Bài tập lập trình 1: Data Understanding

Sinh viên thực hiện:

Bùi Thanh Dân - 23020342

Phạm Tiến Dũng - 23020345

Vũ Nguyên Đan - 23020351

▼ 1. Quan sát để hiểu doanh nghiệp và dữ liệu

1.1. Giới thiệu doanh nghiệp

Công ty Cổ phần Sữa Việt Nam (Vinamilk - mã cổ phiếu: VNM) là doanh nghiệp hàng đầu trong ngành sữa tại Việt Nam, với mạng lưới phân phối rộng khắp và sản phẩm đa dạng. VNM niêm yết trên HOSE từ năm 2006 và là một trong các cổ phiếu blue-chip có vốn hóa lớn nhất.

Hoạt động của công ty chịu ảnh hưởng mạnh từ:

- Giá nguyên liệu đầu vào (sữa bột, thức ăn chăn nuôi).
- Nhu cầu tiêu dùng nội địa và xuất khẩu.
- Biến động kinh tế vĩ mô và tỷ giá ngoại tệ.

1.2. Dữ liệu

Sử dụng hai bảng dữ liệu được cung cấp:

- **Simplize_VNM_PriceHistory_20250315.xlsx:** Lịch sử giá cổ phiếu theo ngày.
- **Simplize_VNM_FinancialIndicator_20250315.xlsx:** Chỉ số tài chính theo quý.

Dữ liệu bao phủ giai đoạn **2021–2025**, gồm khoảng **1.000 dòng dữ liệu** cho lịch sử giá và 4**0 quý** cho các chỉ số tài chính.

1.3. Đọc dữ liệu và tiền xử lý đơn giản

Tiền xử lý dữ liệu đơn giản bằng 3 hàm sau:

- `normalize_text`: Hàm này làm sạch một chuỗi text bằng cách: loại bỏ dấu (diacritics), chuyển thành chữ thường, và xóa các khoảng trắng hoặc ký tự xuống dòng thừa.
- `to_number`: Hàm này cố gắng biến một chuỗi (ví dụ "1,000.50" hoặc "50%") thành một số float. Nó xử lý linh hoạt các dấu phẩy, chấm, ký hiệu phần trăm (%), và trả về `NaN` (Not a Number) nếu không thể chuyển đổi.
- `infer_header_row`: Hàm này duyệt qua DataFrame để tìm xem hàng nào chứa các tên cột mong đợi (expected_keys). Nó trả về chỉ số (index) của hàng đó.

```
def normalize_text(s: str) → str:
    if not isinstance(s, str):
        return s
    s = unicodedata.normalize("NFKD", s)
    s = "".join(ch for ch in s if not unicodedata.combining(ch))
    s = s.lower().strip()
    s = s.replace("\n", " ").replace("\r", " ")
    while "  " in s:
        s = s.replace("  ", " ")
    return s

def to_number(x):
    if isinstance(x, (int, float, np.number)):
        return float(x)
    if not isinstance(x, str):
        return np.nan
    s = x.strip().replace(" ", "")
    if s == "":
        return np.nan
    allowed = set("0123456789-.,%")
    s = "".join(ch for ch in s if ch in allowed)
    pct = s.endswith("%")
    if pct:
        s = s[:-1]

    if "," in s and "." in s:
        s = s.replace(",", "")
    elif "," in s and "." not in s:
        s = s.replace(".", "").replace(",", ".")
    elif s.count(".") > 1:
        s = s.replace(".", "")
    try:
        val = float(s)
        return val/100 if pct else val
    except:
        return np.nan

def infer_header_row(df_raw: pd.DataFrame, expected_keys: List[str]) → int:
    keys_norm = [normalize_text(k) for k in expected_keys]
    for i, row in df_raw.iterrows():
        row_texts = [normalize_text(str(x)) for x in row.tolist()]
        if any(any(k in cell for k in keys_norm) for cell in row_texts):
            return i
    return 0
```

Đọc dữ liệu báo cáo tài chính:

```
financial_raw = pd.read_excel(FINANCIAL_REPORT_PATH, sheet_name=0, header=None)

header_row = infer_header_row(financial_raw, ["chi tieu", "q1", "q2", "q3", "q4"])
financial = financial_raw.iloc[header_row + 1:].copy()
financial.columns = financial_raw.iloc[header_row].tolist()
financial = financial.reset_index(drop=True)

financial.columns = [normalize_text(str(c)).strip() for c in financial.columns]
for col in financial.columns:
    numeric_ratio = financial[col].apply(
        lambda x: isinstance(x, (int, float, np.number)) or
                  (isinstance(x, str) and any(ch.isdigit() for ch in x))
    ).mean()
    if numeric_ratio > 0.5:
```

```
        financial[col] = financial[col].apply(to_number)
financial = financial.dropna(how="all").reset_index(drop=True)

financial.head(5)
```

