Final Project

Farooq Mahmud

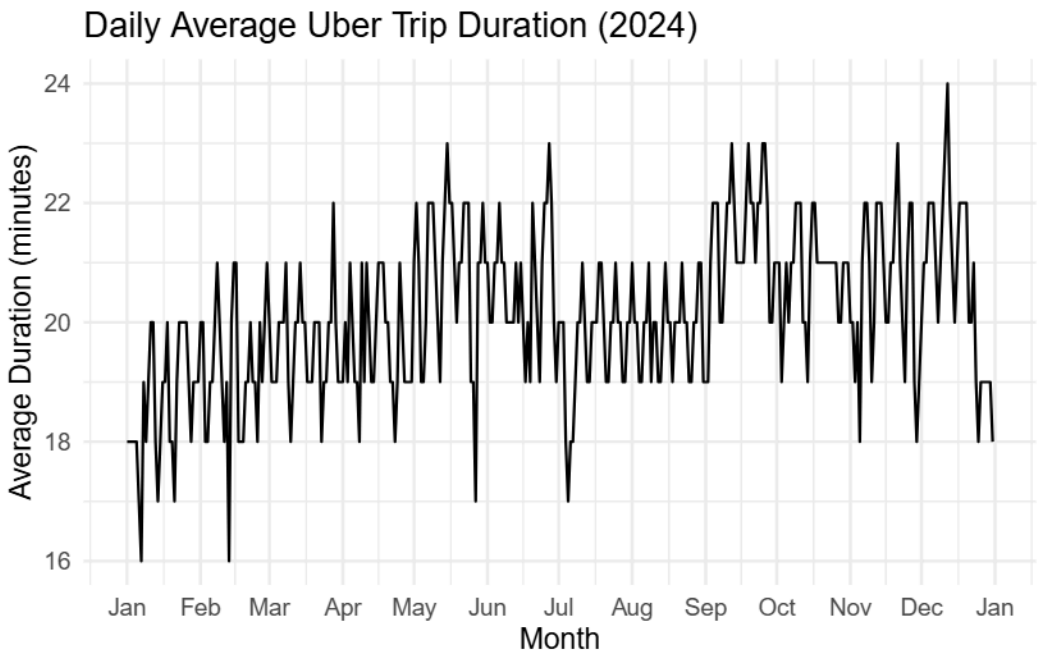
Load time series

```
data <- read.csv(file.path(getwd(), "../data", "finalproject.csv"))
ts_data <- ts(data$avg_duration_min, frequency = 366, start = c(2024, 1))
```

Plot time series

```
library(ggplot2)
library(scales)

start_date <- as.Date("2024-01-01")
date_seq <- seq(from = start_date, by = "day", length.out = length(ts_data))

df <- data.frame(
  Date = date_seq,
  Duration = as.numeric(ts_data)
)

ggplot(df, aes(x = Date, y = Duration)) +
  geom_line() +
  labs(
    title = "Daily Average Uber Trip Duration (2024)",
    x = "Month",
    y = "Average Duration (minutes)"
  ) +
  scale_x_date(
    date_breaks = "1 month",
    date_labels = "%b"
```

```
  ) +
  theme_minimal()
```



