DS116 Data Visualization - Practical Session 1

Analyzing Gold Price Trends: A Time Series Case Study

American University of Armenia

Contents

Introduction

In this practical session, you will analyze **10 years of gold price data** (2016-2026) to practice the skills learned from:

- **Introduction to R**: Data loading, exploration, cleaning, and manipulation
- **Introduction to ggplot2**: Creating informative visualizations
- **Single Numeric Variables**: Understanding distributions of financial data

Learning Objectives

By the end of this session, you will be able to:

1. Load and explore real-world financial data
2. Handle missing values in time series data
3. Calculate and interpret financial metrics (returns, volatility)
4. Create visualizations for price distributions and trends
5. Compare distributions across time periods
6. Identify patterns and outliers in financial data

About the Dataset

The dataset contains **daily gold price records** from 2016 to 2026, including:

Column	Description
Date	Trading date
Open, High, Low, Close	Daily price range
Adj Close	Adjusted closing price
Volume	Trading volume
Daily_Return	Daily percentage change
MA_20, MA_50, MA_200	Moving averages
Volatility_20	20-day rolling volatility
Year, Month, Day_of_Week, Quarter	Time features

Step 1: Setup and Data Loading

1.1 Load Required Libraries

```r
library(dplyr)
library(tidyr)
library(ggplot2)
library(scales)  # For formatting axes
```

1.2 Load the Gold Prices Data

```r
# Load the dataset
gold <- read.csv("gold_prices_10y.csv")

# Check the first few rows
head(gold)
```

```
##         Date  Close   High    Low   Open   Volume Adj.Close Daily_Return MA_20
## 1 2016-01-29 106.95 107.00 106.26 106.61  8098700    106.95           NA    NA
## 2 2016-02-01 108.05 108.15 107.53 107.54 10471800    108.05   1.02852374    NA
## 3 2016-02-02 108.09 108.18 107.35 107.92  6656000    108.09   0.03701368    NA
## 4 2016-02-03 109.25 109.58 107.90 107.91 15785200    109.25   1.07318318    NA
## 5 2016-02-04 110.57 110.70 109.92 110.45 13213700    110.57   1.20823771    NA
## 6 2016-02-05 112.32 112.35 109.58 109.79 14777300    112.32   1.58270779    NA
##   MA_50 MA_200 Volatility_20 Year Month Day_of_Week Quarter
## 1    NA     NA            NA 2016     1           4       1
## 2    NA     NA            NA 2016     2           0       1
## 3    NA     NA            NA 2016     2           1       1
## 4    NA     NA            NA 2016     2           2       1
## 5    NA     NA            NA 2016     2           3       1
## 6    NA     NA            NA 2016     2           4       1
```

1.3 Basic Data Exploration

```r
# Structure of the data
str(gold)
```

```
## 'data.frame':    2511 obs. of  16 variables:
##  $ Date         : chr  "2016-01-29" "2016-02-01" "2016-02-02" "2016-02-03" ...
##  $ Close        : num  107 108 108 109 111 ...
##  $ High         : num  107 108 108 110 111 ...
##  $ Low          : num  106 108 107 108 110 ...
##  $ Open         : num  107 108 108 108 110 ...
##  $ Volume       : int  8098700 10471800 6656000 15785200 13213700 14777300 28341200 18156700 1331110
##  $ Adj.Close    : num  107 108 108 109 111 ...
##  $ Daily_Return : num  NA 1.029 0.037 1.073 1.208 ...
##  $ MA_20        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ MA_50        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ MA_200       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Volatility_20: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Year         : int  2016 2016 2016 2016 2016 2016 2016 2016 2016 2016 ...
##  $ Month        : int  1 2 2 2 2 2 2 2 2 2 ...
##  $ Day_of_Week  : int  4 0 1 2 3 4 0 1 2 3 ...
##  $ Quarter      : int  1 1 1 1 1 1 1 1 1 1 ...
```

```r
# Dimensions
cat("Dataset contains", nrow(gold), "rows and", ncol(gold), "columns\n")
```

```
## Dataset contains 2511 rows and 16 columns
```

```r
cat("Date range:", min(gold$Date), "to", max(gold$Date))
```

```
## Date range: 2016-01-29 to 2026-01-23
```

```r
# Summary statistics for numeric columns
summary(gold[, c("Close", "Volume", "Daily_Return", "Volatility_20")])
```

```
##      Close           Volume          Daily_Return       Volatility_20
##  Min.   :107.0   Min.   : 1436500   Min.   :-6.42689   Min.   :0.2872
##  1st Qu.:123.6   1st Qu.: 5779900   1st Qu.:-0.44559   1st Qu.:0.6457
##  Median :164.9   Median : 7757000   Median : 0.07156   Median :0.8088
##  Mean   :173.2   Mean   : 9006663   Mean   : 0.06236   Mean   :0.8716
##  3rd Qu.:183.4   3rd Qu.:10696100   3rd Qu.: 0.55217   3rd Qu.:1.0026
##  Max.   :458.0   Max.   :62025000   Max.   : 4.90384   Max.   :2.4874
##                                     NA's   :1          NA's   :20
```

Your Turn: Initial Questions

Q1: How many trading days are in the dataset?

Q2: What is the range of closing prices over the 10-year period?

Q3: Which columns have missing values?

```r
# Count missing values per column
colSums(is.na(gold))
```

```
##          Date         Close          High           Low          Open
##             0             0             0             0             0
##        Volume     Adj.Close  Daily_Return         MA_20         MA_50
##             0             0             1            19            49
##        MA_200 Volatility_20          Year         Month   Day_of_Week
##           199            20             0             0             0
##       Quarter
##             0
```

Step 2: Data Cleaning and Preparation

2.1 Convert Date Column

```r
# Convert Date to proper date format
gold$Date <- as.Date(gold$Date)

# Verify the conversion
class(gold$Date)
```

```
## [1] "Date"
```

```r
range(gold$Date)
```

```
## [1] "2016-01-29" "2026-01-23"
```

2.2 Handle Missing Values

```r
# Which rows have missing Daily_Return?
missing_returns <- gold %>% filter(is.na(Daily_Return))
cat("Rows with missing Daily_Return:", nrow(missing_returns), "\n")
```

```
## Rows with missing Daily_Return: 1
```

```r
head(missing_returns[, c("Date", "Close", "Daily_Return")])
```

```
##         Date  Close Daily_Return
## 1 2016-01-29 106.95           NA
```

Note: The first row typically has no return because there's no previous day to compare.

```r
# For analysis, we'll work with complete cases for return-related analysis
gold_complete <- gold %>% filter(!is.na(Daily_Return))
cat("Complete observations:", nrow(gold_complete))
```

```
## Complete observations: 2510
```

2.3 Create Additional Variables

```r
# Add price change category
gold_complete <- gold_complete %>%
  mutate(
    Return_Category = case_when(
      Daily_Return < -2 ~ "Large Drop",
      Daily_Return < 0 ~ "Small Drop",
      Daily_Return < 2 ~ "Small Gain",
      TRUE ~ "Large Gain"
    ),
    Return_Category = factor(Return_Category,
                             levels = c("Large Drop", "Small Drop", "Small Gain", "Large Gain")),
    # Price level categories
    Price_Level = case_when(
      Close < 120 ~ "Low (<$120)",
      Close < 180 ~ "Medium ($120-180)",
      TRUE ~ "High (>$180)"
    ),
    Price_Level = factor(Price_Level, levels = c("Low (<$120)", "Medium ($120-180)", "High (>$180)"))
  )

# Check the distribution
table(gold_complete$Return_Category)
```

```
##
## Large Drop Small Drop Small Gain Large Gain
##         44       1102       1313         51
```

```r
table(gold_complete$Price_Level)
```

```
##
##       Low (<$120) Medium ($120-180)      High (>$180)
##               364              1420               726
```

Step 3: Summary Statistics for Gold Prices

3.1 Central Tendency of Closing Prices

```r
# Mean closing price
mean_price <- mean(gold_complete$Close)
cat("Mean Closing Price: $", round(mean_price, 2), "\n")
```

```
## Mean Closing Price: $ 173.24
```

```r
# Median closing price
median_price <- median(gold_complete$Close)
cat("Median Closing Price: $", round(median_price, 2), "\n")
```

```
## Median Closing Price: $ 164.93
```

```r
# Difference
cat("Mean - Median: $", round(mean_price - median_price, 2))
```

```
## Mean - Median: $ 8.3
```

Interpretation: A positive difference (mean > median) suggests the price distribution is right-skewed, with some periods of exceptionally high prices.

