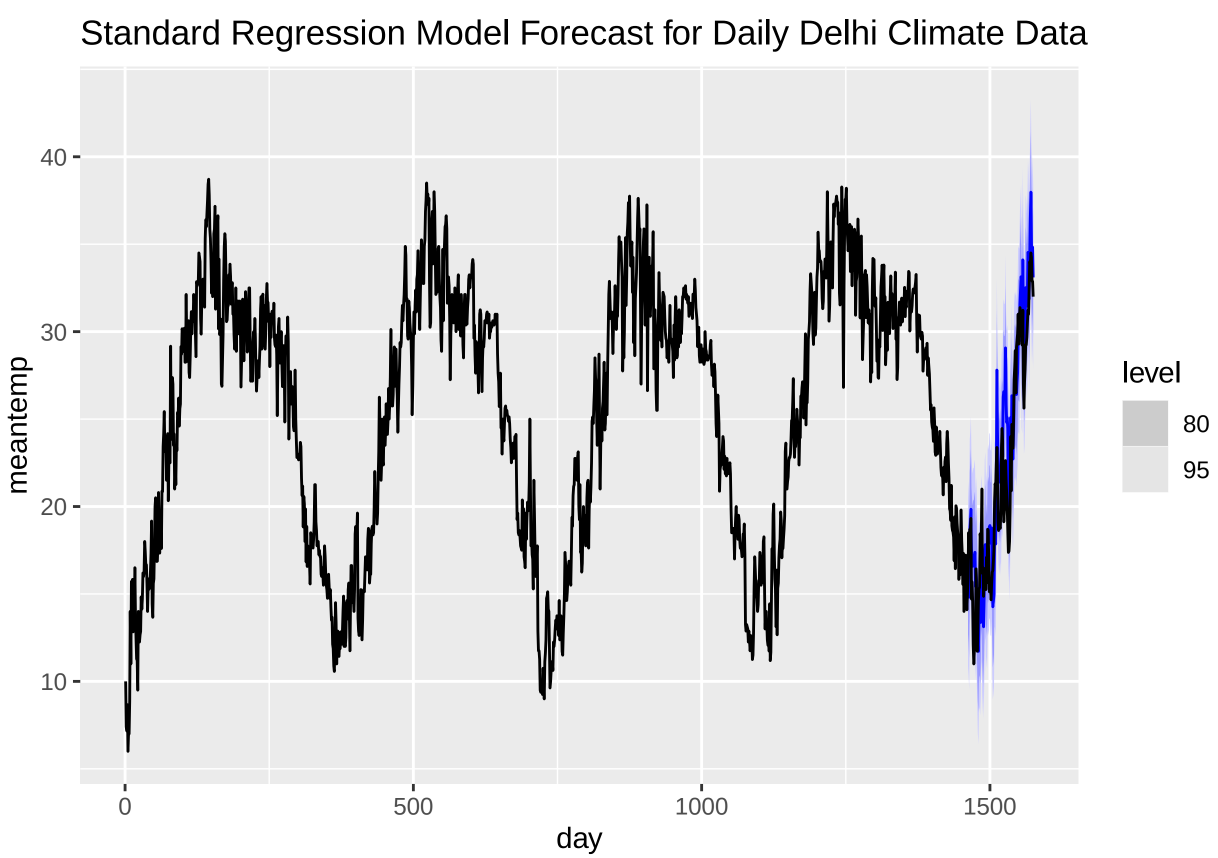
standard regression model

```
# Fit a regression model with standard time series regression model
regression_fit_model <- train_data |>
  model(
    TSLM(meantemp ~ humidity + wind_speed + meanpressure)
  )

report(regression_fit_model)
```

```
## Series: meantemp
## Model: TSLM
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -9.71261 -1.80066  0.08741  1.84391  7.89276
```

```
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  818.942022  10.399093   78.75  < 2e-16 ***
## humidity      -0.146824   0.004736  -31.00  < 2e-16 ***
## wind_speed    -0.096572   0.018937   -5.10 3.85e-07 ***
## meanpressure  -0.777462   0.010343  -75.17  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.702 on 1457 degrees of freedom
## Multiple R-squared: 0.8648,  Adjusted R-squared: 0.8645
## F-statistic:  3106 on 3 and 1457 DF, p-value: < 2.22e-16
```

