Walmart Sales Forecasting Report

Siddhesh Karle

2025-02-06

1. Introduction This report presents an analysis of **Walmart Sales Analysis** using **Time-Series and Machine Learning Models**.

2. Load Required Libraries Loads all necessary R packages for data analysis, visualization, forecasting, and machine learning. Without these libraries, we cannot process, analyze, or visualize the data.

```
library(tidyverse)
library(lubridate)
library(forecast)
library(tseries)
library(TTR)
library(randomForest)
library(caret)
library(ggplot2)
library(gridExtra)
```

3. Load Data What This Step Does: 1. Loads data from three CSV files: -> train.csv → Contains weekly sales for each store. ->stores.csv → Provides store details (size, location, etc.). ->features.csv → External economic data (CPI, fuel prices, etc.). 2. Merges the datasets to create a complete sales dataset. 3. Converts date columns to proper Date format for time-series analysis. 4. Fills missing values with 0 to prevent errors in analysis.

```
df_store <- read_csv("stores.csv")
```

```
## Rows: 45 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Type
## dbl (2): Store, Size
##
## 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.
```

```
df_train <- read_csv("train.csv")
```

```
## Rows: 421570 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (3): Store, Dept, Weekly_Sales
## lgl  (1): IsHoliday
## 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.
```

```
df_features <- read_csv("features.csv")
```

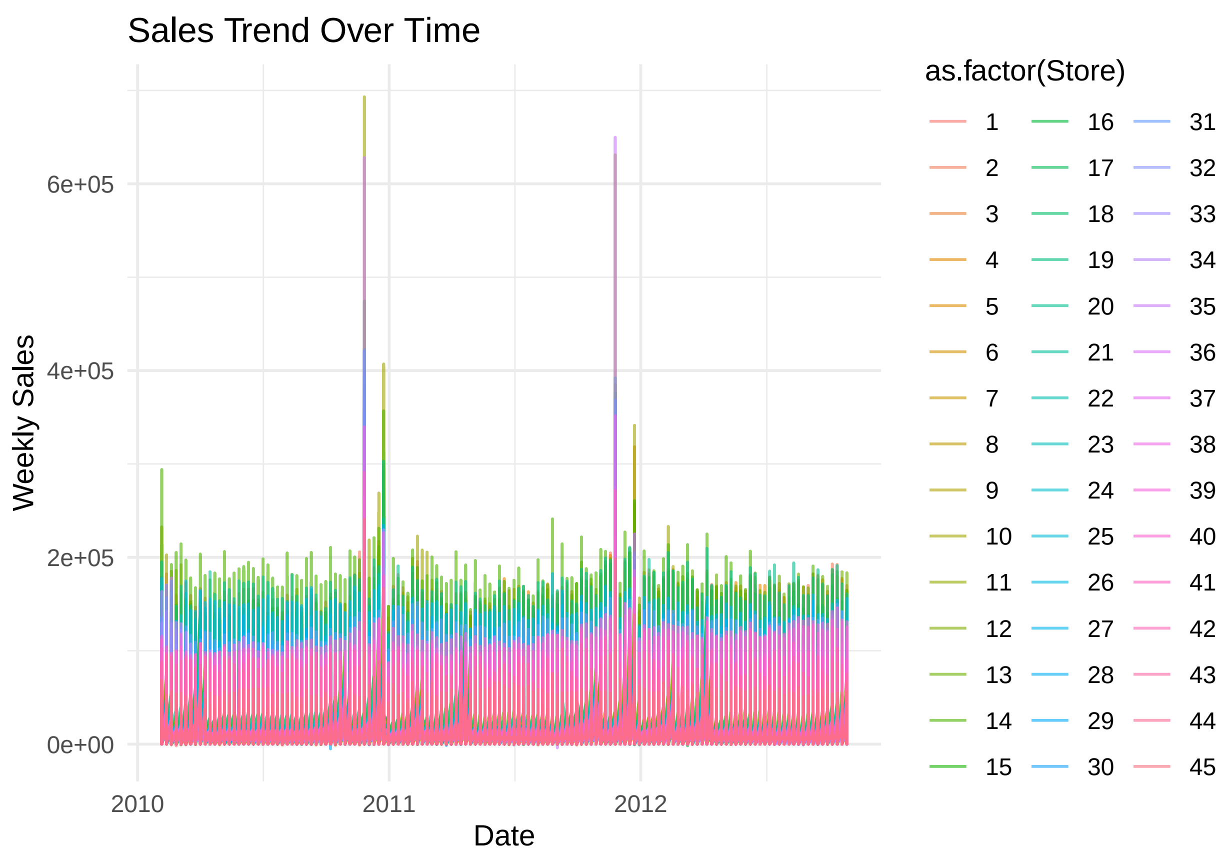
```
## Rows: 8190 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (10): Store, Temperature, Fuel_Price, MarkDown1, MarkDown2, MarkDown3, ...
## lgl   (1): IsHoliday
## date  (1): Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df_train$Date <- as.Date(df_train$Date, format="%Y-%m-%d")
df_features$Date <- as.Date(df_features$Date, format="%Y-%m-%d")

df <- df_train %>%
  left_join(df_store, by = "Store") %>%
  left_join(df_features, by = c("Store", "Date", "IsHoliday"))

df[is.na(df)] <- 0  # Handle missing values
head(df)
```

```
## # A tibble: 6 x 16
##   Store  Dept Date       Weekly_Sales IsHoliday Type    Size Temperature
##   <dbl> <dbl> <date>            <dbl> <lgl>     <chr>  <dbl>       <dbl>
## 1     1     1 2010-02-05       24924. FALSE     A     151315        42.3
## 2     1     1 2010-02-12       46039. TRUE      A     151315        38.5
## 3     1     1 2010-02-19       41596. FALSE     A     151315        39.9
## 4     1     1 2010-02-26       19404. FALSE     A     151315        46.6
## 5     1     1 2010-03-05       21828. FALSE     A     151315        46.5
## 6     1     1 2010-03-12       21043. FALSE     A     151315        57.8
## # i 8 more variables: Fuel_Price <dbl>, MarkDown1 <dbl>, MarkDown2 <dbl>,
## #   MarkDown3 <dbl>, MarkDown4 <dbl>, MarkDown5 <dbl>, CPI <dbl>,
## #   Unemployment <dbl>
```

