TimeExploration

Yesh Onipede

2024-04-17

Setting up data paths

```
merged_data <- read.csv("Gin_LiquorSales_Education.csv")
```

Loading packages

```
library(forecast)
```

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

```
library(ggplot2)
library(tidyr)
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
```

```
library(cowplot)
library(stats)
```

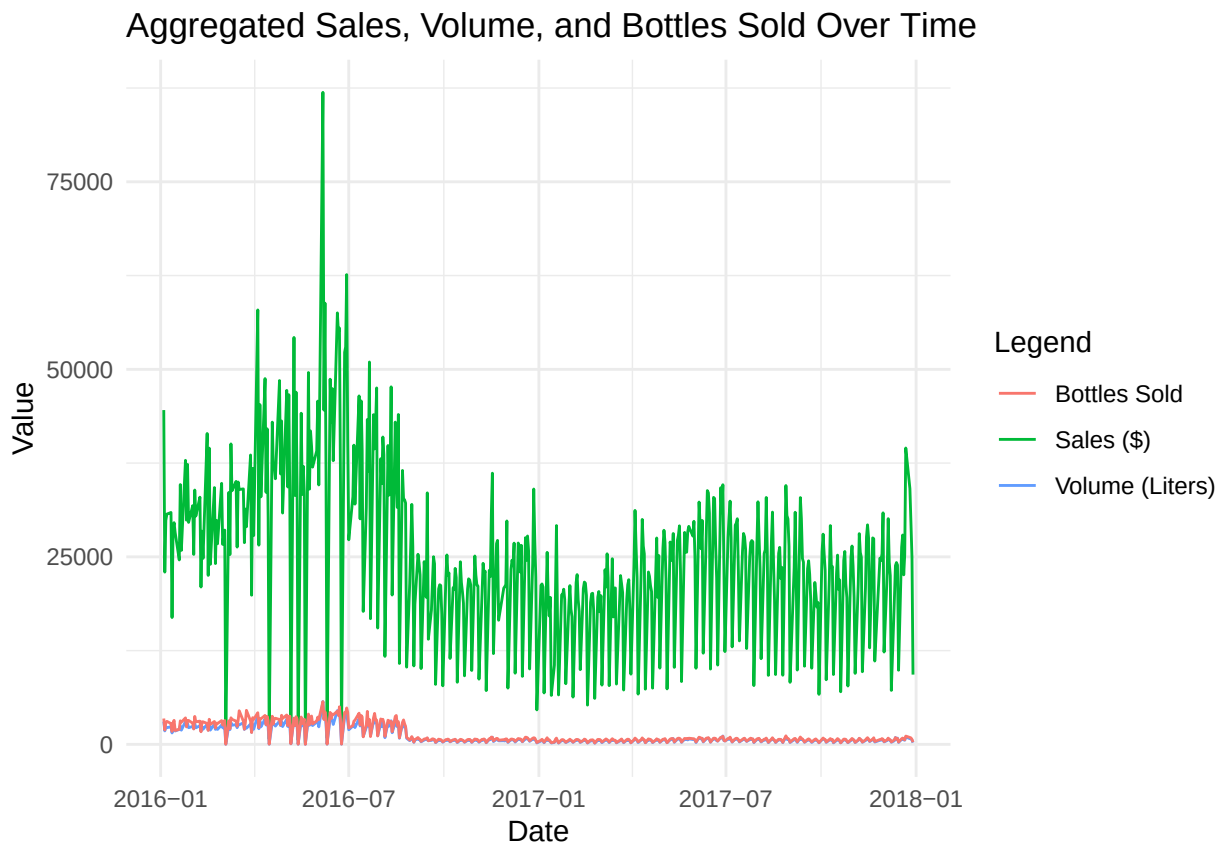
Creating an aggregated table and plotting

```
# Convert the date column to Date format
merged_data$date <- as.Date(merged_data$date)

# Group the data by 'date and summarize to calculate the sum of each variable for each date
aggregated_data <- merged_data %>%
  group_by(date) %>%
```

```r
  summarize(total_sale_dollars = sum(sale.dollars),
            total_sale_volume = sum(sale.volume),
            total_sale_bottles = sum(sale.bottles))

# Print the aggregated data frame (merged data grouped)
print(aggregated_data)
```

```
## # A tibble: 492 x 4
##    date       total_sale_dollars total_sale_volume total_sale_bottles
##    <date>                  <dbl>             <dbl>               <int>
##  1 2016-01-04             44565.             3035.                3446
##  2 2016-01-05             22968.             1797.                1923
##  3 2016-01-06             29866.             2385.                2877
##  4 2016-01-07             30697.             2215.                3072
##  5 2016-01-11             30916.             2359.                2772
##  6 2016-01-12             16911.             1520.                1730
##  7 2016-01-13             28690.             2193.                2869
##  8 2016-01-14             29557.             2105.                3202
##  9 2016-01-15             28258.             1782.                1841
## 10 2016-01-19             24588.             1999.                2008
## # i 482 more rows
```

```r
# Plotting
ggplot(aggregated_data, aes(x = date)) +
  geom_line(aes(y = total_sale_dollars, color = "Sales ($)")) +
  geom_line(aes(y = total_sale_volume, color = "Volume (Liters)")) +