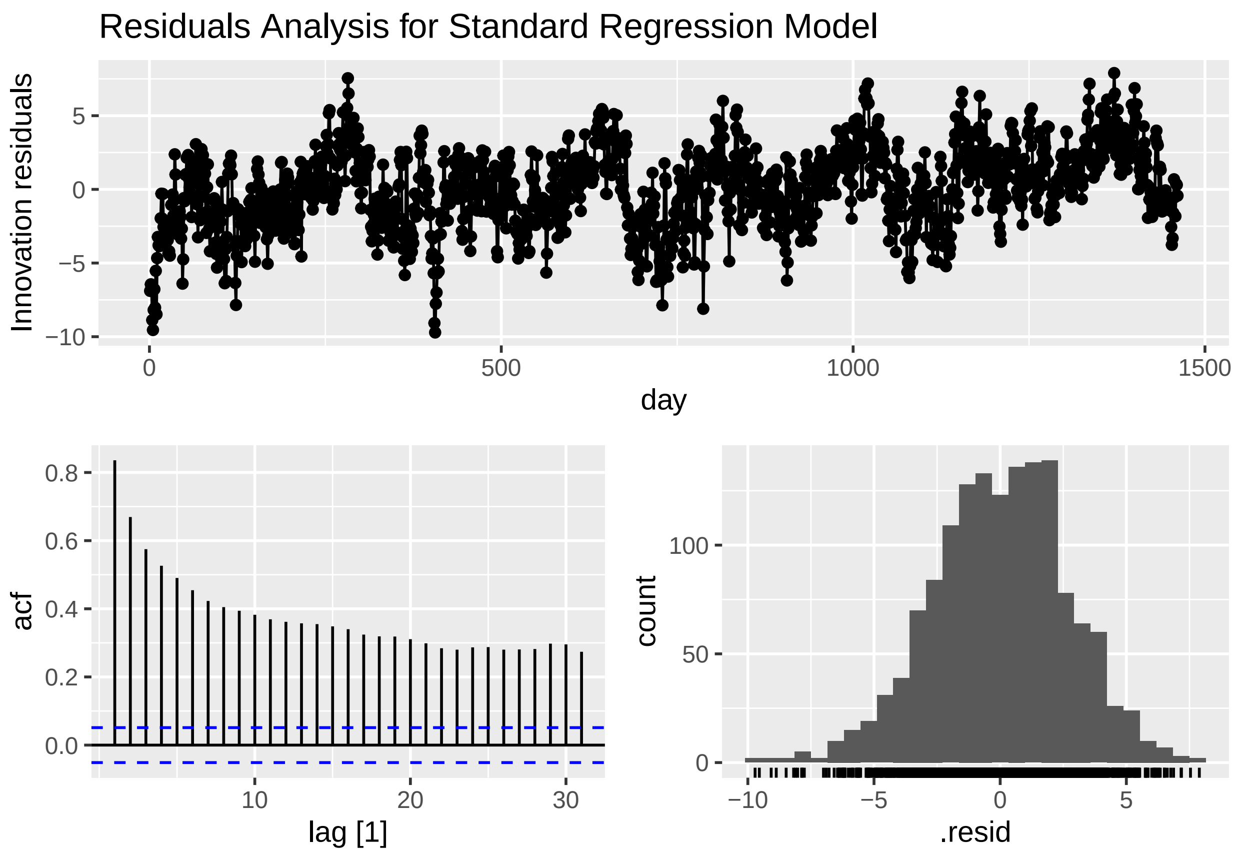
```r
# Forecast using regression model on the test set
forecast_regression_model <- regression_fit_model |>
  forecast(new_data = test_data)

# Plot regression forecasts
forecast_regression_model |>
  autoplot(full_data_tsibble) +
  labs(title = "Standard Regression Model Forecast for Daily Delhi Climate Data")
```




```r
# accuracy
forecast_regression_model.acc <- accuracy(forecast_regression_model$.mean, test_data$meantemp)
forecast_regression_model.acc
```

```
##                  ME     RMSE      MAE       MPE     MAPE
## Test set -1.056238 2.852384 2.326552 -5.682515 12.11451
```

```r
# check residual plot
regression_fit_model |> gg_tsresiduals() +
  labs(title = "Residuals Analysis for Standard Regression Model")
```




```r
# check error term

augment(regression_fit_model) |>
  features(.innov, ljung_box, lag = 10)
```

```
## # A tibble: 1 x 3
##   .model                                                  lb_stat lb_pvalue
##   <chr>                                                     <dbl>     <dbl>
## 1 TSLM(meantemp ~ humidity + wind_speed + meanpressure)   4175.          0
```