3.2 Spread of Daily Returns

```r
# Standard deviation of returns
sd_return <- sd(gold_complete$Daily_Return)
cat("Std Dev of Daily Returns:", round(sd_return, 4), "%\n")
```

```
## Std Dev of Daily Returns: 0.9373 %
```

```r
# Range
return_range <- range(gold_complete$Daily_Return)
cat("Return Range:", round(return_range[1], 2), "% to", round(return_range[2], 2), "%\n")
```

```
## Return Range: -6.43 % to 4.9 %
```

```r
# IQR
iqr_return <- IQR(gold_complete$Daily_Return)
cat("IQR of Returns:", round(iqr_return, 4), "%")
```

```
## IQR of Returns: 0.9978 %
```

3.3 Quantiles of Returns

```r
# Key percentiles for risk analysis
quantile(gold_complete$Daily_Return,
         probs = c(0.01, 0.05, 0.10, 0.25, 0.50, 0.75, 0.90, 0.95, 0.99))
```

```
##          1%          5%         10%         25%         50%         75%
## -2.39329941 -1.47384246 -0.97451017 -0.44558560  0.07156283  0.55216608
##         90%         95%         99%
##  1.15488999  1.56608672  2.44154449
```

Interpretation: - The 1st percentile shows the worst 1% of daily returns (Value at Risk) - The 99th percentile shows the best 1% of daily returns

3.4 Summary by Year

```r
yearly_stats <- gold_complete %>%
  group_by(Year) %>%
  summarise(
    Trading_Days = n(),
    Mean_Price = mean(Close),
    Median_Price = median(Close),
    Min_Price = min(Close),
    Max_Price = max(Close),
    Avg_Return = mean(Daily_Return),
    Volatility = sd(Daily_Return)
  ) %>%
  arrange(Year)

yearly_stats
```

```
## # A tibble: 11 x 8
##     Year Trading_Days Mean_Price Median_Price Min_Price Max_Price Avg_Return
##    <int>        <int>      <dbl>        <dbl>     <dbl>     <dbl>      <dbl>
##  1  2016          233       121.         121.      107.      131.     0.0158
```

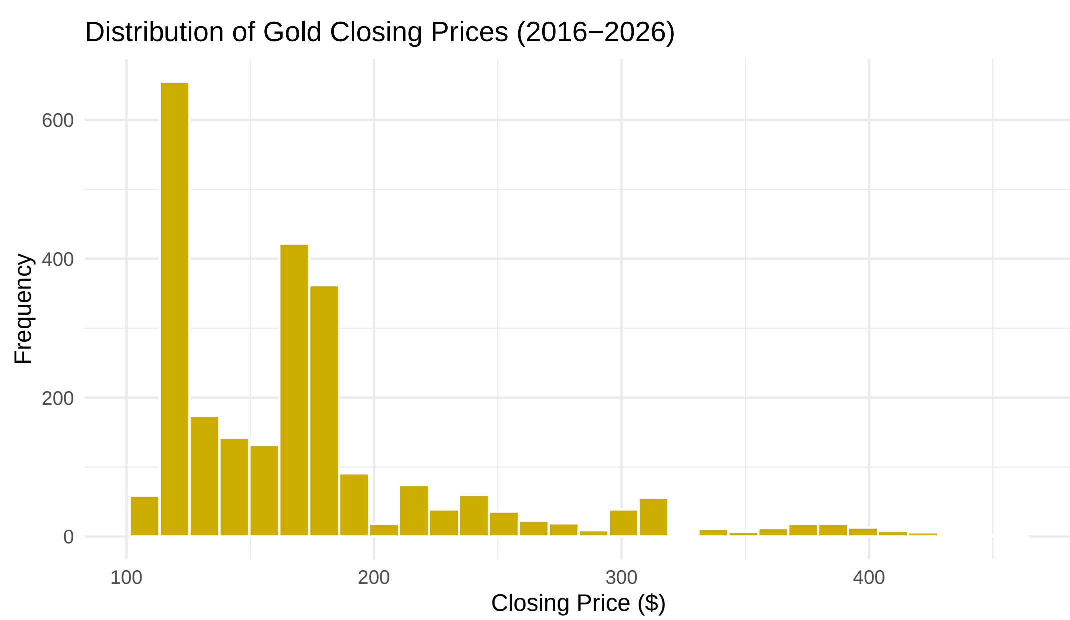
```
##  2  2017           251        120.          120.       110.       128.     0.0500
##  3  2018           251        120.          120.       111.       129.    -0.00593
##  4  2019           252        132.          133.       120.       147.     0.0679
##  5  2020           253        167.          167.       138.       194.     0.0952
##  6  2021           252        168.          168.       157.       183.    -0.0131
##  7  2022           251        168.          168.       151.       192.     0.00152
##  8  2023           250        180.          181.       168.       193.     0.0513
##  9  2024           252        221.          221.       184.       258.     0.0983
## 10  2025           250        318.          308.       243.       417.     0.205
## 11  2026            15        424.          422.       398.       458      0.977
## # i 1 more variable: Volatility <dbl>
```

Step 4: Visualizing Single Numeric Variables

4.1 Histogram of Closing Prices

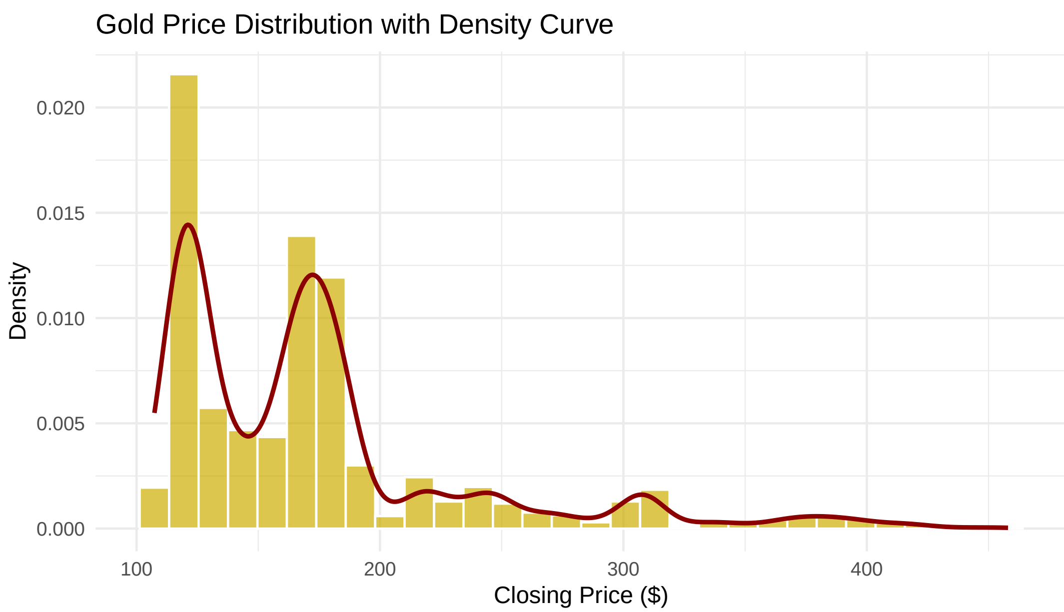
Basic Histogram

```r
ggplot(gold_complete, aes(x = Close)) +
  geom_histogram(bins = 30, fill = "gold3", color = "white") +
  labs(
    title = "Distribution of Gold Closing Prices (2016-2026)",
    x = "Closing Price ($)",
    y = "Frequency"
  ) +
  theme_minimal()
```