  geom_line(aes(y = total_sale_bottles, color = "Bottles Sold")) +
  labs(x = "Date", y = "Value", color = "Legend") +
  ggtitle("Aggregated Sales, Volume, and Bottles Sold Over Time") +
  theme_minimal()
```




```
# Convert the date column to Date format
merged_data$date <- as.Date(merged_data$date)

# Extract year and month from the date column
merged_data$year <- lubridate::year(merged_data$date)
merged_data$month <- lubridate::month(merged_data$date)

# Group the data by year and month, and summarize to calculate the sum of each variable for each month
aggregated_data_month <- merged_data %>%
  group_by(year, month) %>%
  summarize(total_sale_dollars = sum(sale.dollars),
            total_sale_volume = sum(sale.volume),
            total_sale_bottles = sum(sale.bottles))
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
aggregated_data_month$date <- as.Date(paste(aggregated_data_month$year, aggregated_data_month$month, "0

# Print the aggregated data frame (merged data grouped by month and year)
print(aggregated_data_month)
```

```
## # A tibble: 24 x 6
## # Groups:   year [2]
```

```
##     year month total_sale_dollars total_sale_volume total_sale_bottles
##    <dbl> <dbl>              <dbl>             <dbl>              <int>
##  1  2016     1            482181.            36250.              44069
##  2  2016     2            505842.            39008.              47078
##  3  2016     3            590910.            44994.              58071
##  4  2016     4            628990.            44507.              53304
##  5  2016     5            721247.            50966.              59892
##  6  2016     6            896498.            64111.              73628
##  7  2016     7            679458.            47153.              55461
##  8  2016     8            730100.            48754.              56523
##  9  2016     9            436309.            11931.              13732
## 10  2016    10            414997.            10347.              11934
## # i 14 more rows
## # i 1 more variable: date <date>
```

