UNIVERSITAS
GADJAH MADA

Numerical Method Prediction

By Kreshnayogi Dava Berliansyach
22/496686/PA/21352

Dataset

Microsoft Stock Price Dataset

Prediction on the stock price based on a certain date

X : Date

Y : Adj Close

```
            Date        Open        High         Low       Close   Adj Close  \
0     1986-03-13    0.088542    0.101563    0.088542    0.097222    0.060055
1     1986-03-14    0.097222    0.102431    0.097222    0.100694    0.062199
2     1986-03-17    0.100694    0.103299    0.100694    0.102431    0.063272
3     1986-03-18    0.102431    0.103299    0.098958    0.099826    0.061663
4     1986-03-19    0.099826    0.100694    0.097222    0.098090    0.060591
...          ...         ...         ...         ...         ...         ...
9627  2024-05-24  427.190002  431.059998  424.410004  430.160004  430.160004
9628  2024-05-28  429.630005  430.820007  426.600006  430.320007  430.320007
9629  2024-05-29  425.690002  430.940002  425.690002  429.170013  429.170013
9630  2024-05-30  424.299988  424.299988  414.239990  414.670013  414.670013
9631  2024-05-31  416.410004  416.630005  404.519989  406.760010  406.760010

          Volume
0     1031788800
1      308160000
2      133171200
3       67766400
4       47894400
...          ...
9627    11845800
9628    15718000
9629    15517100
9630    28394500
9631    17190518

[9632 rows x 7 columns]
```

Methods

There are two methods used:

- Linear Regression
- Quadratic Regression

Data Split:

- 70% training
- 30% testing

```
           Date        Open        High         Low       Close   Adj Close  \
0    1986-03-13    0.088542    0.101563    0.088542    0.097222    0.060055
1    1986-03-14    0.097222    0.102431    0.097222    0.100694    0.062199
2    1986-03-17    0.100694    0.103299    0.100694    0.102431    0.063272
3    1986-03-18    0.102431    0.103299    0.098958    0.099826    0.061663
4    1986-03-19    0.099826    0.100694    0.097222    0.098090    0.060591
...         ...         ...         ...         ...         ...         ...
9627 2024-05-24  427.190002  431.059998  424.410004  430.160004  430.160004
9628 2024-05-28  429.630005  430.820007  426.600006  430.320007  430.320007
9629 2024-05-29  425.690002  430.940002  425.690002  429.170013  429.170013
9630 2024-05-30  424.299988  424.299988  414.239990  414.670013  414.670013
9631 2024-05-31  416.410004  416.630005  404.519989  406.760010  406.760010

          Volume
0     1031788800
1      308160000
2      133171200
3       67766400
4       47894400
...          ...
9627    11845800
9628    15718000
9629    15517100
9630    28394500
9631    17190518

[9632 rows x 7 columns]
```