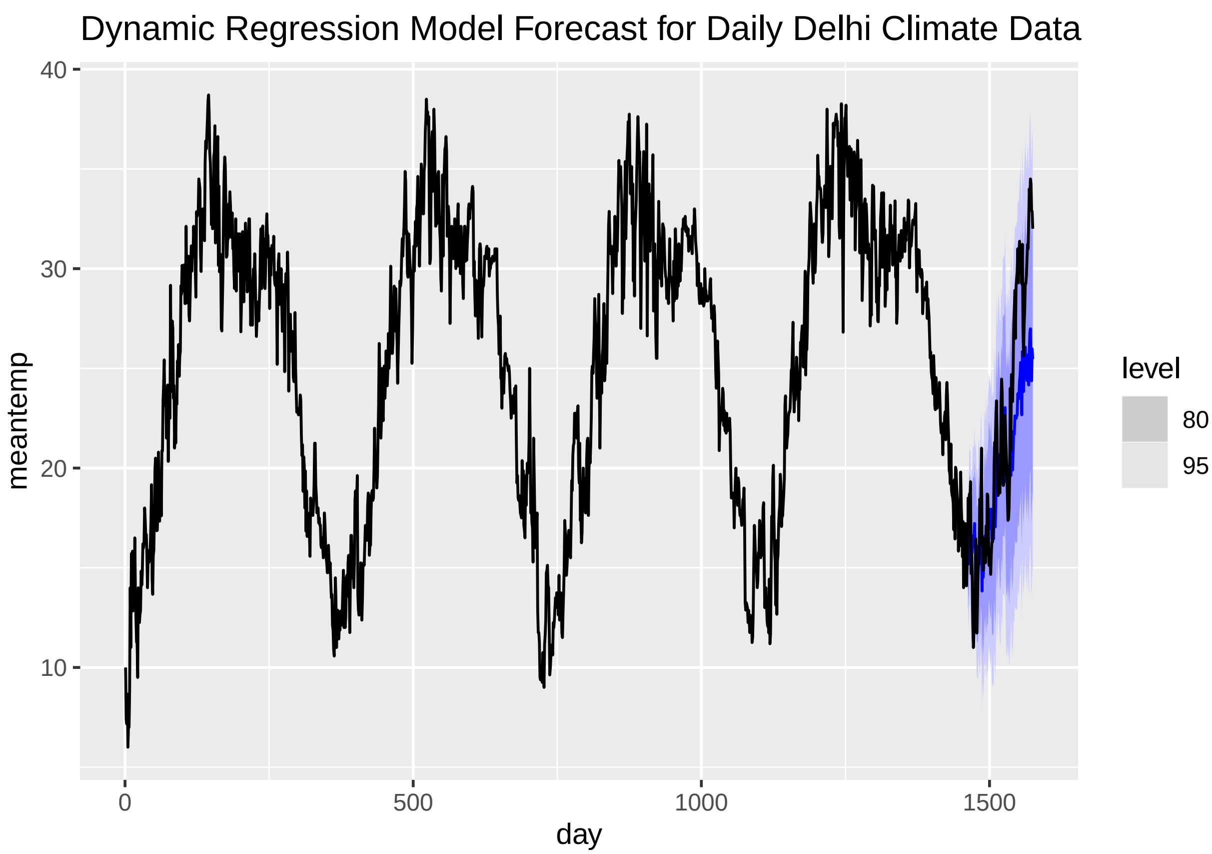
p-value is 0, can reject the null hypothesis, it shows the error term does not follow the white noise behavior.

dynamic regression model

```r
# Fit a regression model with the SARIMA errors process
dynamic_reg_fit_model <- train_data |>
  model(ARIMA(meantemp ~ humidity + wind_speed + meanpressure)
        )

report(dynamic_reg_fit_model)
```

```
## Series: meantemp
## Model: LM w/ ARIMA(2,1,1) errors
##
## Coefficients:
##          ar1      ar2      ma1  humidity  wind_speed  meanpressure
##       0.7208  -0.1472  -0.8136   -0.1324     -0.0400       -0.2375
## s.e.  0.0388   0.0286   0.0303    0.0040      0.0079        0.0197
##
## sigma^2 estimated as 1.326:  log likelihood=-2274.55
## AIC=4563.1   AICc=4563.18   BIC=4600.1
```