I tried to decompse the data seasonally but it failed even though it satisfied the requirement of having two cyles. Therefore it is safe to say that the data does not have any clear seasonality.

```
# Perform seasonal decomposition
#decomposition <- stl(aggregated_data_month$total_sale_dollars, s.window = "periodic")

# Plot the decomposed components
#autoplot(decomposition)
```

Linear regression model for total sale in dollars

```
# Run linear regression
lm_model_dollars <- lm(total_sale_dollars ~ aggregated_data_month$date, data = aggregated_data_month)

# View summary of the regression model
summary(lm_model_dollars)
```

```
##
## Call:
## lm(formula = total_sale_dollars ~ aggregated_data_month$date,
##     data = aggregated_data_month)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -163765 -101728    4546   60888  322204
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 5131326.7  2011893.1   2.550   0.0182 *
## aggregated_data_month$date    -268.8      117.3  -2.292   0.0319 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 121100 on 22 degrees of freedom
## Multiple R-squared:  0.1927, Adjusted R-squared:  0.156
## F-statistic: 5.252 on 1 and 22 DF,  p-value: 0.03187
```

```r
# View summary of the regression model
summary(lm_model_dollars)
```

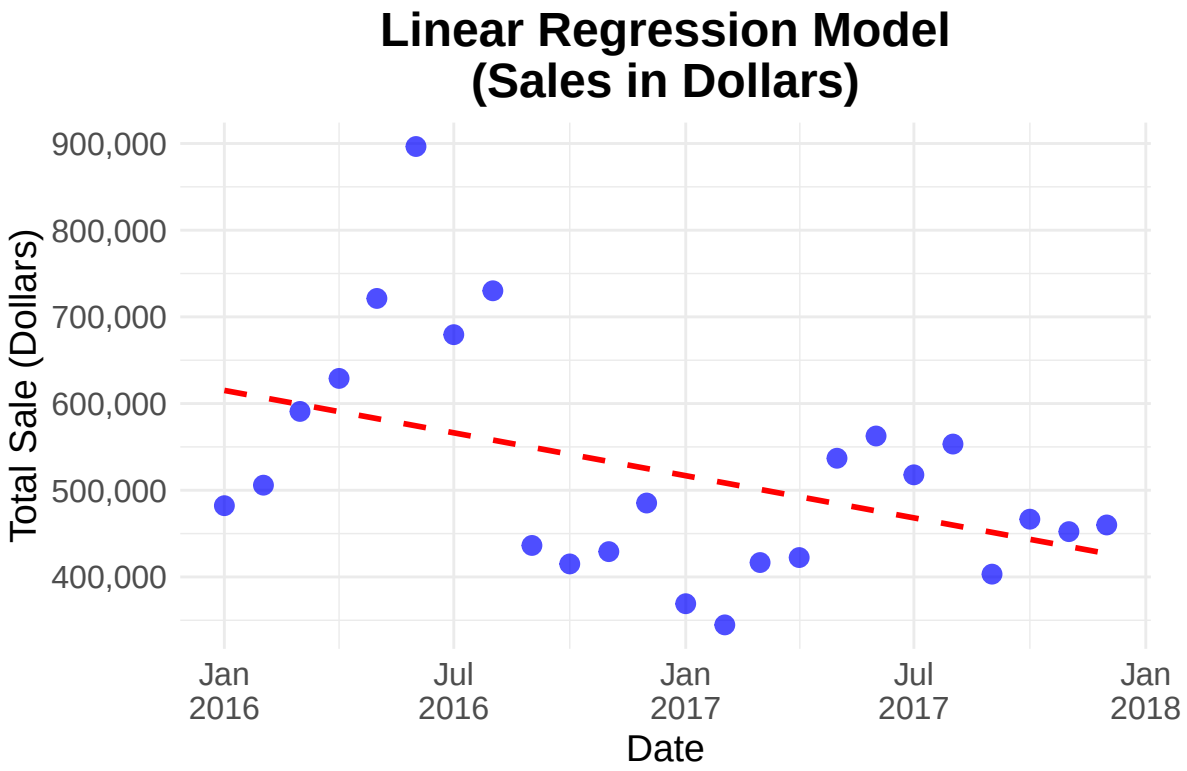
```
##
## Call:
## lm(formula = total_sale_dollars ~ aggregated_data_month$date,
##     data = aggregated_data_month)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -163765 -101728    4546   60888  322204
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 5131326.7  2011893.1   2.550   0.0182 *
## aggregated_data_month$date    -268.8      117.3  -2.292   0.0319 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 121100 on 22 degrees of freedom
## Multiple R-squared:  0.1927, Adjusted R-squared:  0.156
## F-statistic: 5.252 on 1 and 22 DF,  p-value: 0.03187
```

The coefficient date is statistically significant. Approximately 19.27% of variance is explained by the linear regression model.