Histogram with Density Curve

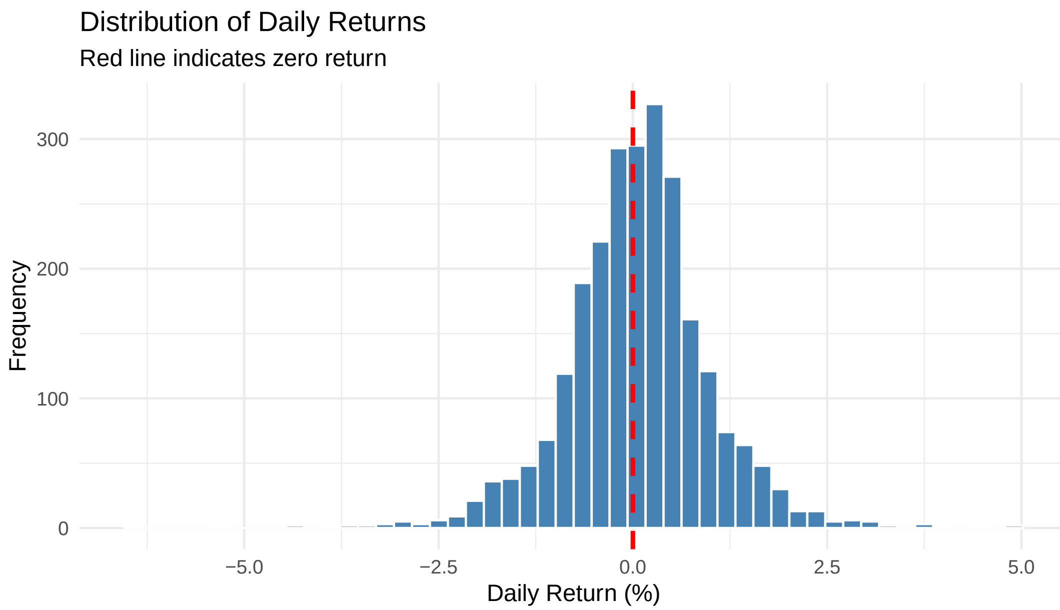
```
ggplot(gold_complete, aes(x = Close)) +
  geom_histogram(aes(y = after_stat(density)), bins = 30,
                 fill = "gold3", color = "white", alpha = 0.7) +
  geom_density(color = "darkred", linewidth = 1) +
  labs(
    title = "Gold Price Distribution with Density Curve",
    x = "Closing Price ($)",
    y = "Density"
  ) +
  theme_minimal()
```



4.2 Distribution of Daily Returns

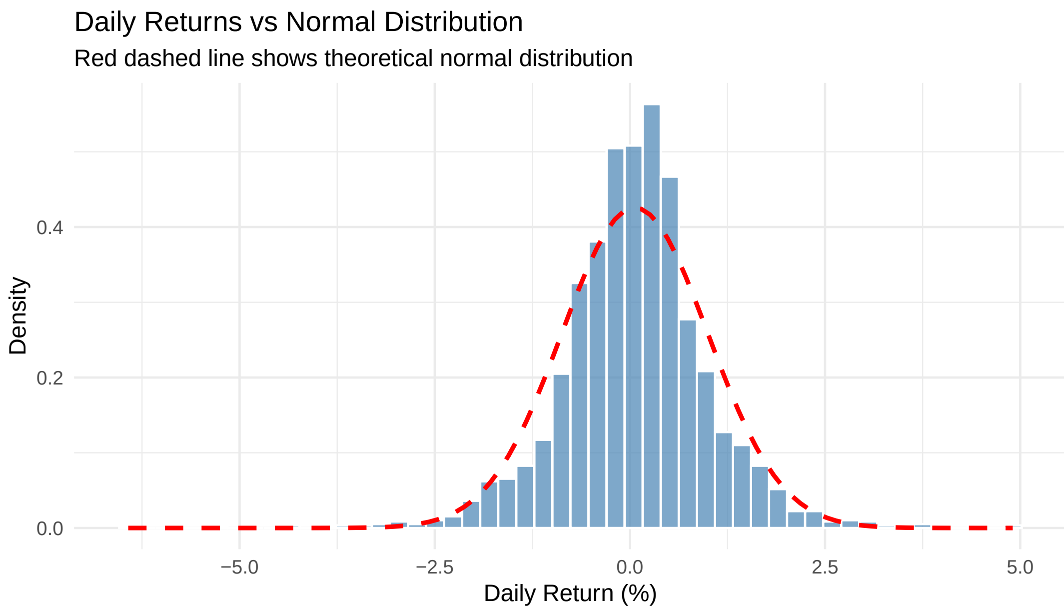
Histogram of Returns

```
ggplot(gold_complete, aes(x = Daily_Return)) +
  geom_histogram(bins = 50, fill = "steelblue", color = "white") +
  geom_vline(xintercept = 0, color = "red", linetype = "dashed", linewidth = 1) +
  labs(
    title = "Distribution of Daily Returns",
    subtitle = "Red line indicates zero return",
    x = "Daily Return (%)",
    y = "Frequency"
  ) +
  theme_minimal()
```



Returns with Normal Overlay

```
# Calculate mean and sd for normal curve
mean_ret <- mean(gold_complete$Daily_Return)
sd_ret <- sd(gold_complete$Daily_Return)

ggplot(gold_complete, aes(x = Daily_Return)) +
  geom_histogram(aes(y = after_stat(density)), bins = 50,
                 fill = "steelblue", color = "white", alpha = 0.7) +
  stat_function(fun = dnorm, args = list(mean = mean_ret, sd = sd_ret),
                color = "red", linewidth = 1, linetype = "dashed") +
  labs(
    title = "Daily Returns vs Normal Distribution",
    subtitle = "Red dashed line shows theoretical normal distribution",
    x = "Daily Return (%)",
    y = "Density"
  ) +
  theme_minimal()
```

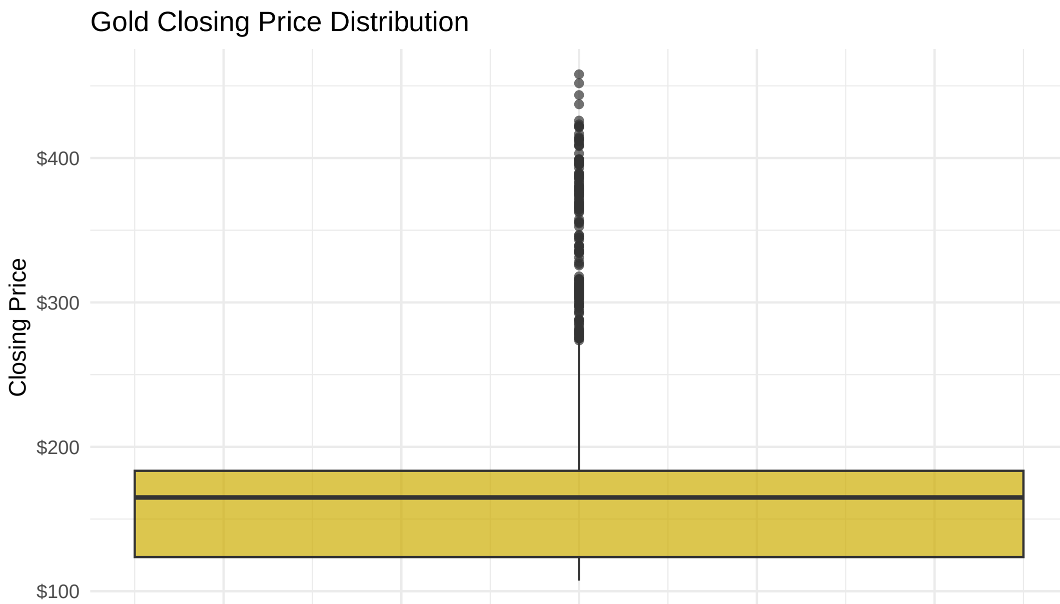


Note: Financial returns often have "fat tails" - more extreme values than a normal distribution would predict.