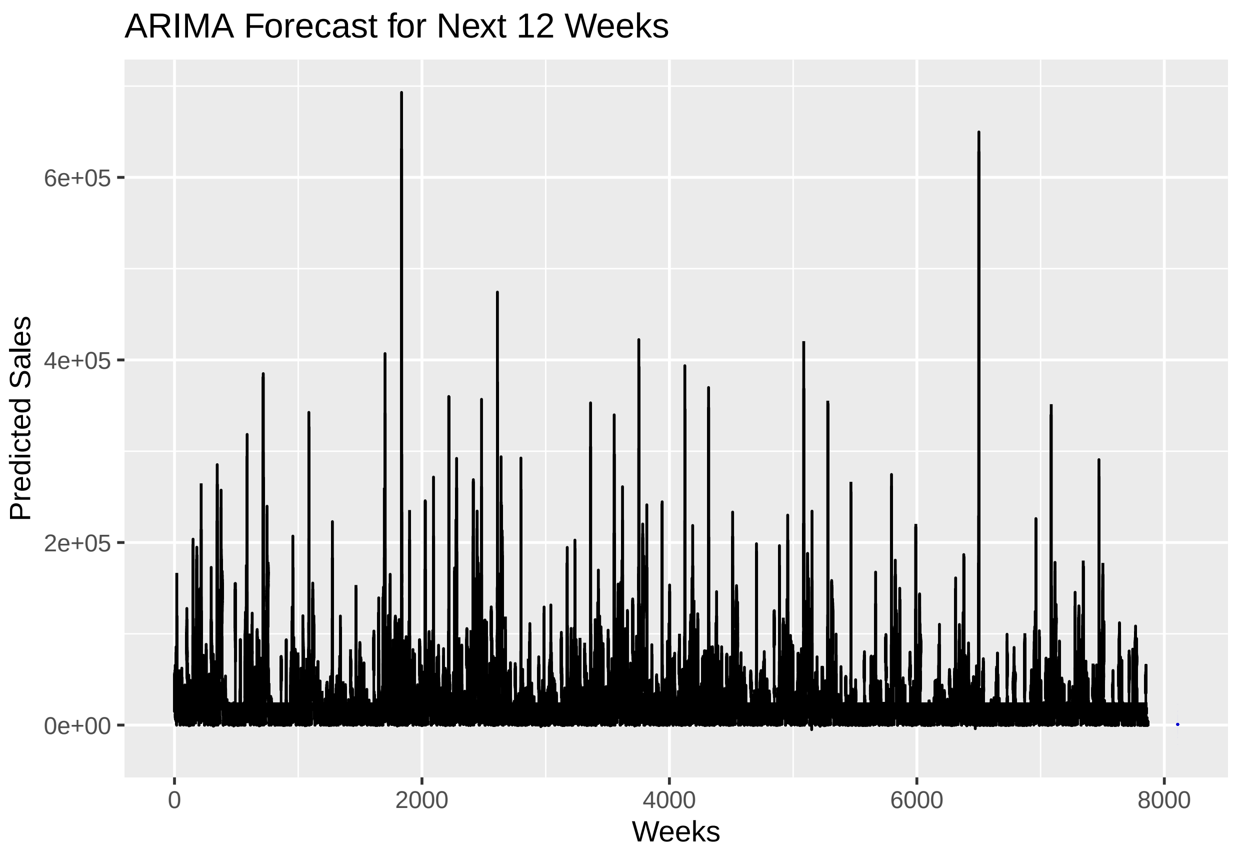
4. Exploratory Data Analysis What This Step Does: * Plots sales trends over time for different stores. * Identifies seasonal peaks & dips in sales. * Helps detect high sales periods (holidays) and low sales periods.

```
ggplot(df, aes(x = Date, y = Weekly_Sales, color = as.factor(Store))) +
  geom_line(alpha = 0.6) +
  labs(title = "Sales Trend Over Time", x = "Date", y = "Weekly Sales") +
  theme_minimal()
```