Plotting linear regression for total sale in dollars

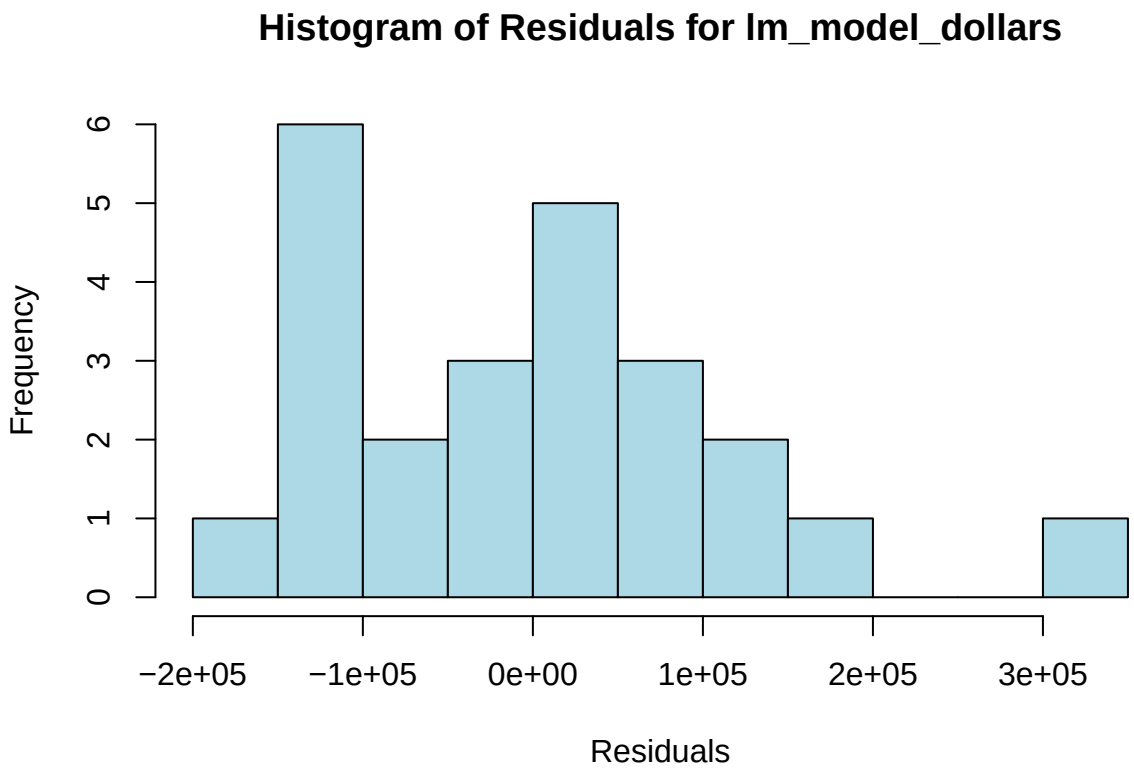
```r
ggplot(aggregated_data_month, aes(x = date, y = total_sale_dollars)) +
  geom_point(color = "blue", size = 3, alpha = 0.7) +
  geom_smooth(method = "lm", color = "red", linetype = "dashed", se = FALSE) +
  labs(x = "Date", y = "Total Sale (Dollars)", title = "Linear Regression Model\n(Sales in Dollars)") +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 18, face = "bold", hjust = 0.5),