4.3 Boxplots

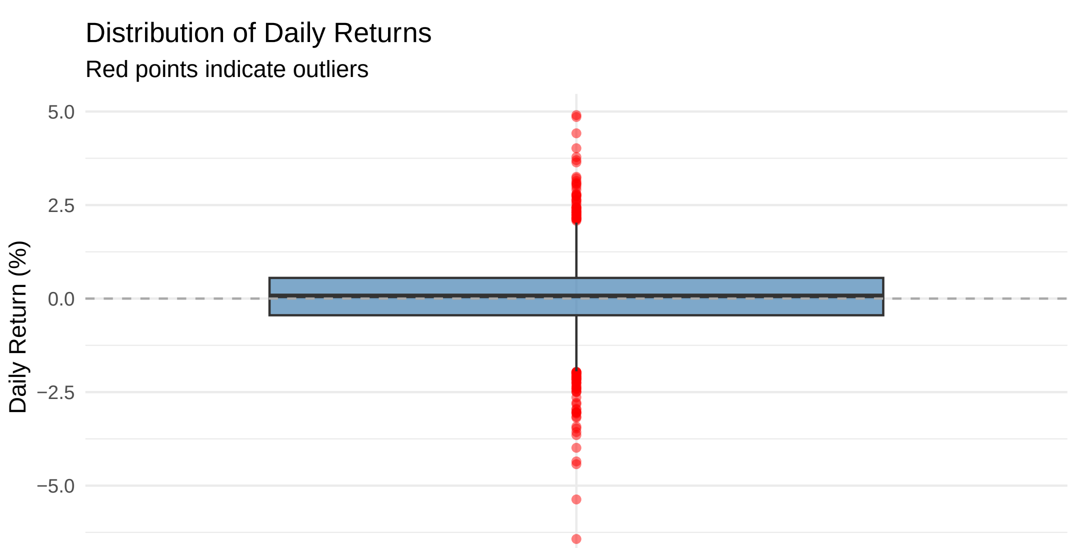
Basic Boxplot of Prices

```
ggplot(gold_complete, aes(y = Close)) +
  geom_boxplot(fill = "gold3", alpha = 0.7, width = 0.5) +
  scale_y_continuous(labels = dollar_format()) +
  labs(
    title = "Gold Closing Price Distribution",
    y = "Closing Price"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())
```



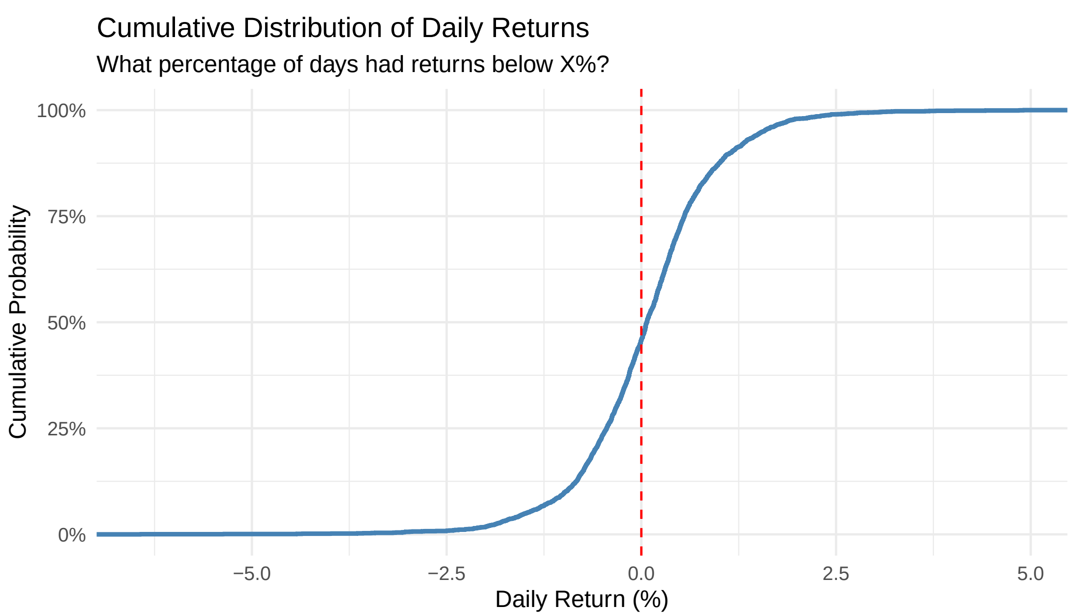
Boxplot of Returns

```
ggplot(gold_complete, aes(x = "", y = Daily_Return)) +
  geom_boxplot(fill = "steelblue", alpha = 0.7, outlier.color = "red", outlier.alpha = 0.5) +
  geom_hline(yintercept = 0, color = "darkgray", linetype = "dashed") +
  labs(
    title = "Distribution of Daily Returns",
    subtitle = "Red points indicate outliers",
    x = "",
    y = "Daily Return (%)"
  ) +
  theme_minimal()
```



4.4 Empirical CDF of Returns

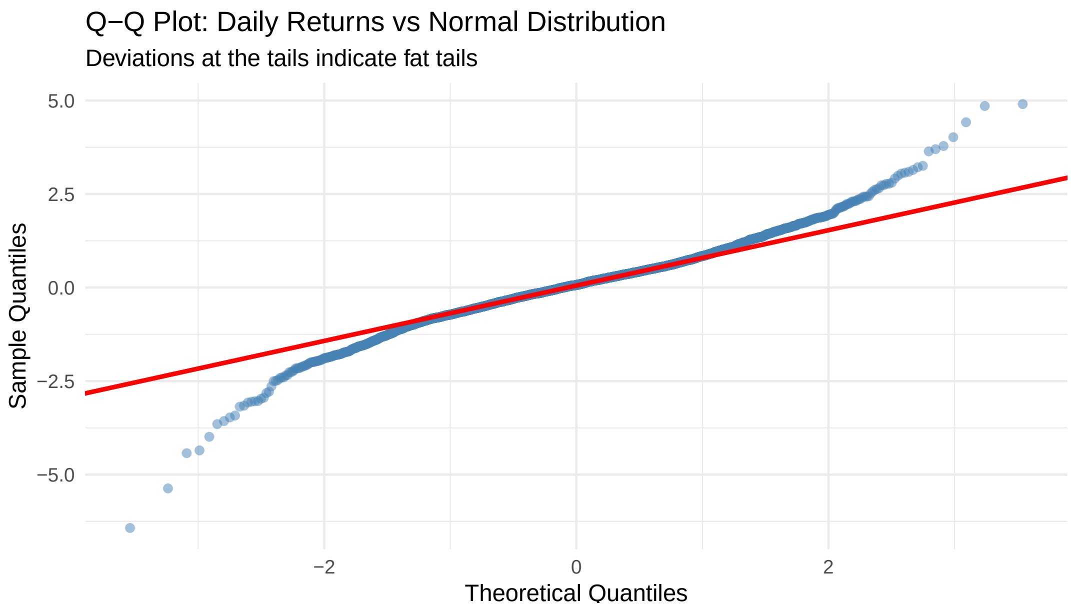
```r
ggplot(gold_complete, aes(x = Daily_Return)) +
  stat_ecdf(geom = "step", color = "steelblue", linewidth = 1) +
  geom_vline(xintercept = 0, color = "red", linetype = "dashed") +
  scale_y_continuous(labels = percent_format()) +
  labs(
    title = "Cumulative Distribution of Daily Returns",
    subtitle = "What percentage of days had returns below X%?",
    x = "Daily Return (%)",
    y = "Cumulative Probability"
  ) +
  theme_minimal()
```



Reading the eCDF: Find where the curve crosses 0% return - this tells you what percentage of days had negative returns.

