

AI-Driven Stock Market Forecasting Using Financial Statement Analysis

Description of the Collected Dataset

By

Marwa Othman Esaa

Midocean University

Advisor

Dr. Ashraf Elsayed

Data Collection

This dataset contains historical financial ratios for 100 Saudi companies listed on the Tadawul All Share Index (TASI). We selected 100 stocks, spanning from 2017Q1 to 2025Q1, encompassing a total of 33 quarters' worth of financial data.

```
df = pd.read_csv('ratio-2025-03-31 (2).csv')
df.head()
```

	CO_ID	Name	Year	Quarter	Sector	Current Ratio	Long Term Debt / Captial	Debt / Equity	Gross Margin	Operating Margin	...	Receivable Turnover	Days Sales In Receivables	ROE	ROTE	ROA	ROI	Va
0	2222	SAUDI ARAMCO	2017	1	Energy	1.770150	8.584252	9.396935	67.476005	28.866559	...	2.073818	35.422139	0.025939	0.026476	0.012890	0.024564	32
1	2222	SAUDI ARAMCO	2017	2	Energy	1.758013	8.584252	9.396935	67.476005	28.866559	...	2.073818	35.422139	0.025939	0.026476	0.012890	0.024564	32
2	2222	SAUDI ARAMCO	2017	3	Energy	1.771570	8.584252	9.396935	67.476005	28.866559	...	2.073818	35.422139	0.025939	0.026476	0.012890	0.024564	32
3	2222	SAUDI ARAMCO	2017	4	Energy	1.771570	8.584252	9.396935	67.476005	28.866559	...	2.073818	35.422139	0.025939	0.026476	0.012890	0.024564	32
4	2222	SAUDI ARAMCO	2018	1	Energy	2.085701	8.971310	9.855666	66.018124	31.253041	...	1.719902	36.640039	0.016489	0.017068	0.009429	0.015432	48

5 rows × 24 columns

```
import pandas as pd
rows, columns = df.shape
print(f"Number of Rows: {rows}")
```

Number of Rows: 8386

```
missing = df.isnull().sum()
missing = missing[missing > 0].sort_values(ascending=False)
print(missing)
```

```
Inventory Turnover      1480
Current Ratio           1456
Long Term Debt / Captial 1456
Debt / Equity           1456
Operating Margin        1456
Gross Margin            1456
Pre-Tax Profit Margin   1456
Net Profit Margin       1456
Asset Turnover          1456
Receivable Turnover     1456
Days Sales In Receivables 1456
ROI                     1456
ROTE                    66
ROE                     66
ROA                     66
Book Value Per Share    66
Operating Cash Flow Per Share 66
Free Case Flow Per Share 66
dtype: int64
```

```
df.columns = df.columns.str.strip()
print(df.columns.tolist())
```

```
['CO_ID', 'Name', 'Year', 'Quarter', 'Sector', 'Current Ratio', 'Long Term Debt / Capital', 'Debt / Equity', 'Gross Margin', 'Operating Margin', 'Pre-Tax Profit Margin', 'Net Profit Margin', 'Asset Turnover', 'Inventory Turnover', 'Receivable Turnover', 'Days Sales In Receivables', 'ROE', 'ROTE', 'ROA', 'ROI', 'Book Value Per Share', 'Operating Cash Flow Per Share', 'Free Case Flow Per Share', 'Stock Return']
```

```
df = pd.read_csv("ratio-2025-03-31 (2).csv", sep=",", encoding="utf-8-sig")
```

```
df.columns = df.columns.str.strip()
print(df.columns.tolist())
```

```
['CO_ID', 'Name', 'Year', 'Quarter', 'Sector', 'Current Ratio', 'Long Term Debt / Capital', 'Debt / Equity', 'Gross Margin', 'Operating Margin', 'Pre-Tax Profit Margin', 'Net Profit Margin', 'Asset Turnover', 'Inventory Turnover', 'Receivable Turnover', 'Days Sales In Receivables', 'ROE', 'ROTE', 'ROA', 'ROI', 'Book Value Per Share', 'Operating Cash Flow Per Share', 'Free Case Flow Per Share', 'Stock Return']
```

```
import numpy as np

numeric_cols = df.select_dtypes(include='number').columns

for row in range(len(df)):
    co_id = df.at[row, "CO_ID"] # now this will work correctly

    for col in range(len(df.columns)):
        col_name = df.columns[col]

        if col_name != "CO_ID" and col_name in numeric_cols:
            if pd.isna(df.iat[row, col]):
                company_rows = df[df["CO_ID"] == co_id].reset_index()

                try:
                    current_index = company_rows[company_rows['index'] == row].index[0]
                except IndexError:
                    continue

            # Previous 3 rows
            start = max(0, current_index - 3)
            prev_window = company_rows.loc[start:current_index - 1, col_name]
            prev_mean = prev_window.dropna().mean()

            if not np.isnan(prev_mean):
                df.iat[row, col] = prev_mean
            else:

            # Next 3 rows
            end = min(len(company_rows), current_index + 4)
            next_window = company_rows.loc[current_index + 1:end, col_name]
            next_mean = next_window.dropna().mean()

            if not np.isnan(next_mean):
                df.iat[row, col] = next_mean
```

