Linear Regression Analysis

Adjei Anthony Juniior

2024-05-07

Forcasting Stock Prices With Linear Regression Analysis.

To assess the feasibility of predicting future stock price movement over time. I am going to use a historical data of a stock price data from github @mediasittich.

```r
# Import libraries
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

Exporing Stock Price Data and Preparing for Analysis..

```r
data<- read.csv("sap_stock.csv")
head(data)
```

```
##         Date  Open  High   Low Close Change Traded.Volume  Turnover
## 1 2009-03-09 25.16 25.82 24.48 25.59     NA       5749357 145200289
## 2 2009-03-10 25.68 26.95 25.68 26.87     NA       7507770 198480965
## 3 2009-03-11 26.50 26.95 26.26 26.64     NA       5855095 155815439
## 4 2009-03-12 26.15 26.47 25.82 26.18     NA       6294955 164489409
## 5 2009-03-13 26.01 26.24 25.65 25.73     NA       6814568 176228331
## 6 2009-03-16 26.22 26.66 25.94 26.48     NA       5248247 138331071
##   Last.Price.of.the.Day Daily.Traded.Units Daily.Turnover
## 1                    NA                 NA             NA
## 2                    NA                 NA             NA
## 3                    NA                 NA             NA
## 4                    NA                 NA             NA
## 5                    NA                 NA             NA
## 6                    NA                 NA             NA
```

```r
str(data)
```

```
## 'data.frame':    2550 obs. of  11 variables:
##  $ Date                 : chr  "2009-03-09" "2009-03-10" "2009-03-11" "2009-03-12" ...
##  $ Open                 : num  25.2 25.7 26.5 26.1 26 ...
##  $ High                 : num  25.8 26.9 26.9 26.5 26.2 ...
##  $ Low                  : num  24.5 25.7 26.3 25.8 25.6 ...
##  $ Close                : num  25.6 26.9 26.6 26.2 25.7 ...
##  $ Change               : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Traded.Volume        : num  5749357 7507770 5855095 6294955 6814568 ...
##  $ Turnover             : num  1.45e+08 1.98e+08 1.56e+08 1.64e+08 1.76e+08 ...
##  $ Last.Price.of.the.Day: logi  NA NA NA NA NA NA ...
##  $ Daily.Traded.Units   : logi  NA NA NA NA NA NA ...
##  $ Daily.Turnover       : num  NA NA NA NA NA NA NA NA NA NA ...
```