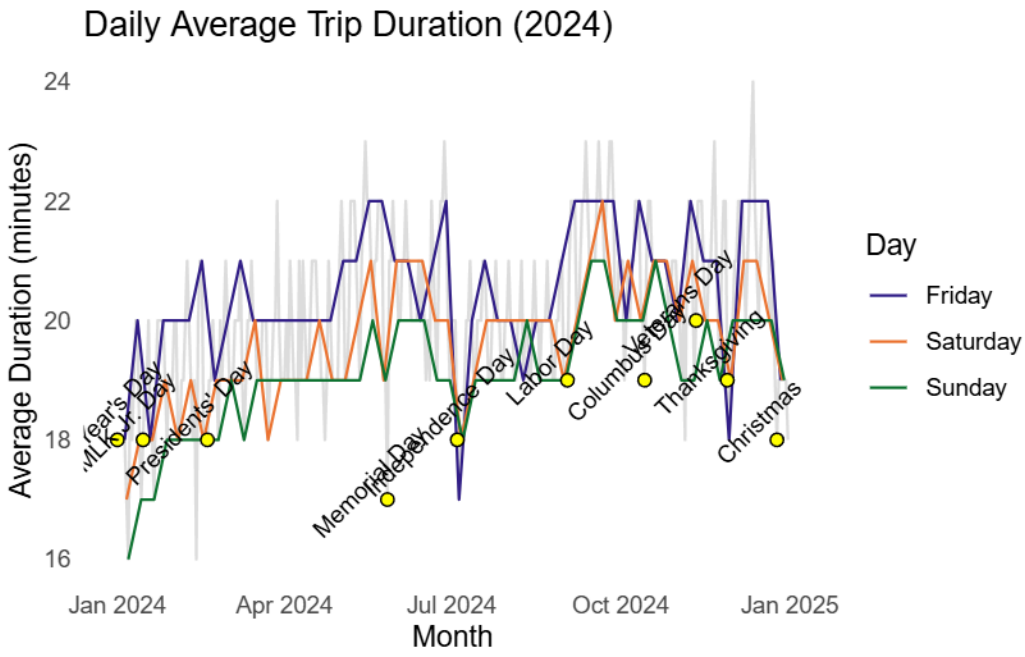
Plot Highlighting Weekends and Holidays

```
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
data$pickup_date <- as.Date(data$pickup_date)

data <- data %>%
  mutate(weekday = weekdays(pickup_date))

fridays <- filter(data, weekday == "Friday")
saturdays <- filter(data, weekday == "Saturday")
sundays <- filter(data, weekday == "Sunday")

holidays <- data.frame(
  date = as.Date(c(
    "2024-01-01",  # New Year's Day
    "2024-01-15",  # Martin Luther King Jr. Day
    "2024-02-19",  # Presidents' Day
    "2024-05-27",  # Memorial Day
    "2024-07-04",  # Independence Day
    "2024-09-02",  # Labor Day
    "2024-10-14",  # Columbus Day
    "2024-11-11",  # Veterans Day
    "2024-11-28",  # Thanksgiving
    "2024-12-25"   # Christmas
  )),
  label = c(
    "New Year's Day", "MLK Jr. Day", "Presidents' Day", "Memorial Day", "Independence Day",
    "Labor Day", "Columbus Day", "Veterans Day", "Thanksgiving", "Christmas"
  )
)

holidays$avg_duration_min <- data$avg_duration_min[match(holidays$date, data$pickup_date)]

ggplot(data, aes(x = pickup_date, y = avg_duration_min)) +
  geom_line(color = "gray", alpha = 0.5) +
  geom_line(data = fridays, aes(color = "Friday")) +
  geom_line(data = saturdays, aes(color = "Saturday")) +
  geom_line(data = sundays, aes(color = "Sunday")) +
  geom_point(data = holidays, aes(x = date, y = avg_duration_min),
             color = "black", fill = "yellow", size = 2, shape = 21) +
  geom_text(data = holidays, aes(x = date, y = avg_duration_min, label = label),
            vjust = -1, size = 3, angle = 45) +
  scale_color_manual(
    values = c(
      "Friday" = "#332288",
```

```
      "Saturday" = "#EE7733",
      "Sunday" = "#117733"
    )
  ) +
  labs(
    title = "Daily Average Trip Duration (2024)",
    x = "Month", y = "Average Duration (minutes)",
    color = "Day"
  ) +
  theme_minimal() +
  theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank()
  )
```



Stationarity Tests

```
library(tseries)
```

```
Registered S3 method overwritten by 'quantmod':
```