4.5 Q-Q Plot for Returns

```r
ggplot(gold_complete, aes(sample = Daily_Return)) +
  stat_qq(color = "steelblue", alpha = 0.5) +
  stat_qq_line(color = "red", linewidth = 1) +
  labs(
    title = "Q-Q Plot: Daily Returns vs Normal Distribution",
    subtitle = "Deviations at the tails indicate fat tails",
    x = "Theoretical Quantiles",
    y = "Sample Quantiles"
  ) +
  theme_minimal()
```



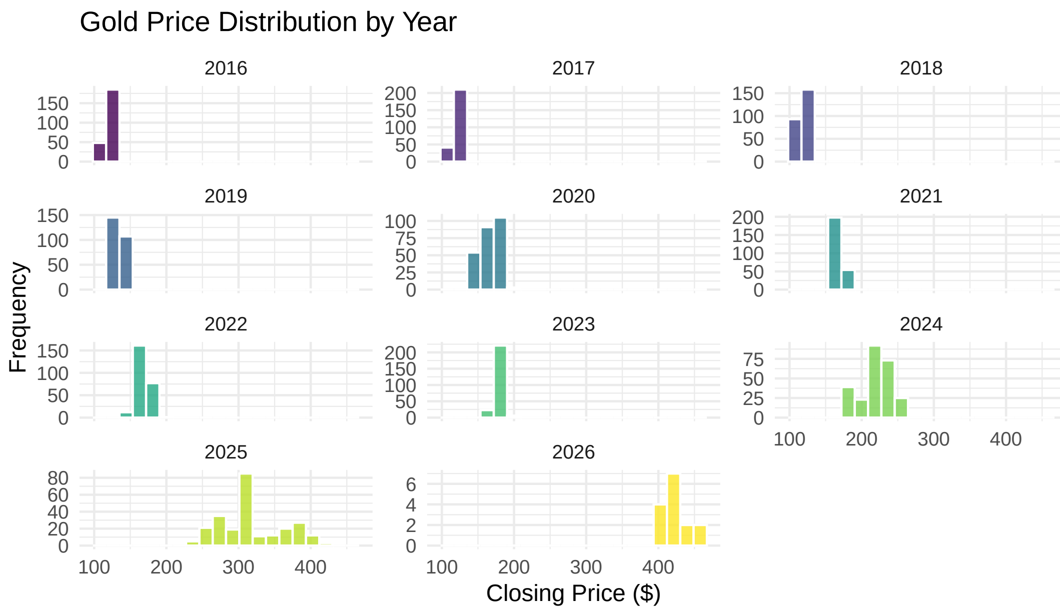
Interpretation: The S-shape at the ends indicates "fat tails" - extreme returns occur more often than a normal distribution would predict. This is typical for financial data.

Step 5: Comparing Distributions Across Groups

5.1 Price Distribution by Year

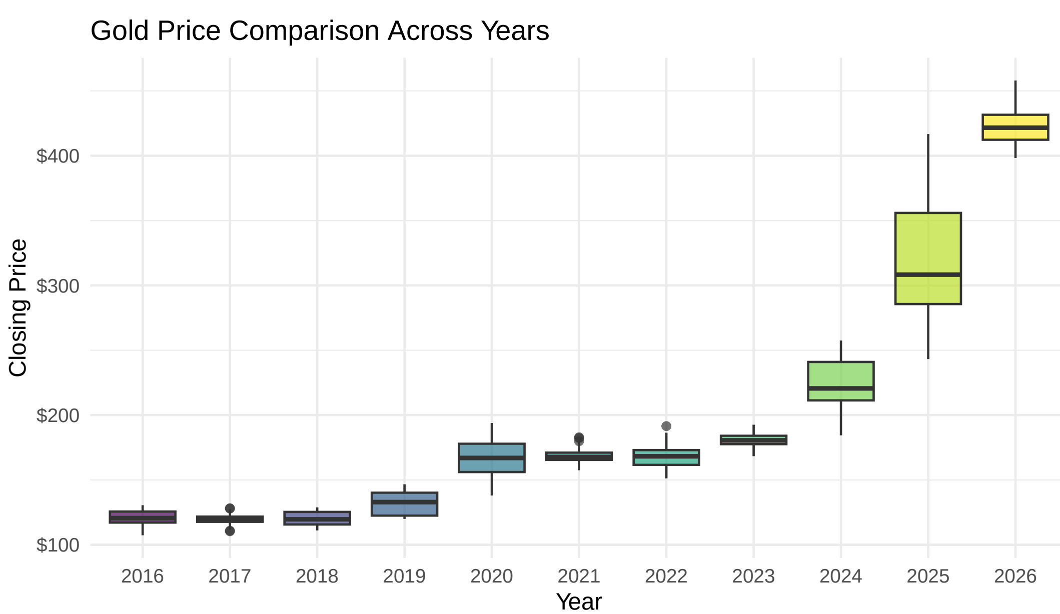
Faceted Histograms

```
ggplot(gold_complete, aes(x = Close, fill = factor(Year))) +
  geom_histogram(bins = 20, color = "white", alpha = 0.8) +
  facet_wrap(~ Year, scales = "free_y", ncol = 3) +
  scale_fill_viridis_d() +
  labs(
    title = "Gold Price Distribution by Year",
    x = "Closing Price ($)",
    y = "Frequency"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```



Boxplots by Year

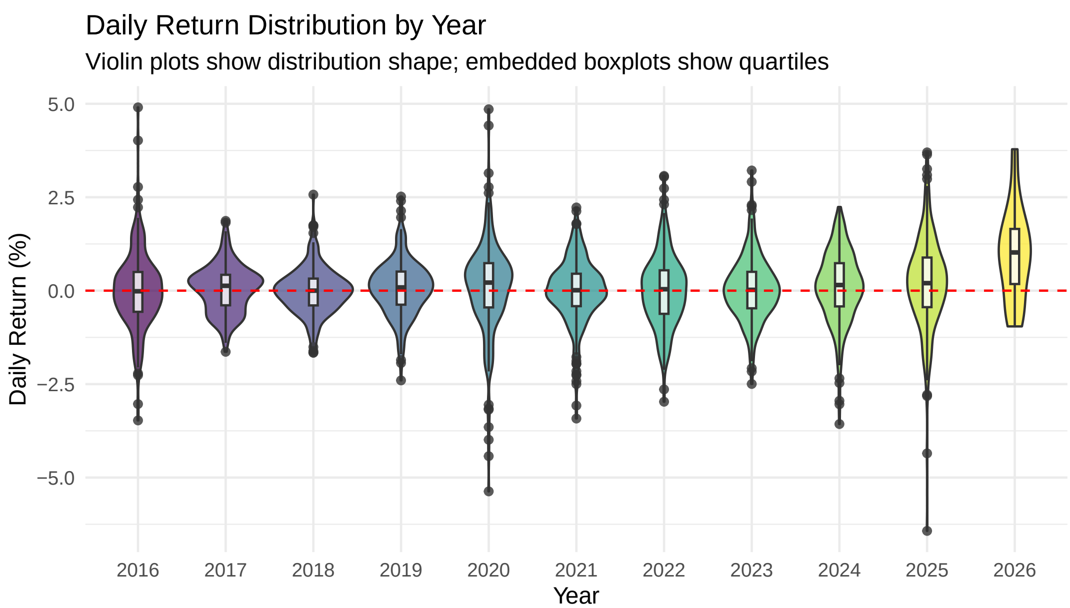
```r
ggplot(gold_complete, aes(x = factor(Year), y = Close, fill = factor(Year))) +
  geom_boxplot(alpha = 0.7) +
  scale_y_continuous(labels = dollar_format()) +
  scale_fill_viridis_d() +
  labs(
    title = "Gold Price Comparison Across Years",
    x = "Year",
    y = "Closing Price"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```