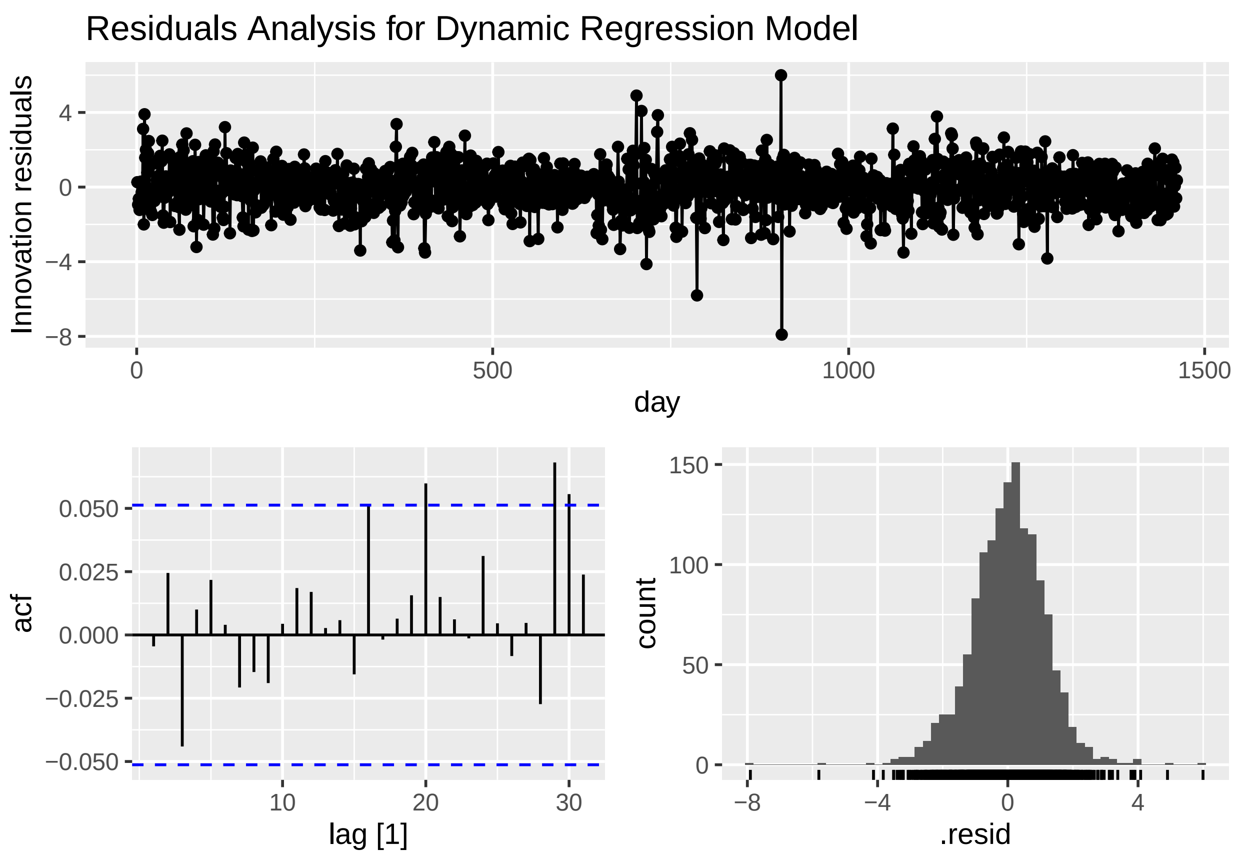
```r
# Forecast using dynamic regression model on the test set
forecast_dynamic_reg_model <- dynamic_reg_fit_model |>
  forecast(new_data = test_data)

# Plot dynamic regression forecasts
forecast_dynamic_reg_model |>
  autoplot(full_data_tsibble) +
  labs(title = "Dynamic Regression Model Forecast for Daily Delhi Climate Data")
```




```r
forecast_dynamic_reg_model.acc <- accuracy(forecast_dynamic_reg_model$.mean, test_data$meantemp)

forecast_dynamic_reg_model.acc
```

```
##                 ME     RMSE      MAE      MPE     MAPE
## Test set 1.914062 3.873715 3.050454 5.341978 13.10199
```

```r
# check residual plot
dynamic_reg_fit_model |> gg_tsresiduals() +
  labs(title = "Residuals Analysis for Dynamic Regression Model")
```




```r
augment(dynamic_reg_fit_model) |>
  features(.innov, ljung_box, dof = 3, lag = 8)
```

```
## # A tibble: 1 x 3
##   .model                                                  lb_stat lb_pvalue
##   <chr>                                                     <dbl>     <dbl>
## 1 ARIMA(meantemp ~ humidity + wind_speed + meanpressure)    5.57     0.351
```

The p-value is 0.351, which is larger than 0.05, thus we cannot reject null hypothesis, and it shows the error term which follow ARIMA(2,1,1) model has white noise behavior.