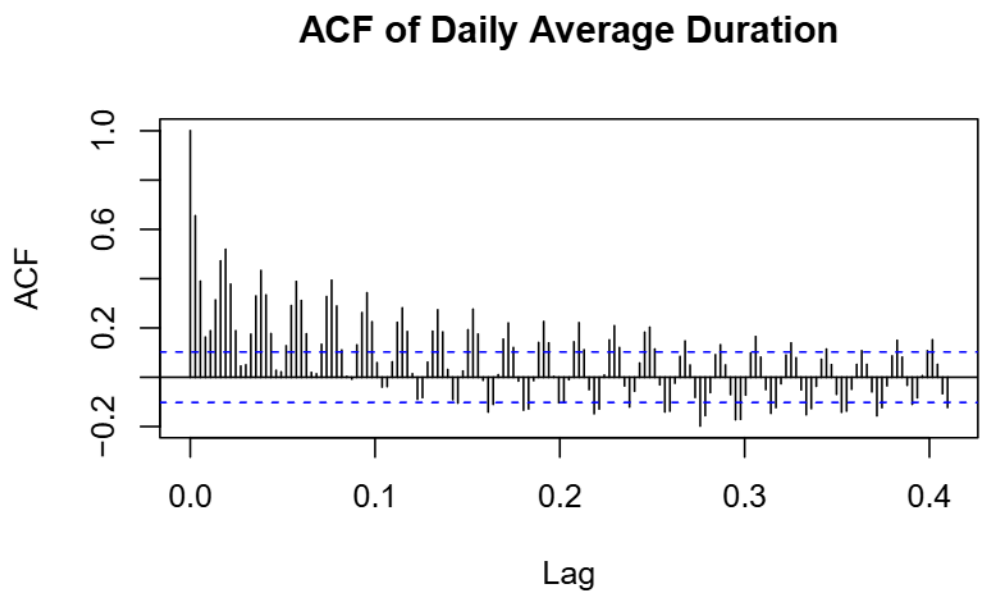
```
  method            from
  as.zoo.data.frame zoo
```

```
adf.test(ts_data)
```

```
    Augmented Dickey-Fuller Test

data:  ts_data
Dickey-Fuller = -3.4658, Lag order = 7, p-value = 0.04612
alternative hypothesis: stationary
```

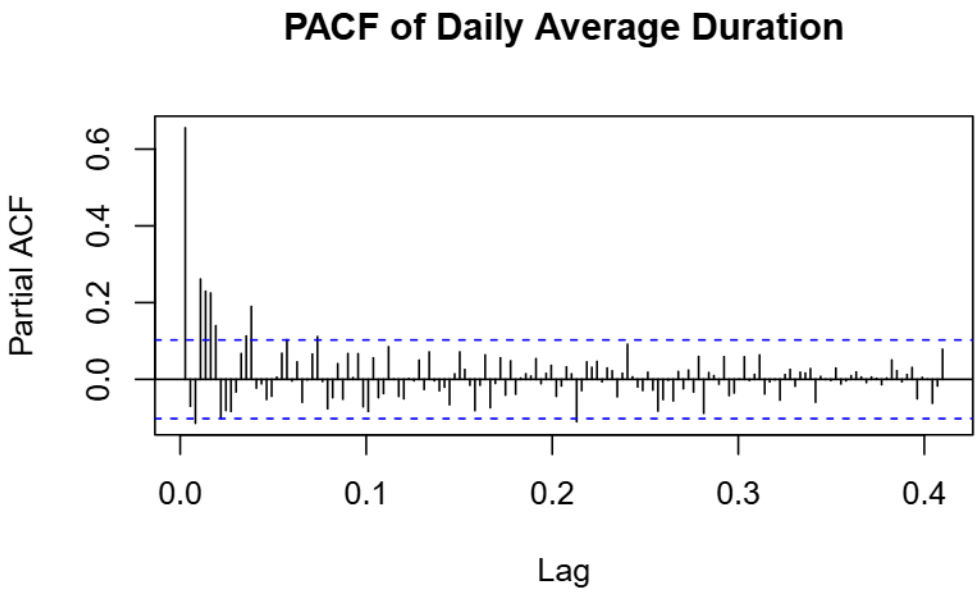
ACF Plot

```
acf(ts_data, main = "ACF of Daily Average Duration", lag.max = 150)
```