```r
summary(data)
```

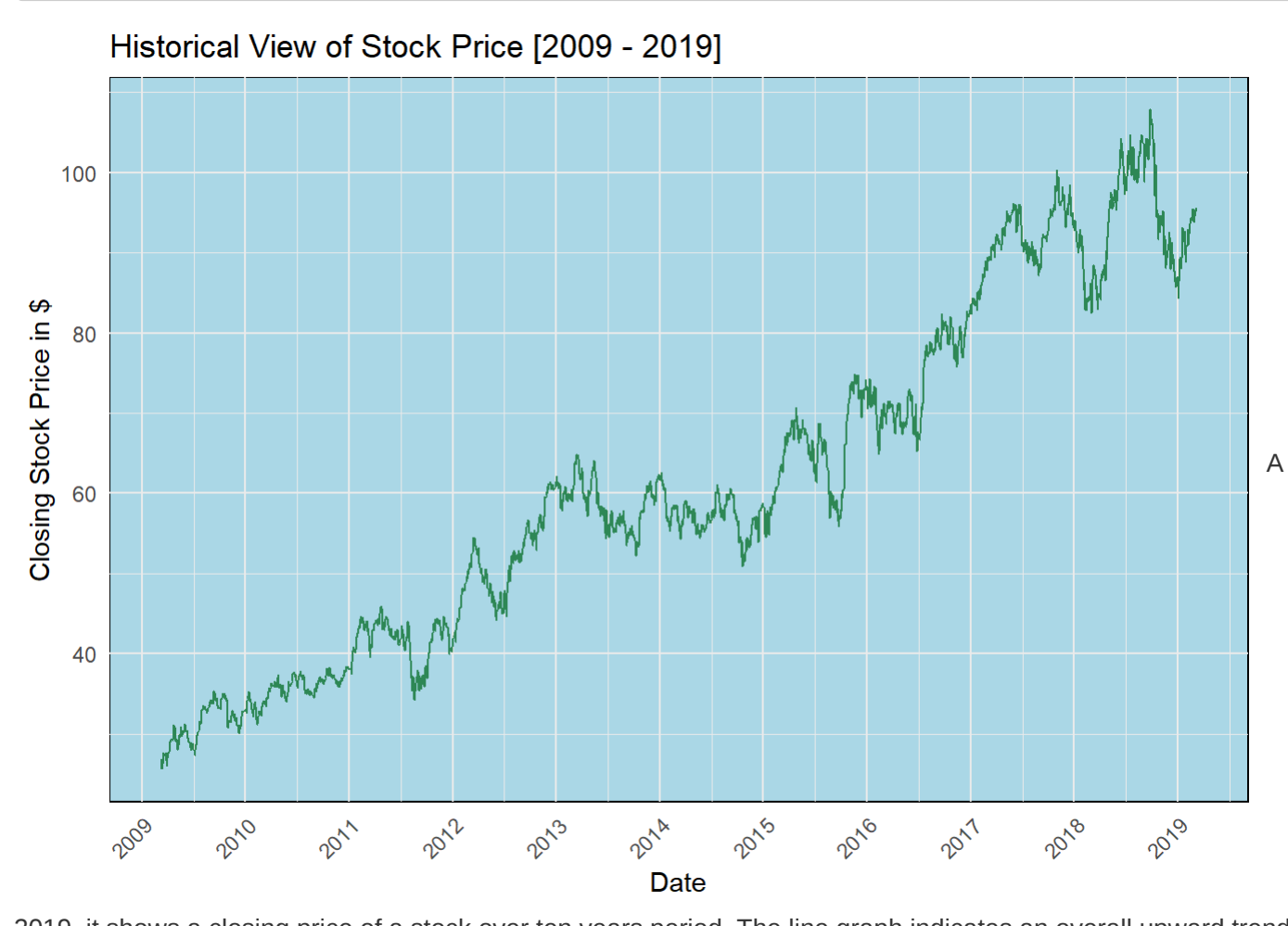
```
##      Date                Open             High             Low
##  Length:2550        Min.   : 25.16   Min.   : 25.82   Min.   : 24.48
##  Class :character   1st Qu.: 41.50   1st Qu.: 43.43   1st Qu.: 42.59
##  Mode  :character   Median : 56.56   Median : 58.48   Median : 57.58
##                     Mean   : 56.69   Mean   : 61.56   Mean   : 60.54
##                     3rd Qu.: 67.73   3rd Qu.: 78.36   3rd Qu.: 77.08
##                     Max.   :100.10   Max.   :108.52   Max.   :107.02
##                     NA's   :308      NA's   :7        NA's   :7
##      Close            Change       Traded.Volume         Turnover
##  Min.   : 25.59   Min.   :-0.740   Min.   :       0   Min.   :1.767e+05
##  1st Qu.: 42.95   1st Qu.:-0.500   1st Qu.: 2131686   1st Qu.:1.300e+08
##  Median : 58.02   Median :-0.290   Median : 2852772   Median :1.627e+08
##  Mean   : 61.00   Mean   :-0.070   Mean   : 3296818   Mean   :1.828e+08
##  3rd Qu.: 77.76   3rd Qu.: 0.085   3rd Qu.: 3878528   3rd Qu.:2.105e+08
##  Max.   :107.80   Max.   : 1.250   Max.   :36456707   Max.   :1.369e+09
##                   NA's   :2539     NA's   :46         NA's   :53
##  Last.Price.of.the.Day Daily.Traded.Units Daily.Turnover
##  Mode:logical          Mode:logical       Min.   :0
##  NA's:2550             NA's:2550          1st Qu.:0
##                                           Median :0
##                                           Mean   :0
##                                           3rd Qu.:0
##                                           Max.   :0
##                                           NA's   :2543
```

Perparing Data for Analysis

```r
#Select only Date and close
df <- subset(data, select = c("Date", "Close"))

#convert date to date format
df$Date<-as.Date(df$Date)

#Extract date and creat new column for year
df$Year <- format(df$Date, "%Y%")
```

A Line Graph Between Date and Close Price.