5.2 Returns Distribution by Year

Violin Plots

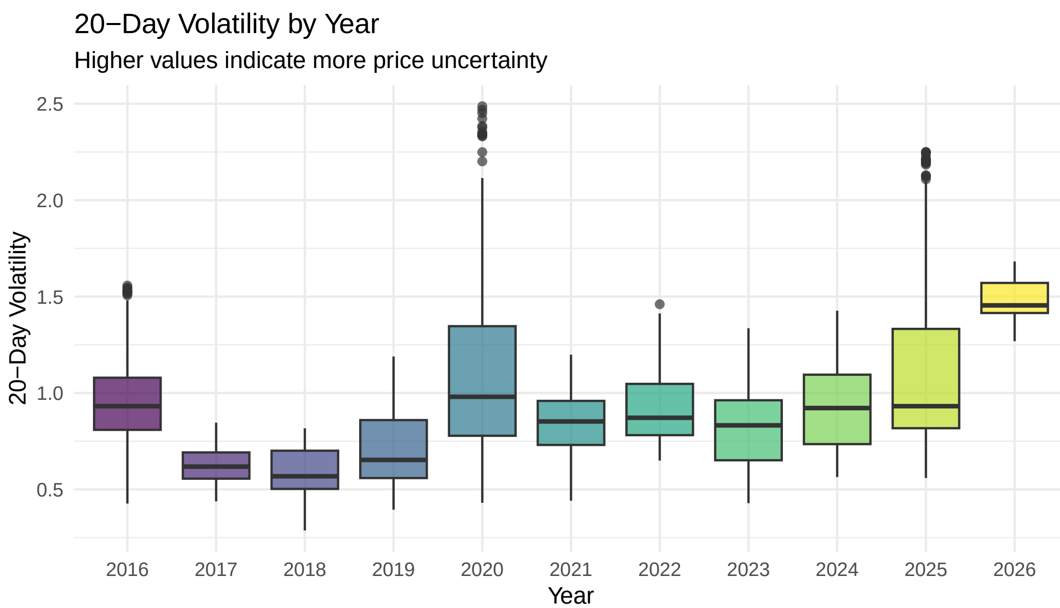
```r
ggplot(gold_complete, aes(x = factor(Year), y = Daily_Return, fill = factor(Year))) +
  geom_violin(alpha = 0.7) +
  geom_boxplot(width = 0.1, fill = "white", alpha = 0.8) +
  geom_hline(yintercept = 0, color = "red", linetype = "dashed") +
  scale_fill_viridis_d() +
  labs(
    title = "Daily Return Distribution by Year",
    subtitle = "Violin plots show distribution shape; embedded boxplots show quartiles",
    x = "Year",
    y = "Daily Return (%)"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```



5.3 Volatility Comparison

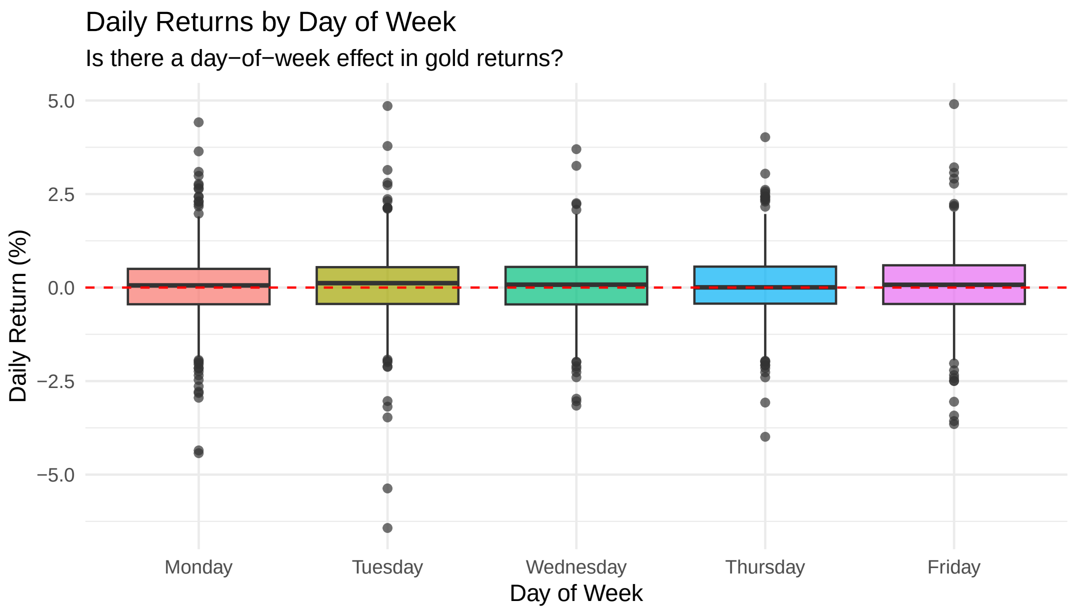
```r
# Filter for complete volatility data
gold_vol <- gold_complete %>% filter(!is.na(Volatility_20))

ggplot(gold_vol, aes(x = factor(Year), y = Volatility_20, fill = factor(Year))) +
  geom_boxplot(alpha = 0.7) +
  scale_fill_viridis_d() +
  labs(
    title = "20-Day Volatility by Year",
    subtitle = "Higher values indicate more price uncertainty",
    x = "Year",
    y = "20-Day Volatility"
```

```r
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```



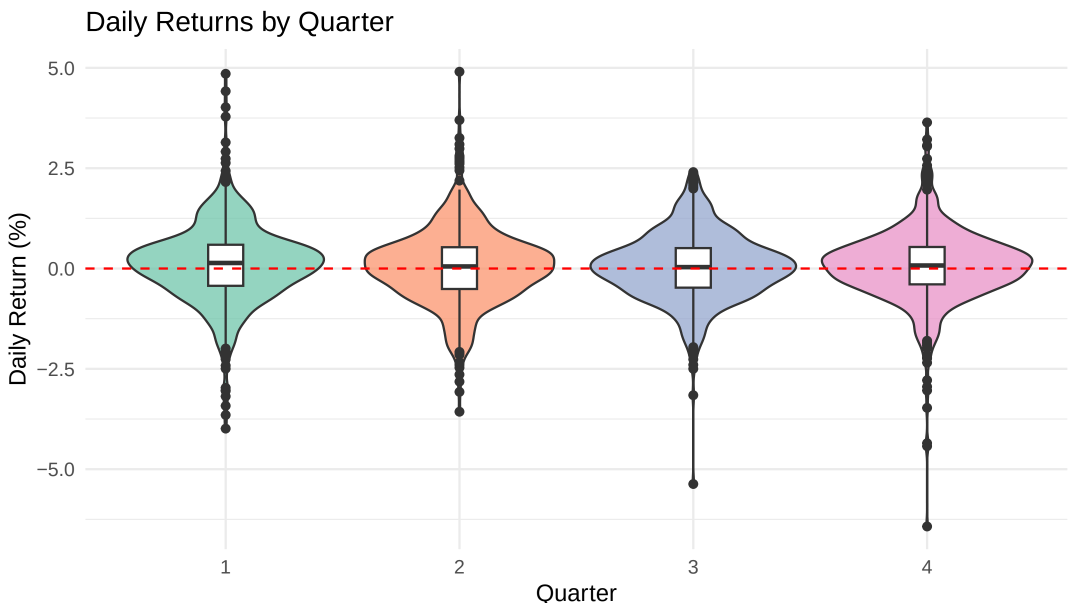
5.4 Returns by Day of Week

```r
# Create day names
dow_names <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
gold_complete$Day_Name <- factor(dow_names[gold_complete$Day_of_Week + 1],
                                  levels = dow_names)

ggplot(gold_complete, aes(x = Day_Name, y = Daily_Return, fill = Day_Name)) +
  geom_boxplot(alpha = 0.7) +
  geom_hline(yintercept = 0, color = "red", linetype = "dashed") +
  labs(
    title = "Daily Returns by Day of Week",
    subtitle = "Is there a day-of-week effect in gold returns?",
    x = "Day of Week",
    y = "Daily Return (%)"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```



5.5 Returns by Quarter