PACF Plot

```r
pacf(ts_data, main = "PACF of Daily Average Duration", lag.max = 150)
```



Model Selection and Diagnostics

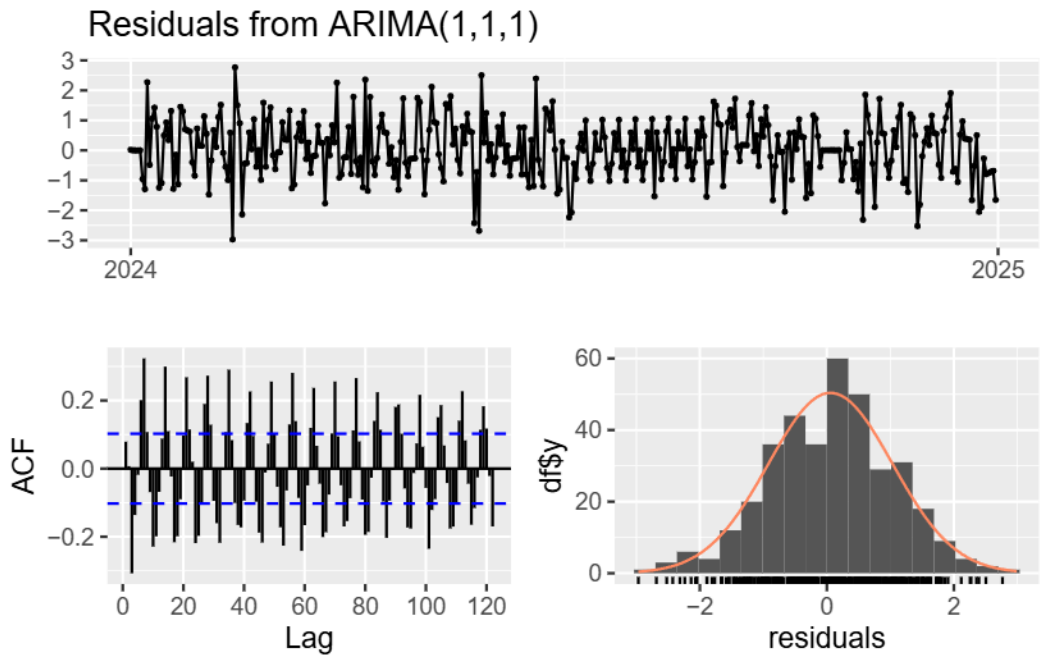
```r
library(forecast)
diff_data <- diff(ts_data)

fit_1 <- Arima(ts_data, order = c(1,1,1))
fit_2 <- Arima(ts_data, order = c(2,1,1))
auto_fit <- auto.arima(ts_data)

model_comparison <- data.frame(
  Model = c("ARIMA(1,1,1)", "ARIMA(2,1,1)", "ARIMA(3,1,2)"),
  AIC = c(AIC(fit_1), AIC(fit_2), AIC(auto_fit)),
  BIC = c(BIC(fit_1), BIC(fit_2), BIC(auto_fit))
)
print(model_comparison)
```

```
          Model       AIC       BIC
```

```
1 ARIMA(1,1,1) 1028.0164 1039.7161
2 ARIMA(2,1,1) 1019.5954 1035.1950
3 ARIMA(3,1,2)  962.0807  985.4801
```