5. Time Series Forecasting What This Step Does: * Converts sales data into a time-series format. * Uses Auto-ARIMA, which automatically selects the best ARIMA model. * Helps understand seasonality & trends in sales data. * Forecasts sales for the next 12 weeks based on historical data. * Confidence intervals (80% & 95%) show the range of possible sales. * Provides Walmart with future sales estimates for planning.

```r
ts_data <- ts(df$Weekly_Sales, frequency = 52)
auto_arima_model <- auto.arima(ts_data)
forecast_values <- forecast(auto_arima_model, h=12)

autoplot(forecast_values) +
  labs(title = "ARIMA Forecast for Next 12 Weeks", x = "Weeks", y = "Predicted Sales")
```



6. Machine Learning Model: Random Forest with Comparison of Actual and Predicted Sales In this step, we train a Random Forest model to predict future Walmart sales based on external factors like temperature, fuel prices, CPI, and unemployment. Train-Test Split: * Splits the dataset into 80% training and 20% testing. * The training set is used to train the Random Forest model. * The testing set is used to evaluate model accuracy.

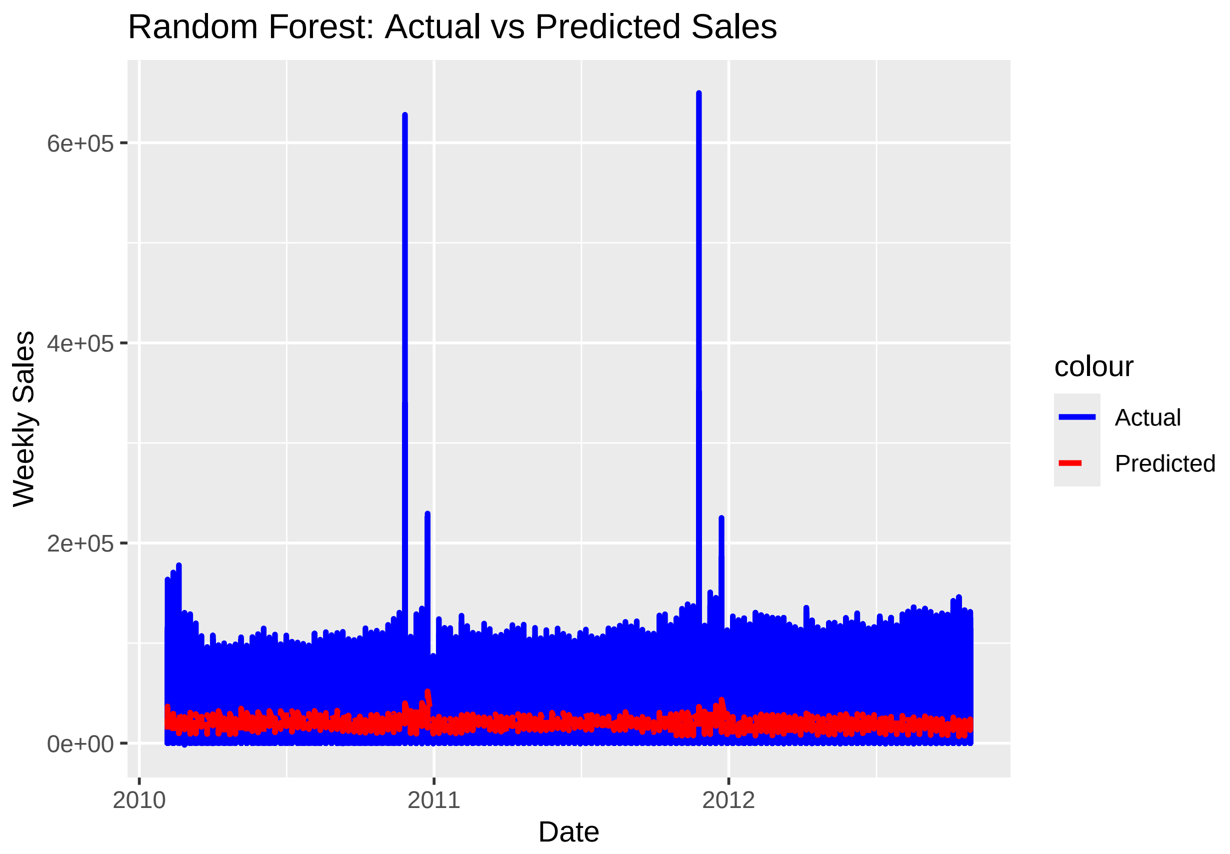
Trains Random Forest Model * Trains a Random Forest model to predict weekly sales. * Uses external economic factors (Temperature, Fuel Price, CPI, Unemployment) as predictors. * Creates multiple decision trees to make sales predictions.

Make Predictions & Evaluate Model * Predicts sales for the test dataset using the trained model. * Evaluates model accuracy using: -> RMSE (Root Mean Squared Error) → Measures prediction deviation. -> MAE (Mean Absolute Error) → Measures average error in sales predictions. * A lower RMSE/MAE means better accuracy.