```
ggplot(gold_complete, aes(x = factor(Quarter), y = Daily_Return, fill = factor(Quarter))) +
  geom_violin(alpha = 0.7) +
  geom_boxplot(width = 0.15, fill = "white") +
  geom_hline(yintercept = 0, color = "red", linetype = "dashed") +
  scale_fill_brewer(palette = "Set2") +
  labs(
    title = "Daily Returns by Quarter",
    x = "Quarter",
    y = "Daily Return (%)"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```



Step 6: Identifying Outliers

6.1 Extreme Returns

```
# Calculate bounds using IQR method
Q1_ret <- quantile(gold_complete$Daily_Return, 0.25)
Q3_ret <- quantile(gold_complete$Daily_Return, 0.75)
IQR_ret <- Q3_ret - Q1_ret

lower_bound <- Q1_ret - 1.5 * IQR_ret
upper_bound <- Q3_ret + 1.5 * IQR_ret

cat("Normal return range:", round(lower_bound, 2), "% to", round(upper_bound, 2), "%\n\n")
```

```
## Normal return range: -1.94 % to 2.05 %
```

```
# Find extreme days
extreme_days <- gold_complete %>%
  filter(Daily_Return < lower_bound | Daily_Return > upper_bound) %>%
  select(Date, Close, Daily_Return, Volume) %>%
  arrange(Daily_Return)

cat("Number of extreme return days:", nrow(extreme_days), "\n")
```

```
## Number of extreme return days: 102
```

```r
cat("That's", round(nrow(extreme_days)/nrow(gold_complete)*100, 1), "% of trading days\n")
```

```
## That's 4.1 % of trading days
```

6.2 Worst and Best Days

```r
# Worst 10 days
cat("=== 10 WORST TRADING DAYS ===\n")
```

```
## === 10 WORST TRADING DAYS ===
```

```r
head(extreme_days, 10)
```

```
##          Date  Close Daily_Return   Volume
## 1  2025-10-21 377.24    -6.426889 54101400
## 2  2020-08-11 179.94    -5.369441 45355000
## 3  2020-11-09 175.08    -4.427098 29800700
## 4  2025-12-29 398.60    -4.352830 20679200
## 5  2020-03-12 147.79    -3.988826 32893400
## 6  2020-02-28 148.38    -3.649347 42699600
## 7  2024-06-07 211.60    -3.568330 12195100
## 8  2016-10-04 120.97    -3.471113 24372700
## 9  2021-01-08 173.34    -3.420994 24399900
## 10 2020-03-31 148.05    -3.184669 13319500
```

```r
# Best 10 days
cat("\n=== 10 BEST TRADING DAYS ===\n")
```

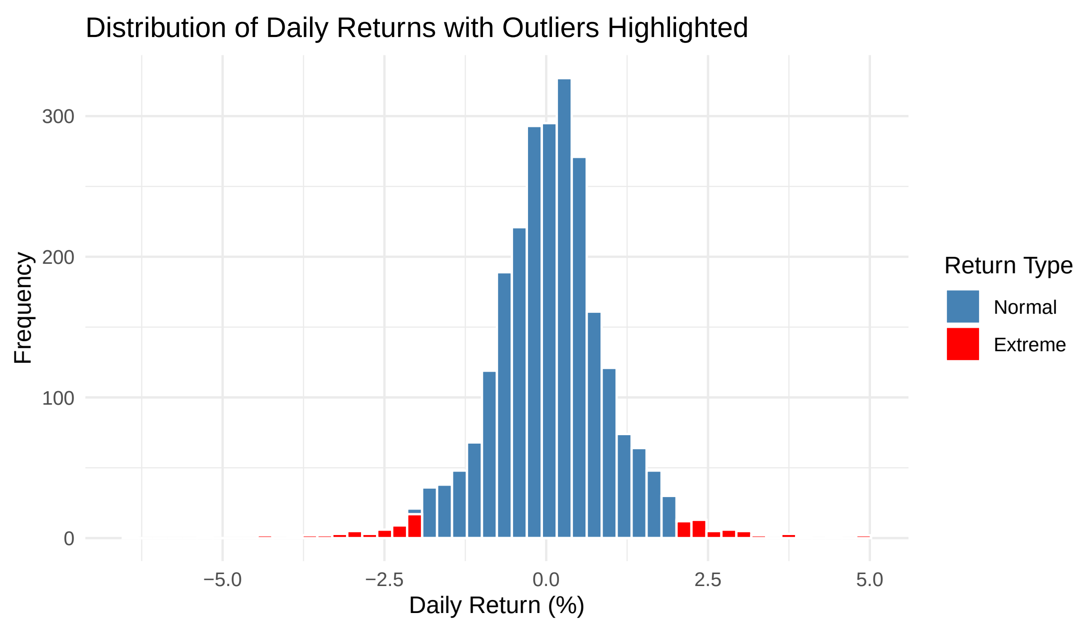
```
##
## === 10 BEST TRADING DAYS ===
```

```r
tail(extreme_days, 10)
```

```
##           Date  Close Daily_Return   Volume
## 93  2020-03-03 153.89     3.143433 28687700
## 94  2023-10-13 178.83     3.214826 18796500
## 95  2025-04-16 307.47     3.254081 20778100
## 96  2025-10-20 403.15     3.640197 34523000
## 97  2025-04-09 285.38     3.699125 25342200
## 98  2026-01-20 437.23     3.783617 21308100
## 99  2016-02-11 119.06     4.018870 49139000
## 100 2020-03-23 146.30     4.417959 28275200
## 101 2020-03-24 153.40     4.853035 20743500
## 102 2016-06-24 126.00     4.903838 35782900
```

6.3 Visualizing Extreme Returns

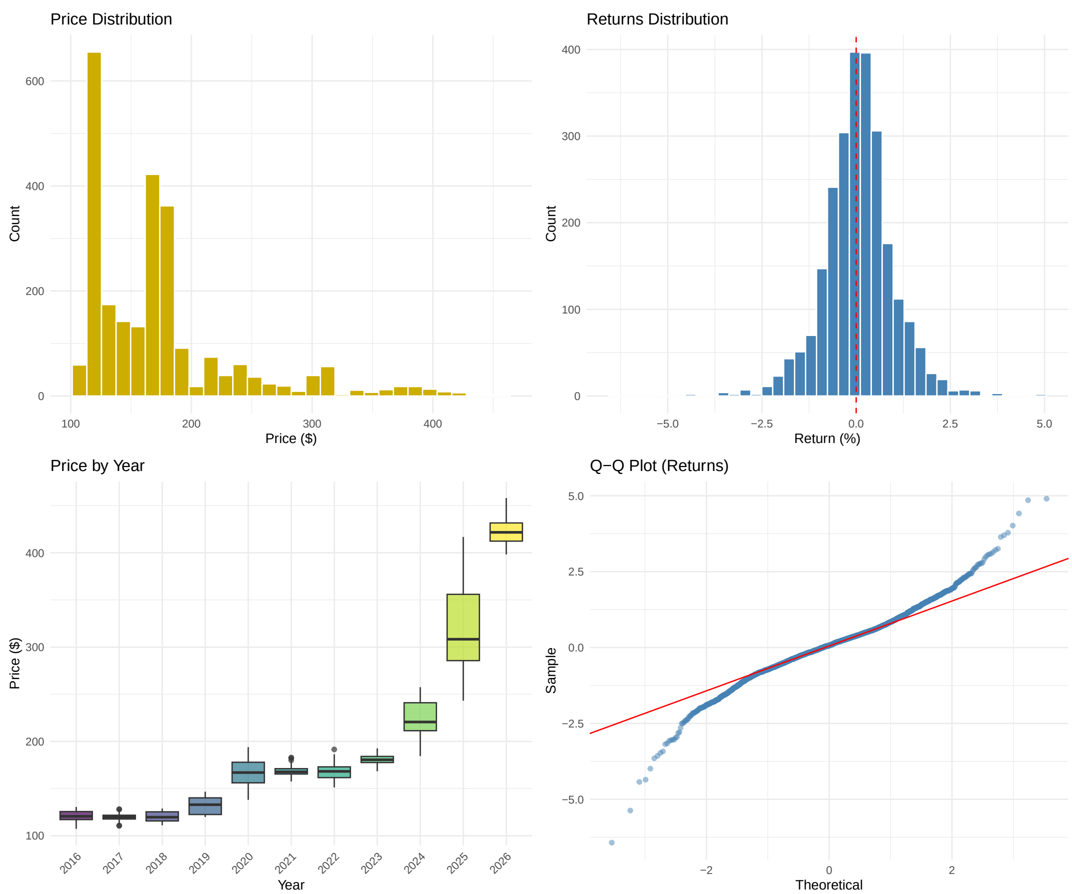
```r
gold_complete <- gold_complete %>%
  mutate(Is_Extreme = Daily_Return < lower_bound | Daily_Return > upper_bound)

ggplot(gold_complete, aes(x = Daily_Return, fill = Is_Extreme)) +
  geom_histogram(bins = 50, color = "white") +
  scale_fill_manual(values = c("steelblue", "red"),
                    labels = c("Normal", "Extreme")) +
  labs(
    title = "Distribution of Daily Returns with Outliers Highlighted",
    x = "Daily Return (%)",
    y = "Frequency",
    fill = "Return Type"
  ) +
  theme_minimal()
```