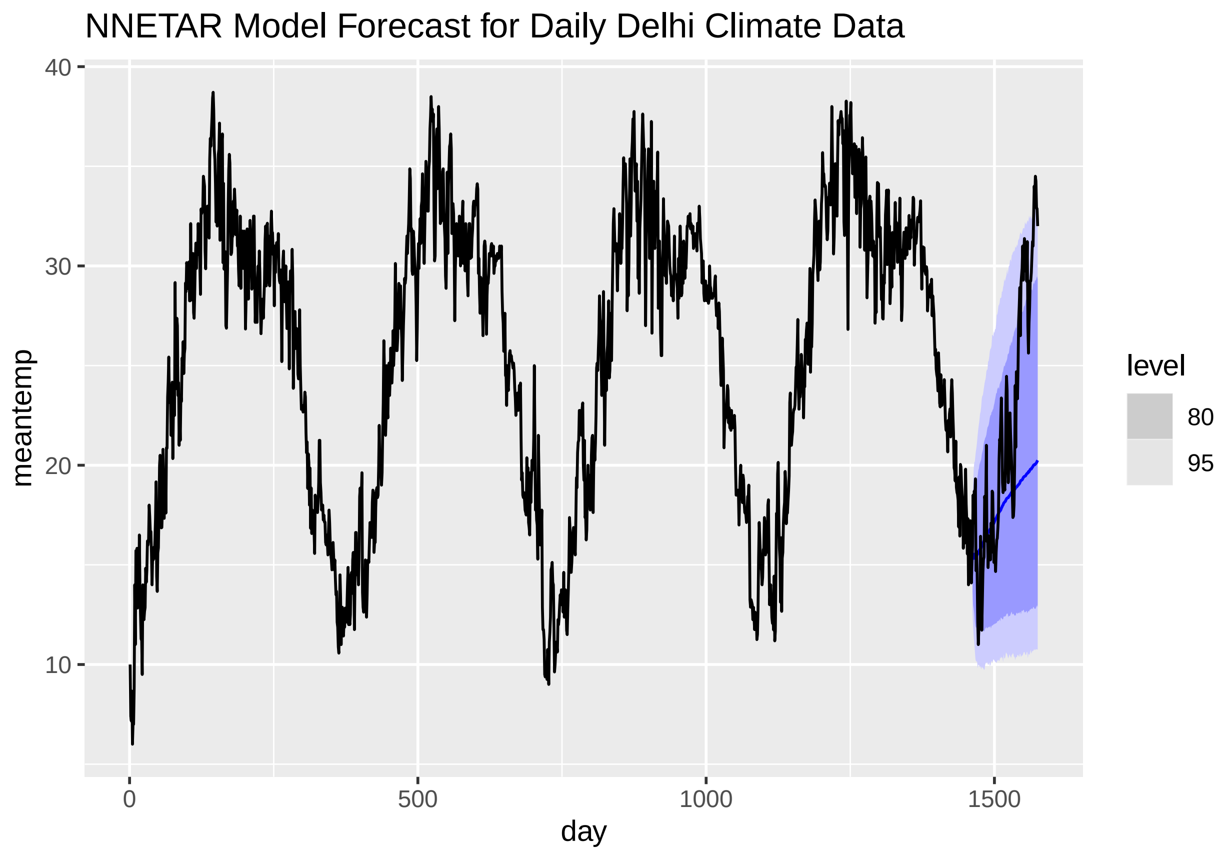
From the residual plot of the fitted dynamic regression model, we can see there is barely heteroscedasticity in the residuals. The model also has few significant autocorrelation in the residuals, and the histogram of the residuals shows normal distribution. It shows ARIMA errors follow the white noise behavior very closely.

Thus, we can indicate that dynamic regression model somehow adequately addressed the autocorrelations seen in the standard time series regression model, because the SARIMA error term in dynamic regression model capture these information which does not explain in the standard regression time series model.