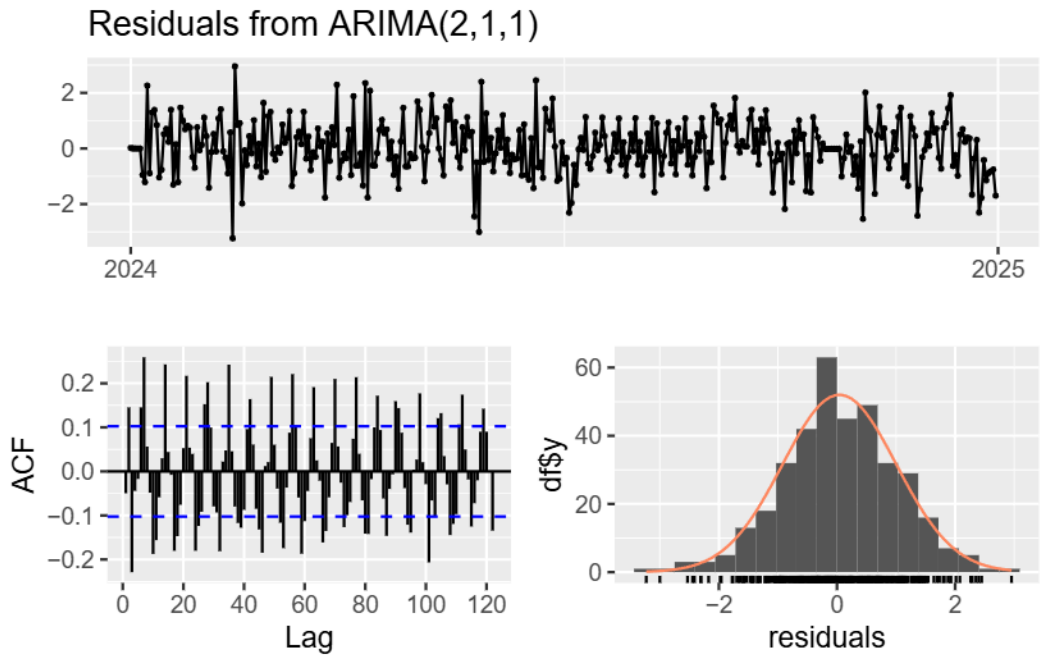
Check Residuals

```
checkresiduals(fit_1)
```



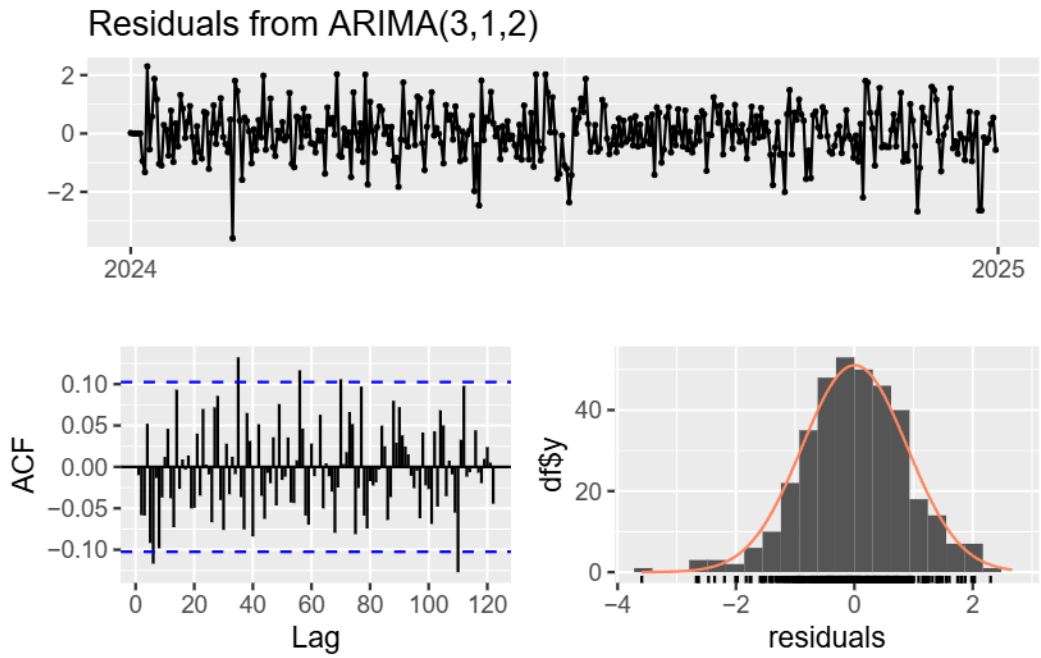

```
    Ljung-Box test

data:  Residuals from ARIMA(1,1,1)
Q* = 810.27, df = 71, p-value < 2.2e-16

Model df: 2.   Total lags used: 73
```

```
checkresiduals(fit_2)
```




```
    Ljung-Box test

data:  Residuals from ARIMA(2,1,1)
Q* = 477.14, df = 70, p-value < 2.2e-16

Model df: 3.   Total lags used: 73
```

```
checkresiduals(auto_fit)
```




```
    Ljung-Box test

data:  Residuals from ARIMA(3,1,2)
Q* = 88.007, df = 68, p-value = 0.05184

Model df: 5.   Total lags used: 73
```