    axis.title = element_text(size = 14),
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12),
    legend.position = "none",
    plot.margin = margin(20, 20, 20, 20)  # Adjust top, right, bottom, left margins
  ) +
  scale_x_date(date_labels = "%b\n%Y", limits = c(min(aggregated_data_month$date), max(aggregated_data_
  scale_y_continuous(labels = scales::comma)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

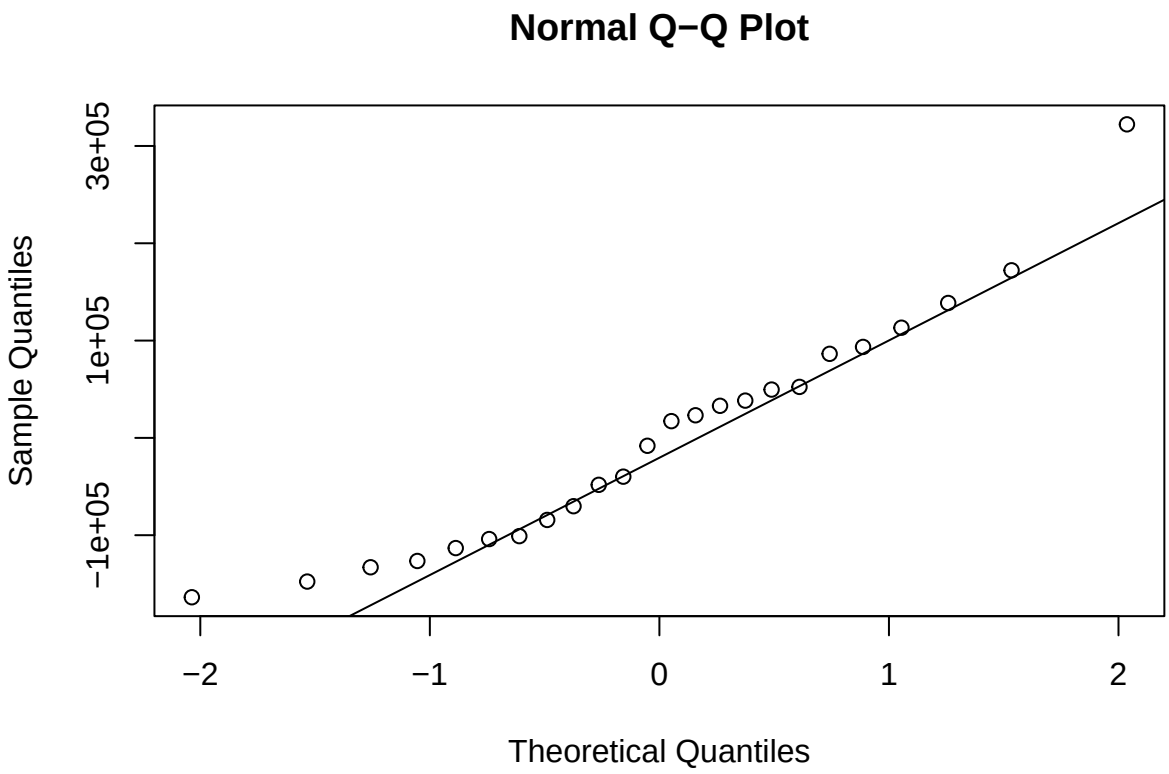


Plotting the residuals of total sales in dollars