```r
ggplot(df, aes(x = Date, y = Close)) +
  geom_line(color = "seagreen") +  # Set line color to sea green
  labs(title = "Historical View of Stock Price [2009 - 2019]",
       x = "Date",
       y = "Closing Stock Price in $") +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y") +
  theme_minimal() +  # Use a minimal theme
  theme(
    panel.background = element_rect(fill = "lightblue"),  # Set background color
    axis.text.x = element_text(angle = 45, hjust = 1)  # Rotate x-axis labels
  )
```



A close stock price history 2009-2019. it shows a closing price of a stock over ten years period. The line graph indicates an overall upward trend in the stock price, suggesting that the value of the stock has generally increased over the decade.

Linear Regression

In our dataset, we have one key variable, the date, serving as the sole independent variable (X). Our aim is to predict the dependent variable (Y), which represents the Stock Price. To establish a relationship between X and Y, we opt for a Simple Linear Regression approach.

The equation of the best-fit line is given as:

$Y = \beta_0 + \beta_1 X$

Where: Y: Predicted value of the dependent variable β_0: Y-intercept β_1: Slope X: Value of the independent variable

Our objective is to determine the coefficients β_0 and β_1 such that the Sum of Squared Errors is minimized. This sum quantifies the disparity between each data point and its corresponding predicted value generated by the model.

Training Linear Regression Model

```r
# Split Train and Test Data
set.seed(123)
index <- createDataPartition(df$Close, p = 0.8, list = FALSE)

train <- df[index,]
test <- df[-index,]
```

Building Regression Model

```r
model <- lm(Close~Date, data=df)
```

Model Evaluation

```r
# coeffiecent slope
slope <- coef(model)["Date"]
print(paste("Slope:",slope))
```