Linear Regression Formulas

$$y_i = \alpha + \beta x_i$$

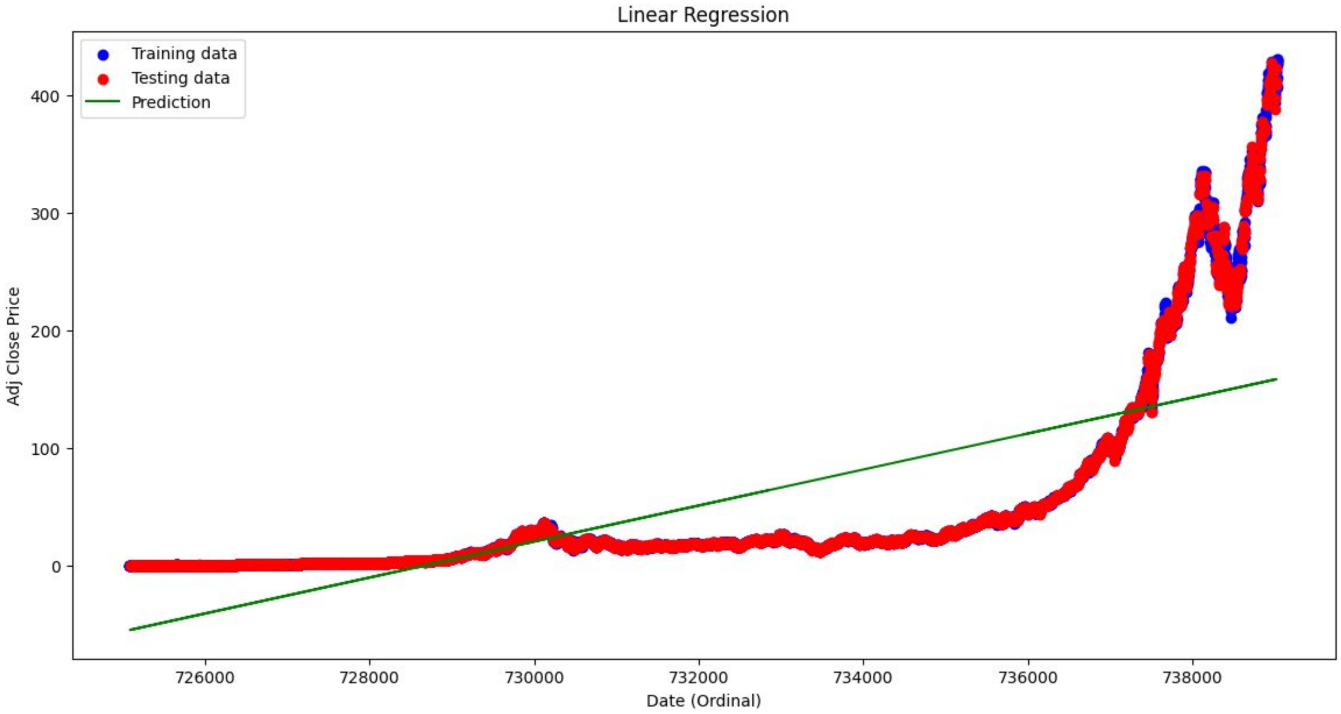
$$\widehat{\beta} = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$ (Slope)

$$\widehat{\alpha} = \bar{y} - \widehat{\beta}\,\bar{x} \,,$$ (Intercept)

Linear Regression Implementation

```python
#Linear Regression

#Calculate mean of x and y
X_mean = np.mean(X_train)
Y_mean = np.mean(Y_train)

#Calculate the slope and intercept
slope = np.sum((X_train - X_mean) * (Y_train - Y_mean))/np.sum((X_train - X_mean) ** 2)
intercept = Y_mean - (slope * X_mean)

print(f"Slope: {slope}")
print(f"Intercept: {intercept}")

#Make predictions
Y_P = slope * X_test + intercept
```

Linear Regression Result




```
Slope: 0.015312893186071326
Intercept: -11158.070969390556
```