NNETAR model

```r
set.seed(6510)
## NNAR model
NNAR_fit <- train_data |>
```

```
  model(NNETAR(meantemp))

NNAR_fc <- NNAR_fit |>
  forecast(new_data = test_data)

NNAR_fc |>
  autoplot(full_data_tsibble) +
  labs(title = "NNETAR Model Forecast for Daily Delhi Climate Data")
```




```
# View(NNAR_fc)
accuracy_NNAR <- fabletools::accuracy(NNAR_fc, full_data_tsibble)
accuracy_NNAR
```

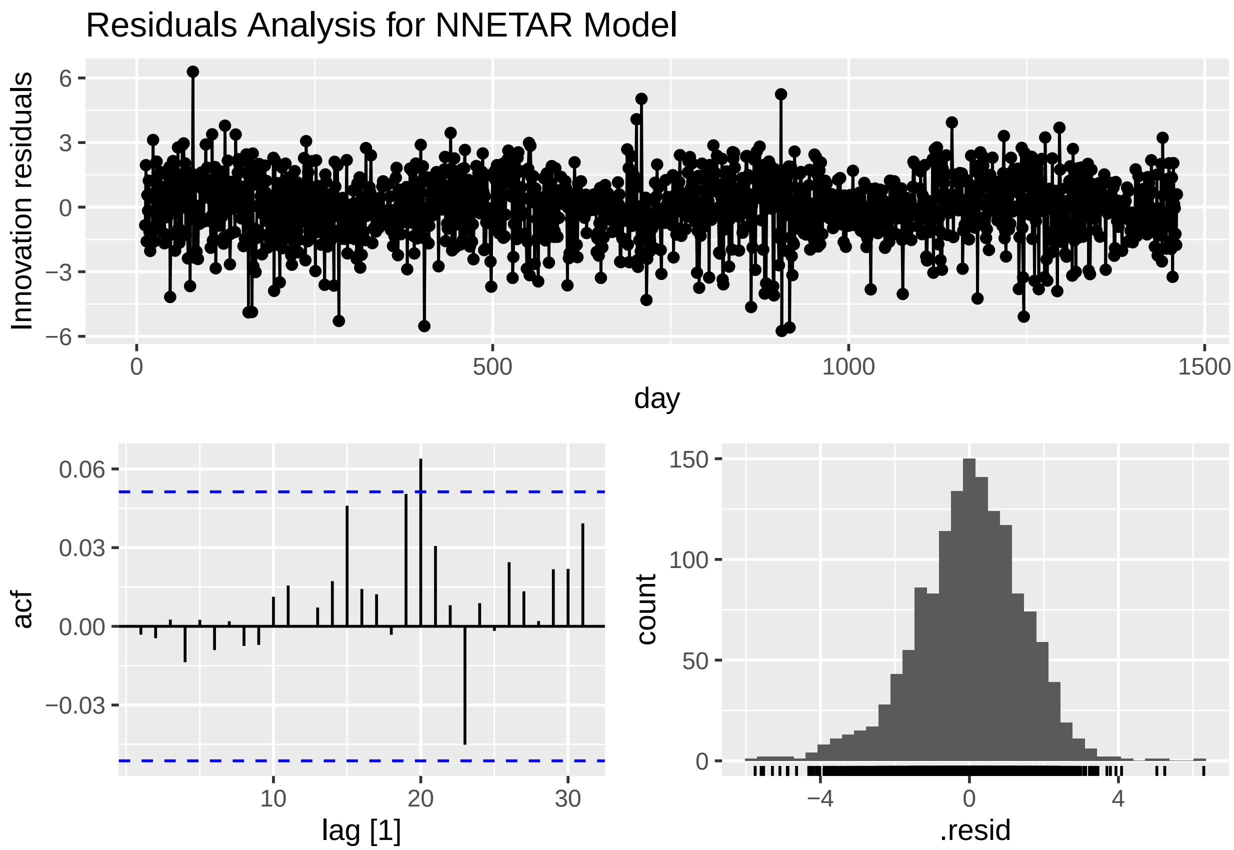
```
## # A tibble: 1 x 10
##   .model           .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE  ACF1
##   <chr>            <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 NNETAR(meantemp) Test   3.77  6.33  4.71  12.0  18.8  3.81  3.80 0.932
```

```
NNAR_fit |> gg_tsresiduals() + labs(title = "Residuals Analysis for NNETAR Model")
```

```
## Warning: Removed 11 rows containing missing values (`geom_line()`).
```

```
## Warning: Removed 11 rows containing missing values (`geom_point()`).
```

```
## Warning: Removed 11 rows containing non-finite values (`stat_bin()`).
```