```
## [1] "Slope: 0.0193271359045544"
```

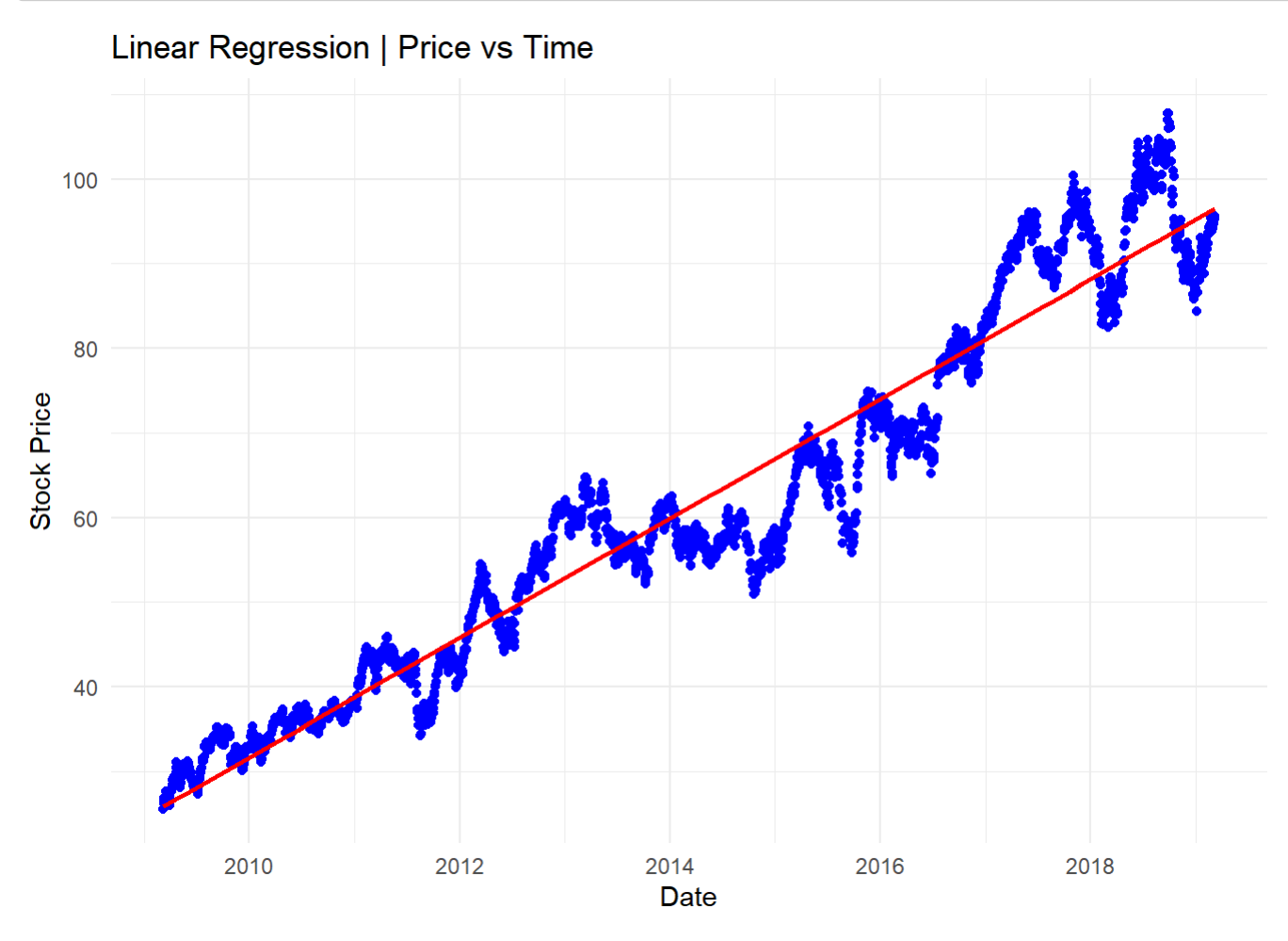
The slope value indicate the average change in the dependent variable (stock price) for one unit increase in the independent variable (Date). That is for every one unit increase in the date (assuming date is measured in a consistent unit such as day), the predicted stock price is expected to increase by approximately 0.0193. This implies that as time progresses, the stock price tends to increase.

```r
# The intercept
intercept <- coef(model)["(Intercept)"]
print(paste("intercept:", intercept))
```

```
## [1] "intercept: -250.7027325695"
```

The intercept value of approximately -250.70 represents the estimated value of the dependent variable when the independent variable is zero.

```r
# Create a scatter plot with regression line using ggplot2
ggplot(df, aes(x = Date, y = Close)) +
  geom_point(color = "blue") +  # Scatter plot with blue points
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE, color = "red") +  # Add regression line
  labs(x = "Date", y = "Stock Price", title = "Linear Regression | Price vs Time") +  # Add labels and title
  theme_minimal()  # Use a minimal theme
```



Prediction from the Model with Test Data

```r
# Prediction from the model
test$Date<-as.Date(test$Date)
prediction<- predict(model, newdata =  test)