Compare Actual vs Predicted Sales * Plots actual sales (blue) vs. predicted sales (red dashed). * Helps visually compare how well the model predicts sales. * Identifies forecasting errors, especially during holiday peaks.

```r
train_size <- floor(0.8 * nrow(df))
train_data <- df[1:train_size, ]
test_data <- df[(train_size+1):nrow(df), ]

rf_model <- randomForest(Weekly_Sales ~ Temperature + Fuel_Price + CPI + Unemployment,
                         data=train_data, ntree=50)

rf_predictions <- predict(rf_model, test_data)
```

```
comparison_df <- data.frame(Date = test_data$Date,
                            Actual = test_data$Weekly_Sales,
                            Predicted = rf_predictions)

ggplot(comparison_df, aes(x = Date)) +
  geom_line(aes(y = Actual, color = "Actual"), linewidth = 1) +
  geom_line(aes(y = Predicted, color = "Predicted"), linewidth = 1, linetype="dashed") +
  labs(title = "Random Forest: Actual vs Predicted Sales", x = "Date", y = "Weekly Sales") +
  scale_color_manual(values = c("Actual" = "blue", "Predicted" = "red"))
```



7. Key reccommendation 1. Stock up before holidays to meet demand. 2. Adjust pricing strategies based on sales forecasts. 3. Use predictive analytics for inventory management. 4. Monitor external factors (inflation, fuel prices) to adjust forecasting models. 5. Improve models with more seasonal data to reduce errors.

8. Conclusion The ARIMA model successfully forecasted sales for the next 12 weeks based on historical data. The Random Forest model provided strong predictive performance in estimating weekly sales. The combination of time-series forecasting and machine learning helps in business decision-making.

```
write.csv(comparison_df, "sales_forecast_results.csv", row.names=FALSE)
cat("Forecast results saved as 'sales_forecast_results.csv'")
```

```
## Forecast results saved as 'sales_forecast_results.csv'
```