Quadratic Regression Formulas

$$\hat{\beta} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}$$

$\hat{\beta}$ = ordinary least squares estimator

X = matrix regressor variable X

$\top$ = matrix transpose

y = vector of the value of the response variable

$$y = ax^2 + bx + c$$

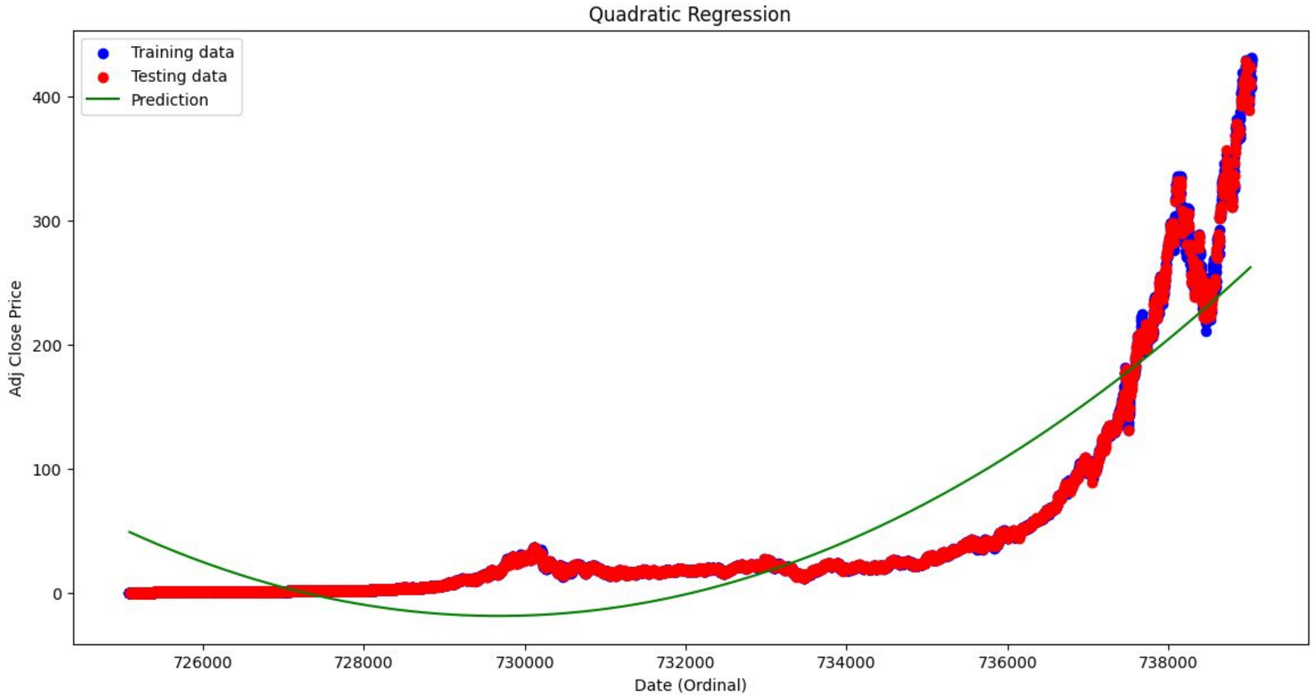
X Train Matrix

```
[[5.37262612e+11 7.32982000e+05 1.00000000e+00]
 [5.32619717e+11 7.29808000e+05 1.00000000e+00]
 [5.45896367e+11 7.38848000e+05 1.00000000e+00]
 ...
 [5.27635168e+11 7.26385000e+05 1.00000000e+00]
 [5.36693969e+11 7.32594000e+05 1.00000000e+00]
 [5.26232726e+11 7.25419000e+05 1.00000000e+00]]
```

Quadratic Regression Implementation

```python
#Quadratic Regression

#Matrix
X_train_matrix = np.vstack([X_train**2, X_train, np.ones(len(X_train))]).T

#Compute the coefficients
coefficients = np.linalg.inv(X_train_matrix.T @ X_train_matrix) @ X_train_matrix.T @ Y_train
a, b, c = coefficients

print(f"a: {a}")
print(f"b: {b}")
print(f"c: {c}")

#Make predictions
Y_P = a * X_test**2 + b * X_test + c
```

Quadratic Regression Result




```
a: 3.213350315022799e-06
b: -4.689389158764781
c: 1710841.1777268536
```

Evaluation Methods

- Mean Squared Error (MSE)

$$\mathrm{MSE} = \underbrace{\boxed{\frac{1}{n}\sum_{i=1}^{n}}}_{\text{Mean}} \underbrace{\boxed{(Y_i - \hat{Y}_i)}}_{\text{Error}}{}^{\boxed{2}\,\text{Squared}}$$

- R Squared

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Evaluation Implementation

```
#Evaluate
test_mse = np.mean((Y_test - Y_P) ** 2)
test_r2 = 1 - (np.sum((Y_test - Y_P) ** 2) / np.sum((Y_test - Y_mean) ** 2))

print(f"Testing MSE: {test_mse}")
print(f"Testing R^2: {test_r2}")
```

Evaluation Result

- Linear Regression

```
Testing MSE: 3601.107779876043
Testing R^2: 0.4965578756343469
```

- Quadratic Regression

```
Testing MSE: 1568.5803618697978
Testing R^2: 0.780664231488658
```

UNIVERSITAS
GADJAH MADA

Thank You