# Predicted values
test$prediction <- prediction
```

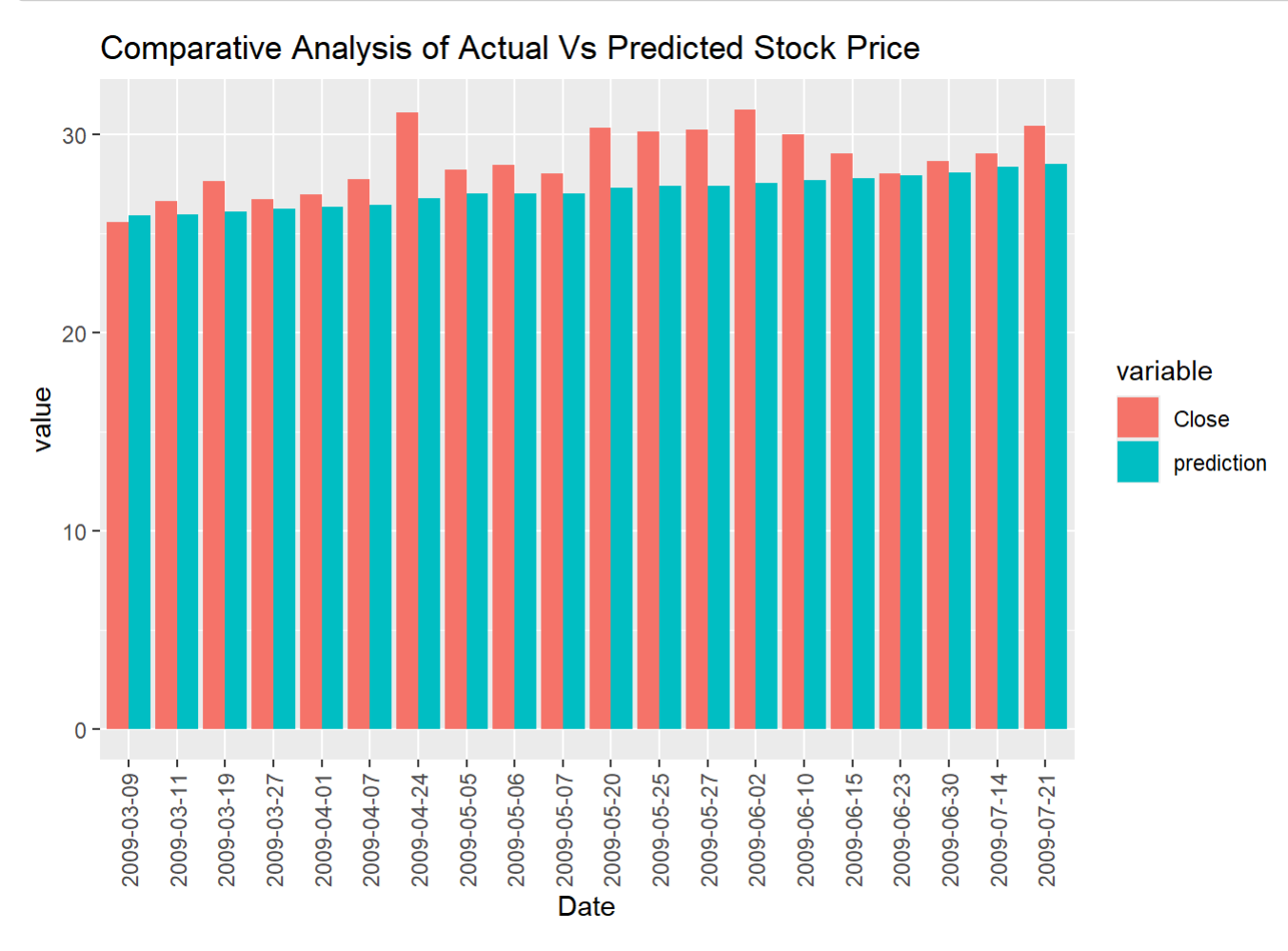
Regression Evaluation.

```r
head(test, 7)
```

```
##          Date Close  Year prediction
## 1  2009-03-09 25.59 2009%   25.90724
## 3  2009-03-11 26.64 2009%   25.94589
## 9  2009-03-19 27.63 2009%   26.10051
## 15 2009-03-27 26.75 2009%   26.25512
## 18 2009-04-01 27.00 2009%   26.35176
## 22 2009-04-07 27.77 2009%   26.46772
## 35 2009-04-24 31.12 2009%   26.79628
```

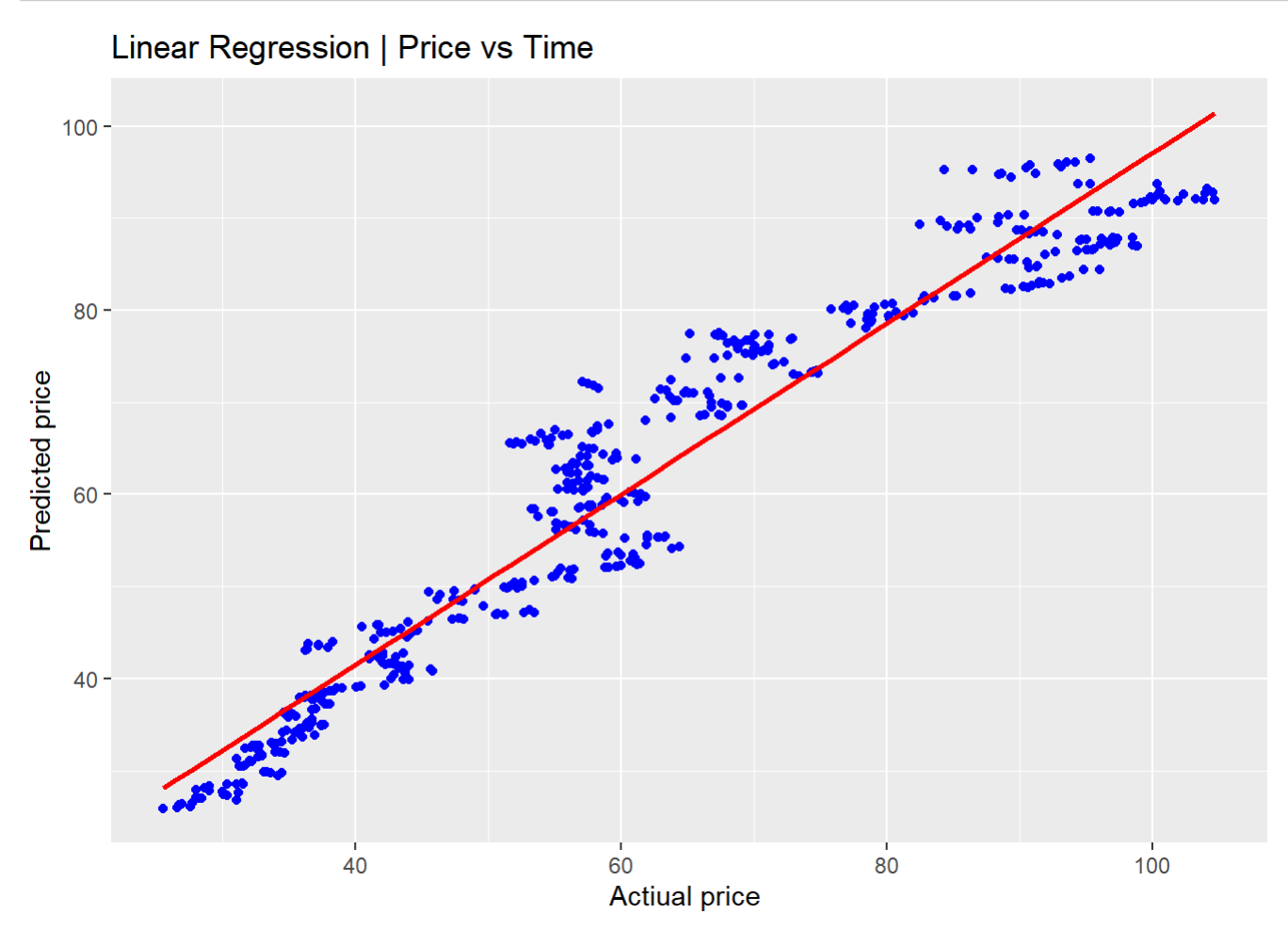
```r
df2<- head(test, 20) # select first 20 Rows
df2$Date<- as.factor(df2$Date) # convert date to factor
df2_long<-tidyr::pivot_longer(df2,cols= c("Close","prediction"),
                              names_to = "variable", values_to = "value")
```

```r
# Bar Plot For Comparative Analysis
ggplot(df2_long, aes(x=Date, y= value, fill = variable)) +
  geom_bar(stat = "identity",position = "dodge") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))+
  labs(title = "Comparative Analysis of Actual Vs Predicted Stock Price")
```



The bar graph represent a time series analysis comparing actual stock price (represented by blue bars) with predicted stock prices (depicted by green bar) over a series of dates. It appears that the predictions closely align with the actual prices, suggesting that the predictive model used for forecasting stock prices is quite accurate. Minor discrepancies between the actual and predicted values visible, which is common in predictive model due to various influencing factors that can cause fluctuations in stock prices.