```
residuals_dollars <- residuals(lm_model_dollars)

# Plot histogram of residuals
hist(residuals_dollars, breaks = 10, col = "lightblue", main = "Histogram of Residuals for lm_model_doll
```




```r
# Q-Q plot of residuals
qqnorm(residuals_dollars)
qqline(residuals_dollars)
```



The residuals are mostly normally distributed aside from the peak in the left tail. It is important that residuals follow a normal distribution as it is an assumption of linear regression.

Looking at the normal Q-Q plot, most of the points fall along the diagonal line which suggests little deviation from normality. This is a good sign for the linear model as it assume normality.

Linear regression model for total sale in volume

```
lm_model_vol <- lm(total_sale_volume ~ date, data = aggregated_data_month)

# View summary of the regression model
summary(lm_model_vol)
```

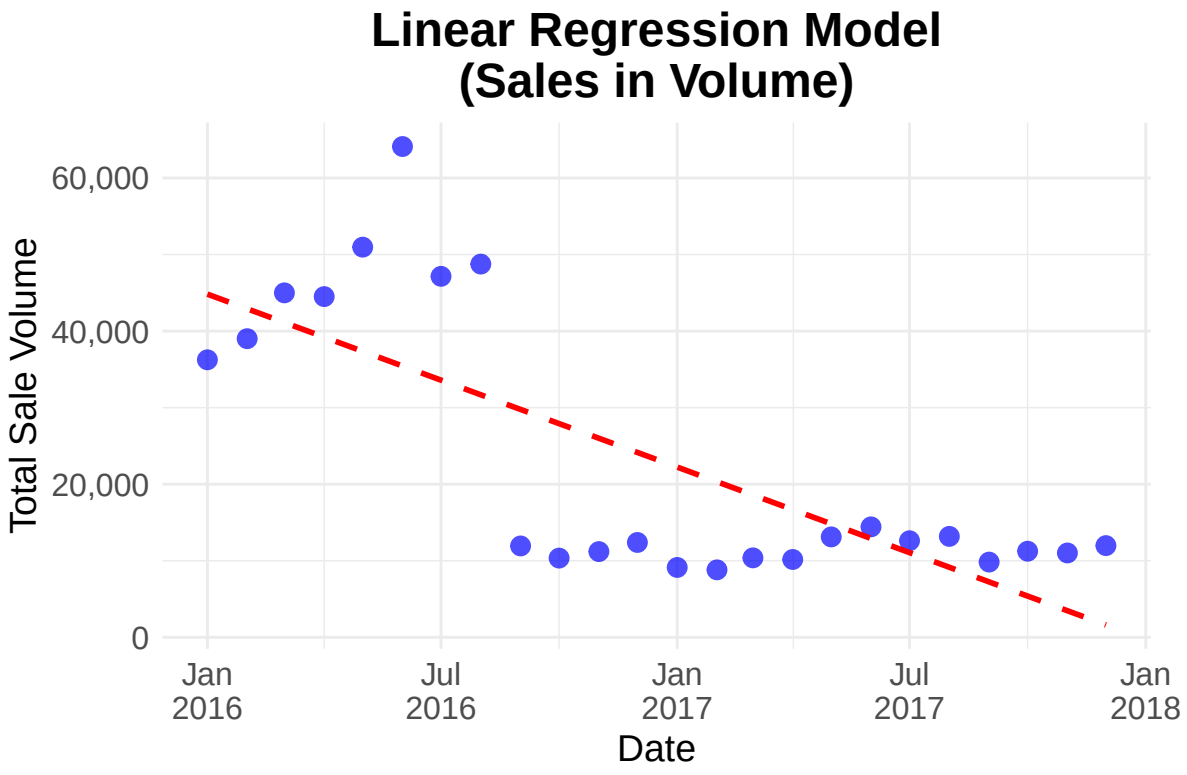
```
##
## Call:
## lm(formula = total_sale_volume ~ date, data = aggregated_data_month)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -17835  -9310   1530   6282  28667
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1081868.75  201726.13   5.363 2.20e-05 ***