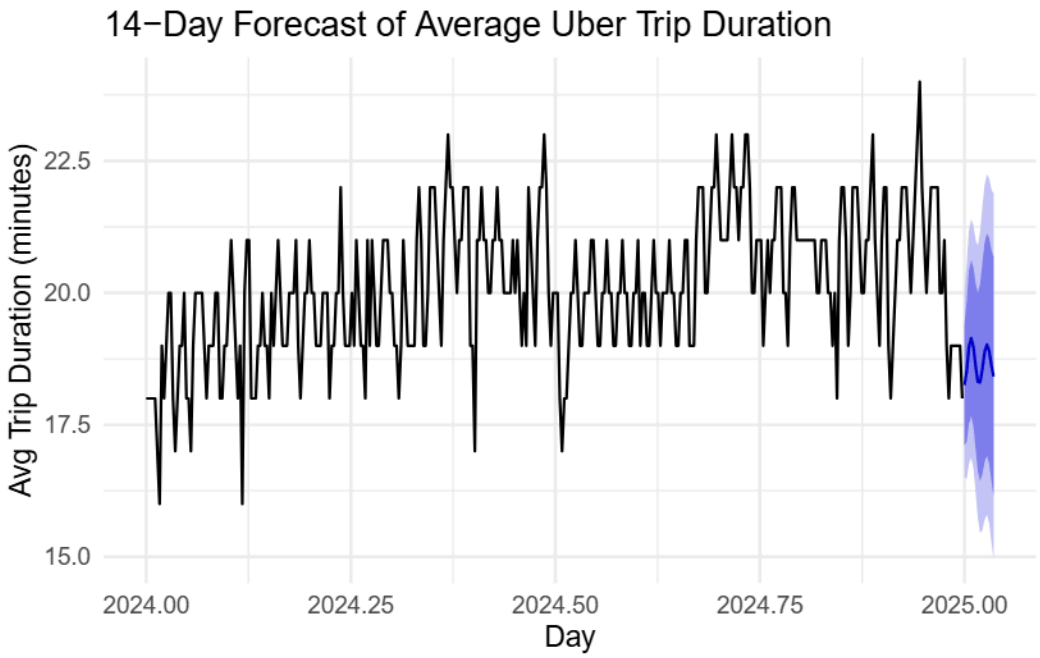
14-Day Forecast

```r
h <- 14
forecast_result <- forecast(auto_fit, h = h)
```