Prophet model

```r
a <- train_data$meantemp
train <- as.data.frame(a)
train <- cbind(ds = train_data$date, train)
rownames(train) <- 1:nrow(train)
colnames(train) <- c ("ds", "y")
head(train)
```

```
##           ds         y
## 1 2013-01-01 10.000000
## 2 2013-01-02  7.400000
## 3 2013-01-03  7.166667
## 4 2013-01-04  8.666667
## 5 2013-01-05  6.000000
## 6 2013-01-06  7.000000
```

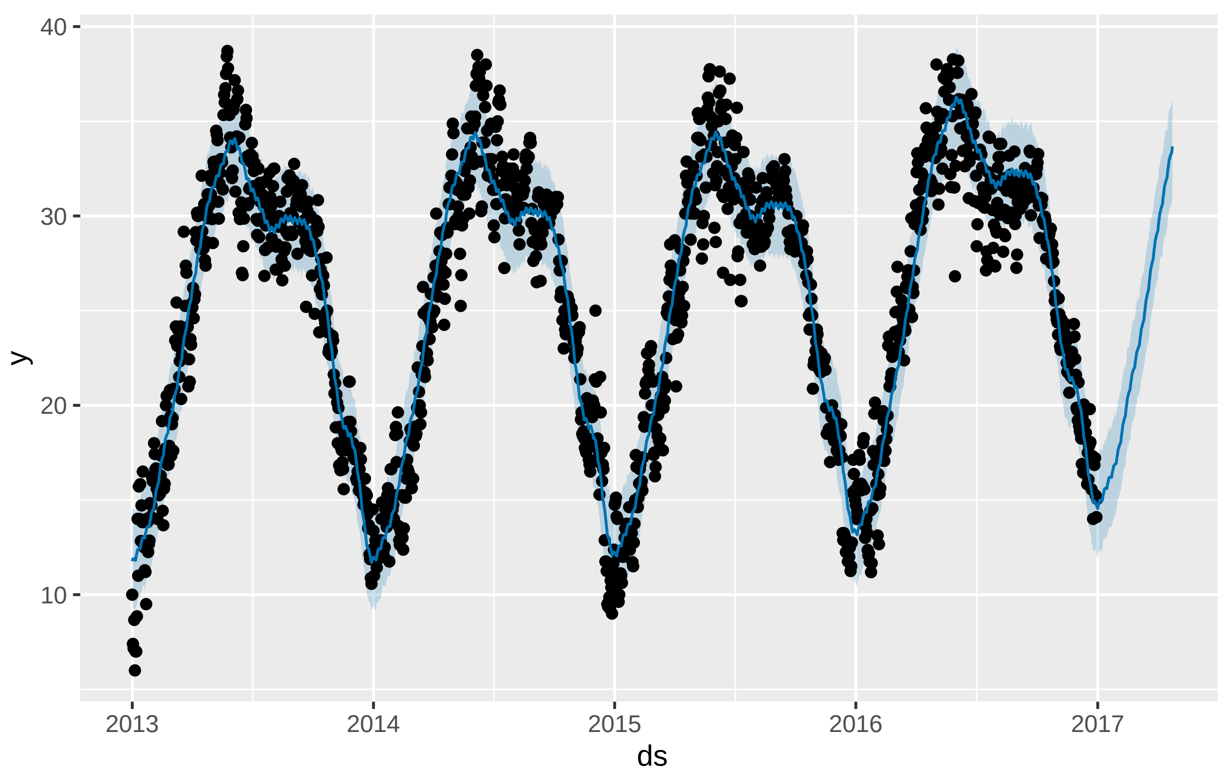
```r
# fit the prophet model
fit.prophet <- prophet(train)
```

```
## Disabling daily seasonality. Run prophet with daily.seasonality=TRUE to override this.
```