Step 7: Summary Dashboard

```r
library(gridExtra)

# Price histogram
p1 <- ggplot(gold_complete, aes(x = Close)) +
  geom_histogram(bins = 30, fill = "gold3", color = "white") +
  labs(title = "Price Distribution", x = "Price ($)", y = "Count") +
  theme_minimal()

# Returns histogram
p2 <- ggplot(gold_complete, aes(x = Daily_Return)) +
  geom_histogram(bins = 40, fill = "steelblue", color = "white") +
```

```r
  geom_vline(xintercept = 0, color = "red", linetype = "dashed") +
  labs(title = "Returns Distribution", x = "Return (%)", y = "Count") +
  theme_minimal()

# Price by year
p3 <- ggplot(gold_complete, aes(x = factor(Year), y = Close, fill = factor(Year))) +
  geom_boxplot(alpha = 0.7) +
  scale_fill_viridis_d() +
  labs(title = "Price by Year", x = "Year", y = "Price ($)") +
  theme_minimal() +
  theme(legend.position = "none", axis.text.x = element_text(angle = 45, hjust = 1))

# Q-Q plot
p4 <- ggplot(gold_complete, aes(sample = Daily_Return)) +
  stat_qq(color = "steelblue", alpha = 0.5) +
  stat_qq_line(color = "red") +
  labs(title = "Q-Q Plot (Returns)", x = "Theoretical", y = "Sample") +
  theme_minimal()

grid.arrange(p1, p2, p3, p4, ncol = 2)
```



Summary Statistics Report

```r
cat("=" , rep("=", 55), "\n", sep = "")
```

```
## ========================================================
```

```r
cat("GOLD PRICE ANALYSIS SUMMARY (2016-2026)\n")
```

```
## GOLD PRICE ANALYSIS SUMMARY (2016-2026)
```

```r
cat("=", rep("=", 55), "\n\n", sep = "")
```

```
## ========================================================
```

```r
cat("DATASET:\n")
```

```
## DATASET:
```

```r
cat("  Trading Days:", nrow(gold_complete), "\n")
```

```
##   Trading Days: 2510
```

```r
cat("  Date Range:", as.character(min(gold_complete$Date)), "to",
    as.character(max(gold_complete$Date)), "\n\n")
```

```
##   Date Range: 2016-02-01 to 2026-01-23
```

```r
cat("PRICE STATISTICS:\n")
```

```
## PRICE STATISTICS:
```

```r
cat("  Mean Price:   $", round(mean(gold_complete$Close), 2), "\n", sep = "")
```

```
##   Mean Price:   $173.24
```

```r
cat("  Median Price: $", round(median(gold_complete$Close), 2), "\n", sep = "")
```

```
##   Median Price: $164.93
```

```r
cat("  Min Price:    $", round(min(gold_complete$Close), 2), "\n", sep = "")
```

```
##   Min Price:    $107.34
```

```r
cat("  Max Price:    $", round(max(gold_complete$Close), 2), "\n\n", sep = "")
```

```
##   Max Price:    $458
```

```r
cat("RETURN STATISTICS:\n")
```

```
## RETURN STATISTICS:
```

```r
cat("  Mean Return:  ", round(mean(gold_complete$Daily_Return), 4), "%\n", sep = "")
```

```
##   Mean Return:  0.0624%
```

```r
cat("  Std Dev:      ", round(sd(gold_complete$Daily_Return), 4), "%\n", sep = "")
```

```
##   Std Dev:      0.9373%
```

```r
cat("  Worst Day:    ", round(min(gold_complete$Daily_Return), 2), "%\n", sep = "")
```

```
##   Worst Day:    -6.43%
```

```r
cat("  Best Day:     ", round(max(gold_complete$Daily_Return), 2), "%\n\n", sep = "")
```

```
##   Best Day:     4.9%
```

```r
cat("RISK METRICS (Value at Risk):\n")
```

```
## RISK METRICS (Value at Risk):
```

```r
cat("  1% VaR:       ", round(quantile(gold_complete$Daily_Return, 0.01), 2), "%\n", sep = "")
```

```
##   1% VaR:       -2.39%
```

```r
cat("  5% VaR:       ", round(quantile(gold_complete$Daily_Return, 0.05), 2), "%\n", sep = "")
```

```
##   5% VaR:       -1.47%
```

```r
cat("\nDISTRIBUTION:\n")
```

```
##
## DISTRIBUTION:
```

```r
pct_negative <- mean(gold_complete$Daily_Return < 0) * 100
cat("  Days with negative returns: ", round(pct_negative, 1), "%\n", sep = "")
```

```
##   Days with negative returns: 45.7%
```

```r
cat("  Days with positive returns: ", round(100 - pct_negative, 1), "%\n", sep = "")
```

```
##   Days with positive returns: 54.3%
```

```r
cat("  Extreme days (outliers):    ", nrow(extreme_days),
    " (", round(nrow(extreme_days)/nrow(gold_complete)*100, 1), "%)\n", sep = "")
```

```
##   Extreme days (outliers):    102 (4.1%)
```

Practice Exercises

Exercise 1: Monthly Analysis

Create a summary of gold prices by **Month** (1-12). Calculate the average return for each month to see if there's a seasonal pattern.

```r
# Your code here
```

Exercise 2: High Volatility Periods

Filter the data to find periods where `Volatility_20` is greater than its 90th percentile. Visualize the distribution of returns during these high-volatility periods compared to normal periods.

```r
# Your code here
```

Exercise 3: Volume Analysis

Create visualizations to explore the relationship between trading volume and: 1. Absolute daily returns (hint: create `abs(Daily_Return)`) 2. Whether volume is higher on up days vs down days

```r
# Your code here
```

Key Takeaways

1. **Financial data has unique characteristics**:
   - Returns are often approximately symmetric around zero
   - "Fat tails" - extreme events occur more often than normal distribution predicts
   - Volatility clusters - high volatility periods tend to persist
2. **Distribution analysis for prices vs returns**:
   - Prices: Often right-skewed, non-stationary (changes over time)
   - Returns: More symmetric, better for statistical analysis

3. **Risk metrics**:

   - Standard deviation measures typical fluctuation
   - Value at Risk (VaR) measures extreme downside risk
   - IQR and outlier detection identify unusual trading days

4. **Temporal patterns**:

   - Compare distributions across years, quarters, days of week
   - Look for seasonal effects or regime changes

5. **Visualization choices**:

   - Histograms: See the shape and frequency
   - Boxplots: Compare groups and spot outliers
   - Violin plots: See distribution shape across groups
   - Q-Q plots: Check normality assumptions

End of Practical Session 2