14-Day Forecast Plot

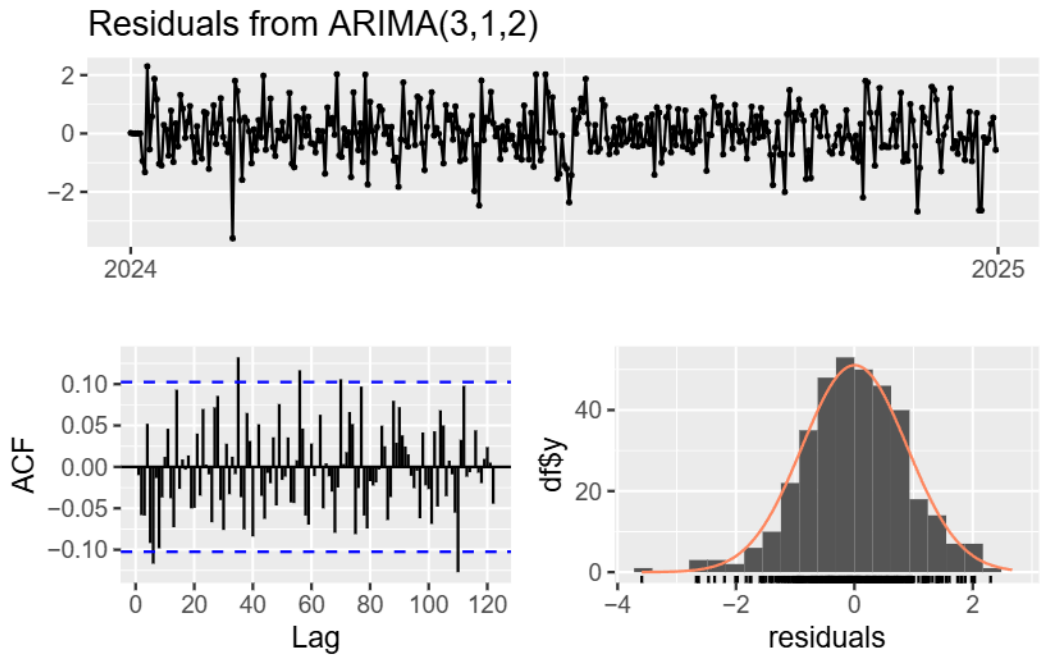
```r
autoplot(forecast_result) +
  labs(
    title = "14-Day Forecast of Average Uber Trip Duration",
    x = "Day",
```

```
    y = "Avg Trip Duration (minutes)"
  ) +
  theme_minimal()
```



Forecast Residuals

```
checkresiduals(forecast_result)
```




```
    Ljung-Box test

data:  Residuals from ARIMA(3,1,2)
Q* = 88.007, df = 68, p-value = 0.05184

Model df: 5.   Total lags used: 73
```

Table of Forecasted Values

```r
start_date <- as.Date("2025-01-01")
forecast_dates <- seq.Date(from = start_date, by = "day", length.out = h)
forecast_summary <- as.data.frame(forecast_result)
formatted_forecast <- cbind(Date = forecast_dates, forecast_summary)

formatted_forecast <- within(formatted_forecast, {
  `Point Forecast` <- round(`Point Forecast`, 2)
  `Lo 80` <- round(`Lo 80`, 2)
  `Hi 80` <- round(`Hi 80`, 2)
  `Lo 95` <- round(`Lo 95`, 2)
  `Hi 95` <- round(`Hi 95`, 2)
```

```
})

print(formatted_forecast)
```

```
               Date Point Forecast Lo 80 Hi 80 Lo 95 Hi 95
2025.0000 2025-01-01          18.25 17.11 19.40 16.50 20.00
2025.0027 2025-01-02          18.52 17.19 19.85 16.48 20.55
2025.0055 2025-01-03          18.98 17.53 20.42 16.76 21.19
2025.0082 2025-01-04          19.14 17.67 20.62 16.89 21.40
2025.0109 2025-01-05          18.98 17.48 20.48 16.68 21.27
2025.0137 2025-01-06          18.60 17.04 20.16 16.22 20.99
2025.0164 2025-01-07          18.31 16.62 20.00 15.73 20.90
2025.0191 2025-01-08          18.30 16.44 20.17 15.46 21.15
2025.0219 2025-01-09          18.56 16.56 20.56 15.50 21.62
2025.0246 2025-01-10          18.88 16.80 20.95 15.71 22.04
2025.0273 2025-01-11          19.02 16.92 21.13 15.80 22.24
2025.0301 2025-01-12          18.91 16.78 21.04 15.65 22.17
2025.0328 2025-01-13          18.64 16.46 20.81 15.31 21.97
2025.0355 2025-01-14          18.42 16.15 20.68 14.95 21.88
```