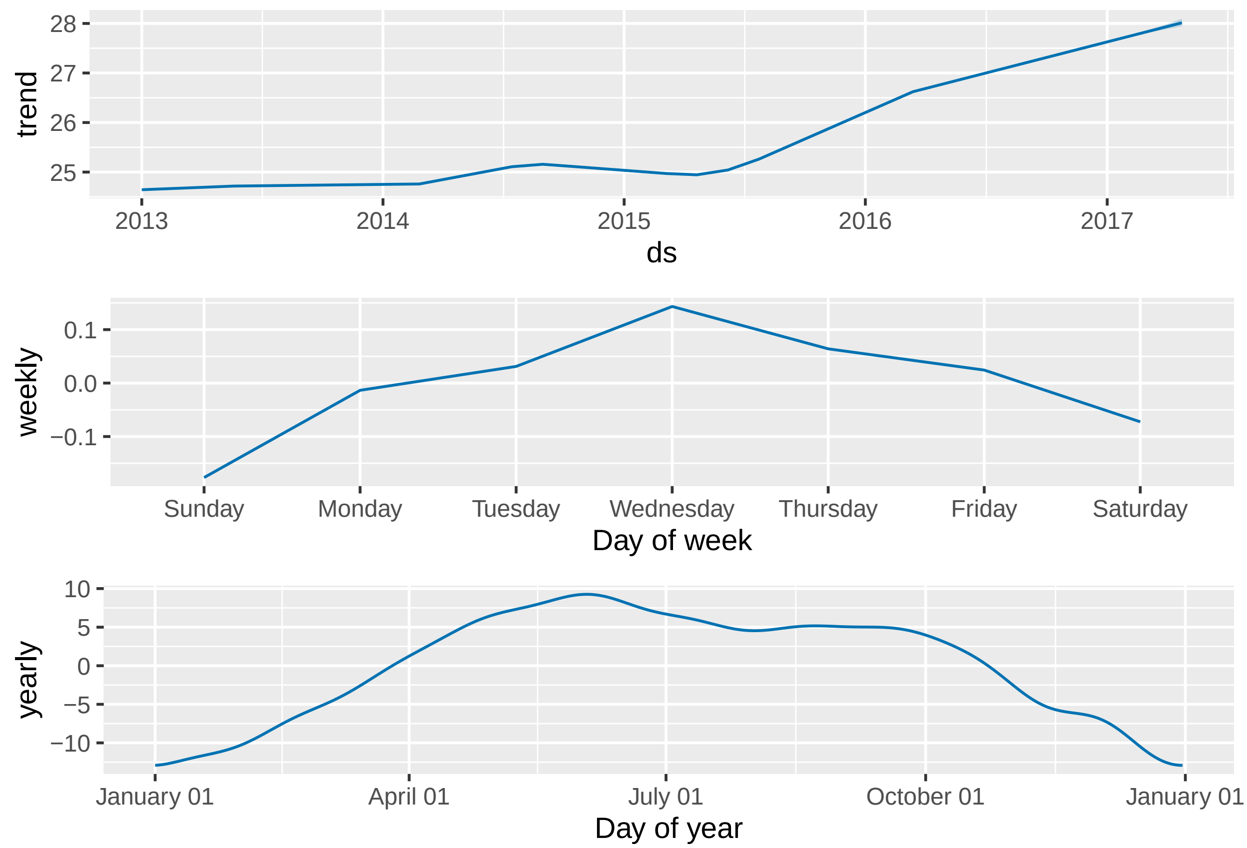
```r
future <- data.frame(ds = full_data_tsibble$date)
tail(future)
```

```
##              ds
## 1570 2017-04-19
## 1571 2017-04-20
## 1572 2017-04-21
## 1573 2017-04-22
## 1574 2017-04-23
## 1575 2017-04-24
```

```r
fit.prophet_fc <- predict(fit.prophet, future)

plot(fit.prophet, fit.prophet_fc)
```




```r
## prophet decomposition
prophet_plot_components(fit.prophet, fit.prophet_fc)
```




```
# accuracy
accuracy_fit_prophet <- forecast::accuracy(fit.prophet_fc$yhat, test_data$meantemp)
accuracy_fit_prophet
```

```
##                ME     RMSE      MAE      MPE     MAPE
## Test set 1.991468 3.171376 2.557689 8.575263 11.87496
```

result

```
## results
result_1 <- data.frame(
  models = c("Three Benchmark models","ARIMA","EWMA","Standard Regression Model", "Dynamic Regression Mo
  RMSE = c(7.38, arima.acc[2], ewma.acc[2], forecast_regression_model.acc[2],forecast_dynamic_reg_model
          accuracy_fit_prophet[2] )
)
kable(result_1)
```

models	RMSE
Three Benchmark models	7.380000
ARIMA	12.248828
EWMA	9.288431
Standard Regression Model	2.852384

models	RMSE
Dynamic Regression Model	3.873715
Combination regression	3.873715
NNETAR Model	6.330071
Prophet Model	3.171376