```r
# plot fitted line  for test data
ggplot(test,aes(x=Close,y=prediction))+geom_point(color ="blue")+
  geom_smooth(method = "lm", formula = y~x,se = FALSE, color = "red") +
  labs(title ="Linear Regression | Price vs Time", x= "Actiual price ",
       y= "Predicted price")
```

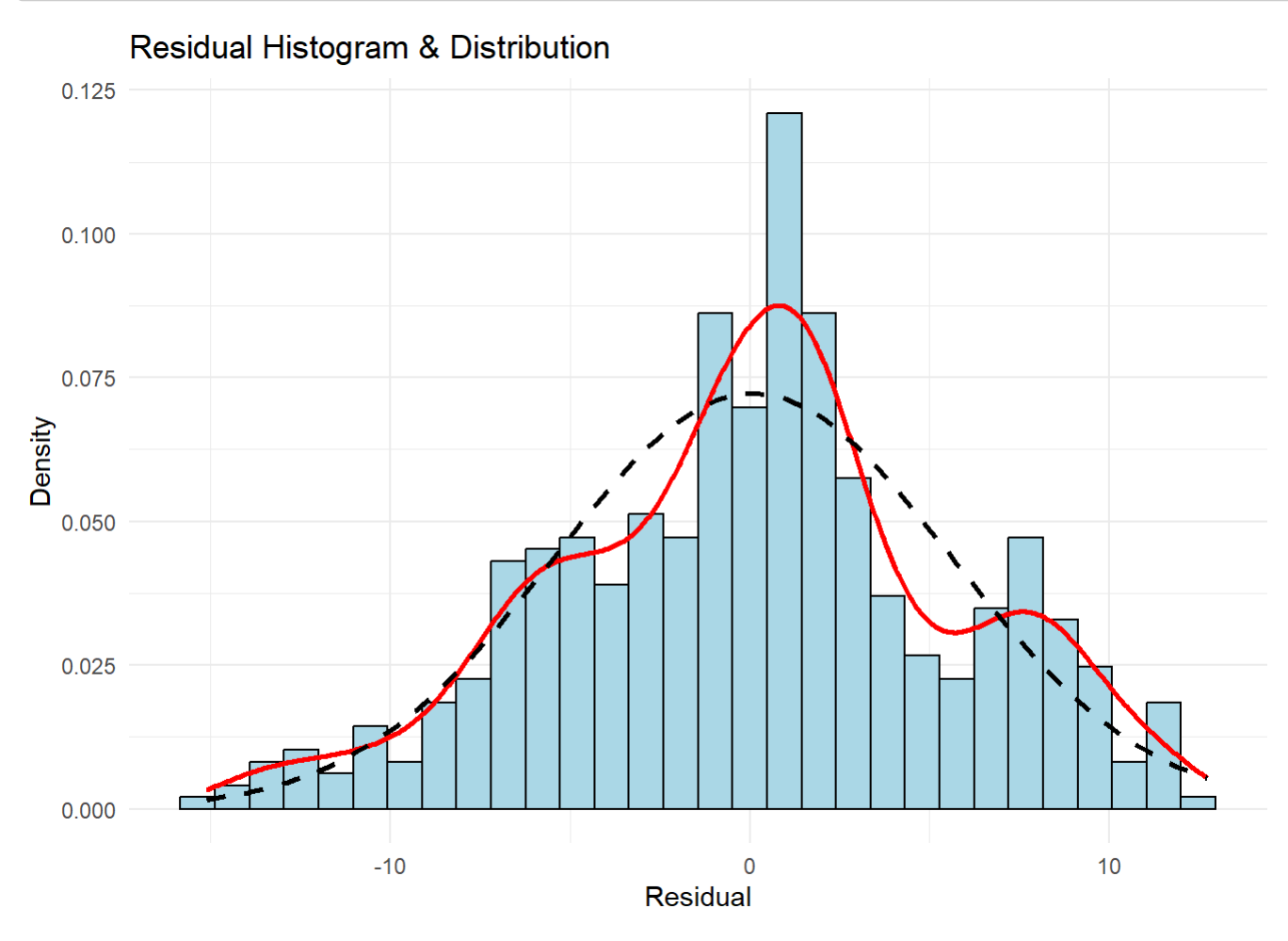


The scatter plot suggest a strong correlation between the two variables. The model predictions are within a similar scale to the actual price. This close alignment between the actual and predicted values signifies the model's effectivrness in forcasting prices over time. # Residual Histogram