## date            -61.73      11.76  -5.248 2.89e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 12140 on 22 degrees of freedom
## Multiple R-squared:  0.556,  Adjusted R-squared:  0.5358
## F-statistic: 27.54 on 1 and 22 DF,  p-value: 2.894e-05
```

Plotting linear regression for sales in volume

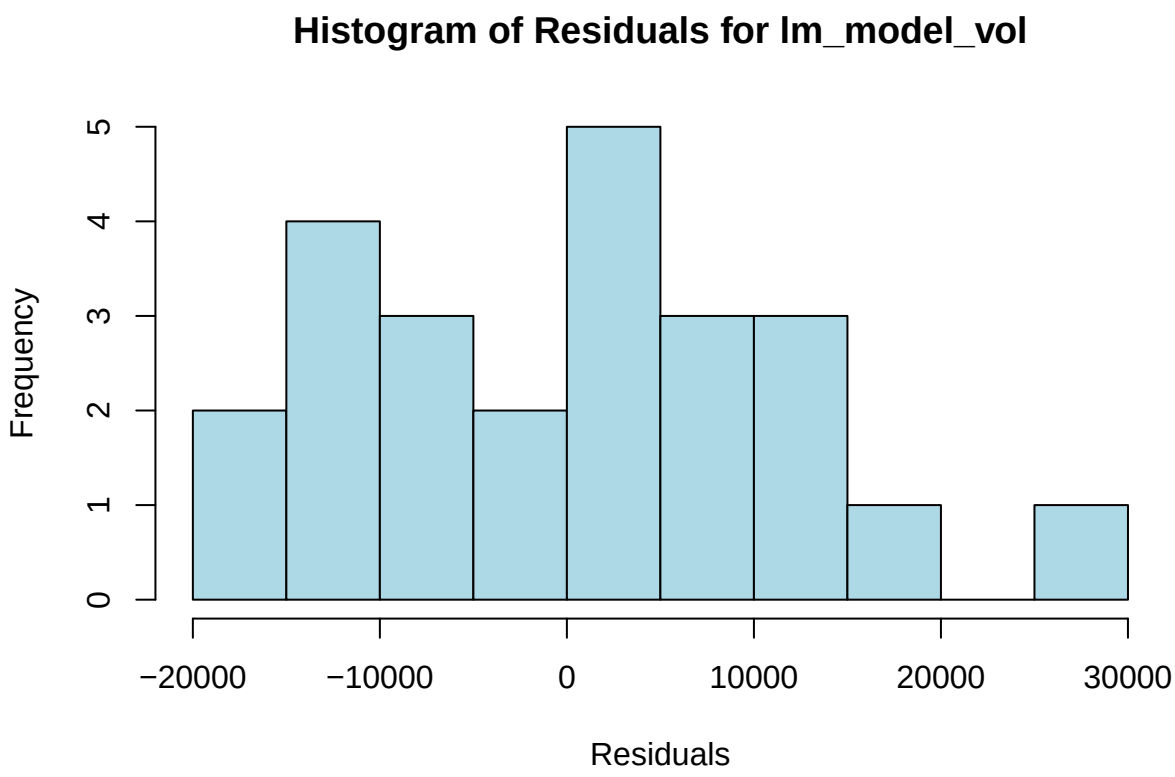
```r
# Plot scatterplot with regression line
ggplot(aggregated_data_month, aes(x = date, y = total_sale_volume)) +
  geom_point(color = "blue", size = 3, alpha = 0.7) +
  geom_smooth(method = "lm", color = "red", linetype = "dashed", se = FALSE) +
  labs(x = "Date", y = "Total Sale Volume", title = "Linear Regression Model\n(Sales in Volume)") +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 18, face = "bold", hjust = 0.5),
    axis.title = element_text(size = 14),
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12),
    legend.position = "none",
    plot.margin = margin(20, 20, 20, 20)  # Adjust top, right, bottom, left margins
  ) +
  scale_x_date(date_labels = "%b\n%Y", limits = c(min(aggregated_data_month$date), max(aggregated_data_
  scale_y_continuous(labels = scales::comma)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