```r
#calculating the residuals
residuals <- test$Close - test$prediction

#fit normal distribution to the residuals
fit<- fitdistr(residuals,"normal")

#create a dataframe for plotting
dff<- data.frame(Residual = residuals)

# Create histogram with fitted normal distribution using ggplot2
ggplot(dff, aes(x = Residual)) +
  geom_histogram(aes(y = ..density..), fill = "lightblue", color = "black", bins = 30) +
  geom_density(color = "red", size = 1) +
  labs(title = "Residual Histogram & Distribution", x = "Residual", y = "Density") +
  stat_function(fun = dnorm, args = list(mean = fit$estimate["mean"], sd = fit$estimate["sd"]), color = "black",
size = 1, linetype = "dashed") +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## ℹ Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



A graph shows the residual distribution of the dataset, a well fitting model would show residual that are symmetrically distributed around zero, indicating that the model's predictions are unbiased. The graph suggest that most residuals are close to zero which is a good sign of model accuracy. However there are some residuals further away from zero, which could indicate potential outlines, with some room for improvement

Error Evaluation Metrics.

```r
# Mean Absolute Error. (MAE)
MAE <- mean(abs(test$Close- test$prediction))
print(paste("Mean Absolute Error:",MAE))
```

```
## [1] "Mean Absolute Error: 4.32339791390662"
```

```r
# Meas Square Error

MSE <- mean((test$Close-test$prediction)^2)
print(paste("Mean Square Error:",MSE))
```

```
## [1] "Mean Square Error: 30.4623779212418"
```

```r
# Root Mean Square Error
RMSE<- sqrt(MSE)
print(paste("Root Mean Square Error:",RMSE))
```

```
## [1] "Root Mean Square Error: 5.51927331459874"
```

$Mean Absolute Error$ 4.32 represent the average absolute difference between the predicted values and the actual values. that is on avreage the models's prediction are off by 4.323 units from the actual values. $Mean Squared Error$ -30.462 measures the avreage of the squares of the errors, which is the average squared difference between the estimated values and the actual value. The model's prediction are on avreage 30.462 squared units aways from the actual values. $Root Mean Square Error$ -5.51 measure the standard deviation of the residuals, providing a measure of the spread of these error. A value of 5.51 is the standard deviation of the prediction error i

#Accuracy Evaluation Metrics

```r
# Extract the R  square value
R2<- summary(model)$r.squared
print(paste("R-squared (R2) score:", R2))
```

```
## [1] "R-squared (R2) score: 0.933605346437416"
```

r square values of 0.933 in a regression model indicates that approximately 93.3% of the variability in the dependent variable can be explain by the independent variable in the model.

Model summary

```r
summary(model)
```

```
##
## Call:
## lm(formula = Close ~ Date, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.2091  -3.5759   0.1272   3.3834  14.4797
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.507e+02  1.650e+00  -151.9   <2e-16 ***
## Date         1.933e-02  1.021e-04   189.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.437 on 2548 degrees of freedom
## Multiple R-squared:  0.9336, Adjusted R-squared:  0.9336
## F-statistic: 3.583e+04 on 1 and 2548 DF,  p-value: < 2.2e-16
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.