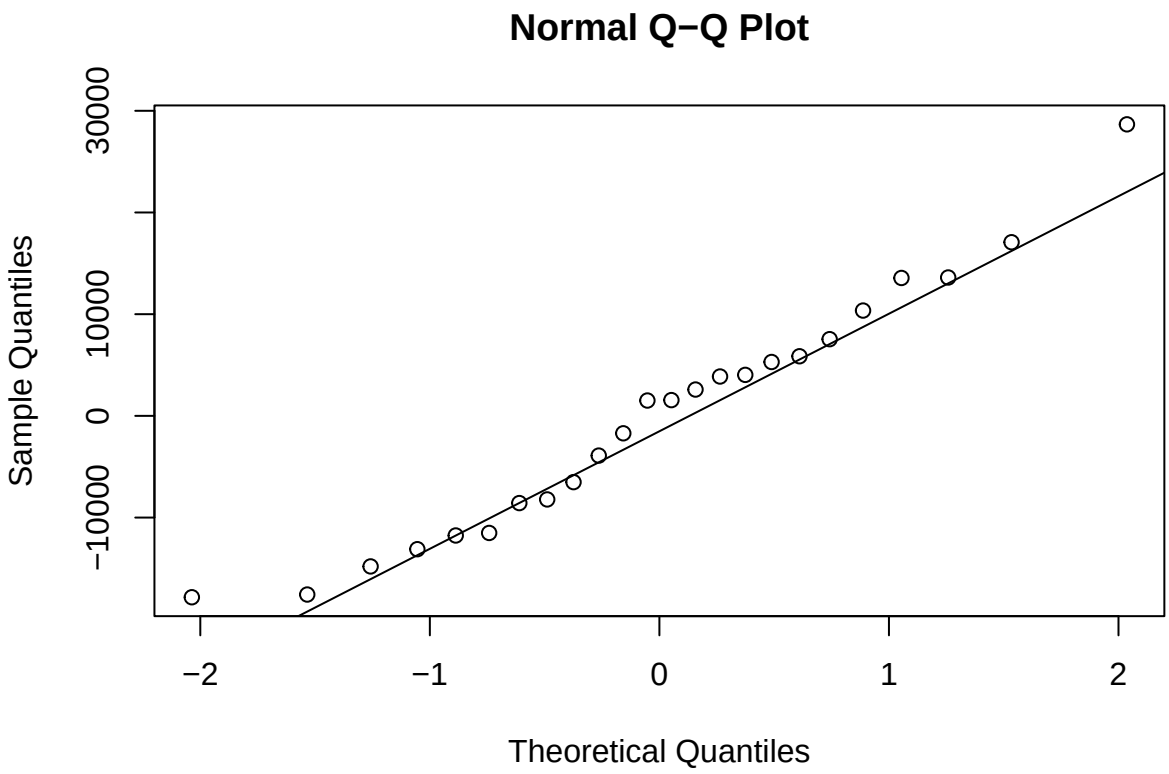


The date variable is significant at the 0.001 level. Approximately 55% of the variance is explained by the linear regression model.

```
residuals_vol <- residuals(lm_model_vol)

# Plot histogram of residuals
hist(residuals_vol, breaks = 10, col = "lightblue", main = "Histogram of Residuals for lm_model_vol", xl
```




```
# Q-Q plot of residuals
qqnorm(residuals_vol)
qqline(residuals_vol)
```



The histogram reveals that the residuals follow a pretty normal distribution with the peak centered in the middle and the bars mostly decreasing as they move toward the tails. The residual points stay very close to the diagonal line in the Normal Q-Q Plot which suggests that the criteria of normalacy is met. This is a good sign for the validation of the linear regression model.

Here I tested out a polynmoial regression models for sale in dollars but it offered no significance in the polynomial term so I will stick to the linear regression model.

```
# Fit a polynomial regression model
poly_degree <- 2  # Set the degree of the polynomial
poly_formula <- as.formula(paste("total_sale_dollars ~ poly(date, ", poly_degree, ")", sep = ""))  # po

poly_lm_model_dollars <- lm(poly_formula, data = aggregated_data_month)

# View summary of the polynomial regression model
summary(poly_lm_model_dollars)
```

```
##
## Call:
## lm(formula = poly_formula, data = aggregated_data_month)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -158697  -93087  -11566   68424  323931
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)      521093      25133  20.733 1.82e-15 ***
## poly(date, 2)1  -277448     123127  -2.253   0.0351 *
## poly(date, 2)2    64026     123127   0.520   0.6085
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 123100 on 21 degrees of freedom
## Multiple R-squared:  0.203,  Adjusted R-squared:  0.1271
## F-statistic: 2.674 on 2 and 21 DF,  p-value: 0.09236
```

Here I tested out a polynomial regression but it revealed high significance at the linear term and hardly any significance at the quadratic term which to me suggests that a linear regression model is best for the volume data.

```
# Fit a polynomial regression model
poly_degree <- 2  # Set the degree of the polynomial
poly_formula <- as.formula(paste("total_sale_volume ~ poly(date, ", poly_degree, ")", sep = "")) # poly

poly_lm_model_vol <- lm(poly_formula, data = aggregated_data_month)

# View summary of the polynomial regression model
summary(poly_lm_model_vol)
```

```
##
## Call:
## lm(formula = poly_formula, data = aggregated_data_month)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17423.2  -7091.5    481.7   2839.8  29260.5
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)       23228       2339   9.931 2.19e-09 ***
## poly(date, 2)1   -63710      11458  -5.560 1.62e-05 ***
## poly(date, 2)2    22018      11458   1.922   0.0683 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11460 on 21 degrees of freedom
## Multiple R-squared:  0.6224, Adjusted R-squared:  0.5864
## F-statistic:  17.3 on 2 and 21 DF,  p-value: 3.625e-05
```