chi tieu	q4/2024	q3/2024	q2/2024	q1/2024	...
Chỉ tiêu Báo cáo kết quả kinh doanh (Tỷ đồng)	**NaN**	**NaN**	**NaN**	**NaN**	...
Doanh thu thuần	15,477,073,125,441.0000	15,537,337,313,473.0000	16,655,787,772,473.0000	14,112,411,317,058.0000	...
Tăng trưởng doanh thu	-0.0091	-0.0064	0.0961	0.0139	...
Lợi nhuận gộp	6,209,690,844,985.0000	6,401,445,454,280.0000	7,067,518,779,284.0000	5,911,521,444,565.0000	...
Tăng trưởng lợi nhuận gộp	-0.0360	-0.0234	0.1492	0.0951	...

Đọc dữ liệu lịch sử giá:

```
price_raw = pd.read_excel(PRICE_HISTORY_PATH, sheet_name=0, header=None)

header_row_price = infer_header_row(price_raw, ["gia mo cua", "gia cao nhat", "gia thap nhat", "gia dong cua", "thay doi gi
a"])
price = price_raw.iloc[header_row_price + 1:].copy()
price.columns = price_raw.iloc[header_row_price].tolist()
price = price.reset_index(drop=True)

price.columns = [normalize_text(str(c)).strip() for c in price.columns]
price['ngay'] = pd.to_datetime(price['ngay'], dayfirst=True, errors="coerce")
for col in price.columns:
    if col != "ngay":
        price[col] = price[col].apply(to_number)
price.head(5)
```

ngay	gia mo cua	gia cao nhat	gia thap nhat	gia dong cua	thay doi gia	% thay doi	...
2025-03-14	62,400.0000	62,700.0000	62,100.0000	62,100.0000	100.0000	0.0016	...
2025-03-13	62,300.0000	62,900.0000	61,900.0000	62,000.0000	-200.0000	-0.0032	...
2025-03-12	62,500.0000	62,800.0000	62,200.0000	62,200.0000	-200.0000	-0.0032	...
2025-03-11	62,300.0000	62,500.0000	62,100.0000	62,400.0000	-100.0000	-0.0016	...
2025-03-10	63,000.0000	63,000.0000	62,500.0000	62,500.0000	-200.0000	-0.0032	...

1.4. Thống kê miêu tả

Sử dụng hàm `descriptive_summary` để tính toán các chỉ số:

```
def descriptive_summary(df: pd.DataFrame) → pd.DataFrame:
    num = df.select_dtypes(include=[np.number])
    return pd.DataFrame({
        "count": num.count(),
        "min":   num.min(),
        "Q1":    num.quantile(0.25),
        "median(Q2)": num.median(),
        "Q3":    num.quantile(0.75),
        "max":   num.max(),
        "mean":  num.mean(),
        "std":   num.std(ddof=1),
        "var":   num.var(ddof=1),
    })
```

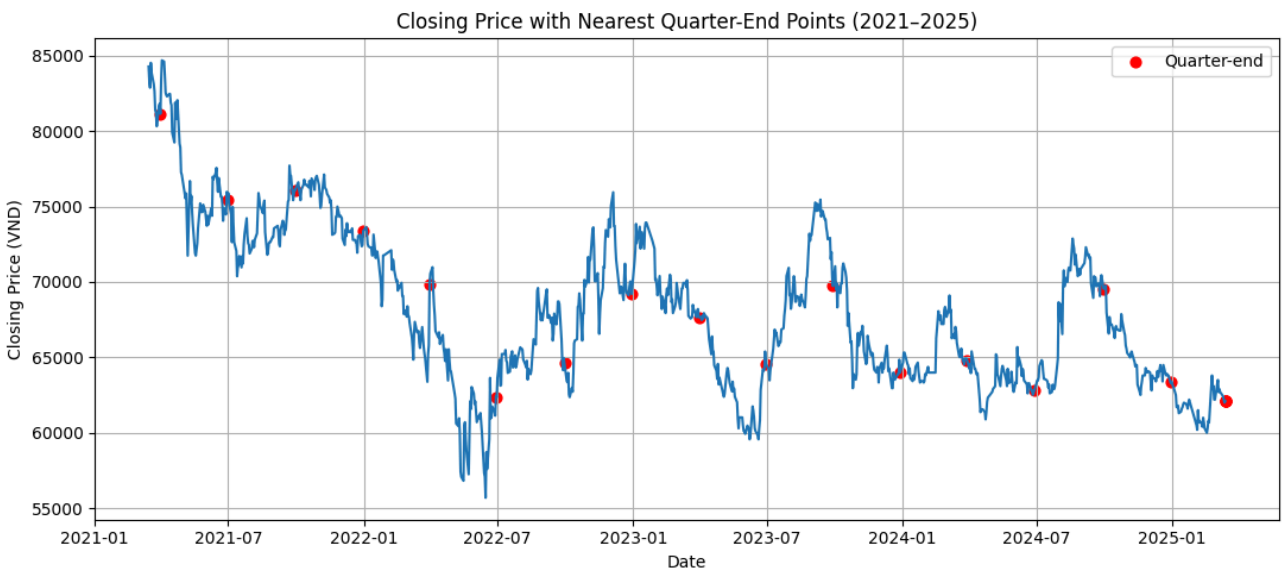
Kết quả:

	count	min	Q1	median(Q2)	Q3	max	...
gia mo cua	1000	56,214.8111	64,018.1539	67,571.0007	72,110.8986	86,174.4921	...
gia cao nhat	1000	57,423.7318	64,495.3416	68,099.1461	72,710.8525	86,174.4921	...
gia thap nhat	1000	55,696.7022	63,611.6482	67,037.3783	71,477.9757	84,277.8287	...
gia dong cua	1000	55,696.7022	63,992.7177	67,571.0007	72,051.6628	84,690.1468	...
thay doi gia	547	-2,500.0000	-500.0000	-100.0000	400.0000	3,900.0000	...
% thay doi	547	-0.0338	-0.0065	-0.0013	0.0055	0.0576	...
khoi luong	1000	691,300.0000	1,932,900.0000	2,702,950.0000	3,882,825.0000	21,564,900.0000	...

	count	min	Q1	median(Q2)	Q3	...
q4/2024	52	-0.4482	0.1012	1.6198	842,246,249,439.0000	...
q3/2024	53	-0.5717	0.1547	1.6267	411,532,810,496.0000	...
q2/2024	53	-0.4239	0.2081	2.0001	453,196,849,401.0000	...
q1/2024	53	-0.4246	0.1820	1.9460	491,691,149,713.0000	...
...	...	...	...	...	...	...

1.5. Theo dõi sự biến động của giá cổ phiếu

Qua một vài bước tính toán nhỏ, sử dụng thư viện matplotlib trong python, thu được biểu đồ biến động giá đóng cửa của cổ phiếu như sau:



Biểu đồ cho thấy hai giai đoạn chính:

1. **Giai đoạn 1 (Đầu 2021 - Giữa 2022): Xu hướng giảm (Downtrend) rõ rệt.**
   - Giá bắt đầu ở mức đỉnh rất cao (khoảng 85,000 VND) vào đầu năm 2021.
   - Sau đó, giá liên tục giảm mạnh và kéo dài, tạo ra các đỉnh sau thấp hơn đỉnh trước và đáy sau thấp hơn đáy trước.
   - Nguyên nhân chủ yếu do kết quả kinh doanh suy giảm, bị tác động bởi chi phí nguyên liệu tăng và sự cạnh tranh khốc liệt nội địa.
2. **Giai đoạn 2 (Giữa 2022 - Đầu 2025): Biến động đi ngang (Sideways Volatility).**
   - Sau khi chạm đáy (khoảng 56,000 - 58,000 VND) vào giữa năm 2022, giá không tiếp tục giảm sâu.
   - Thay vào đó, giá dao động trong một biên độ rộng, chủ yếu từ 60,000 VND đến khoảng 75,000 VND.
   - Trong giai đoạn này, có nhiều đợt tăng và giảm ngắn hạn, cho thấy sự giằng co giữa bên mua và bên bán mà không có bên nào hoàn toàn chiếm ưu thế để tạo ra một xu hướng mới.
   - Trong giai đoạn này, nhờ vào sự cải thiện về biên lợi nhuận, chiến lược tái cấu trúc hệ thống phân phối và danh mục sản phẩm, tăng trưởng doanh thu xuất khẩu, cũng như triển vọng tích cực từ sự phục hồi của GDP và nhu cầu tiêu dùng, giá cổ phiếu có xu hướng ổn định hơn.

▼ 2. Tiền xử lý dữ liệu

Mục tiêu của phần này là tạo ra một bảng dữ liệu mới, duy nhất từ hai nguồn dữ liệu đã cho (`Simplize_VNM_PriceHistory` và `Simplize_VNM_FinancialIndicator`). Bảng dữ liệu mới này sẽ có tần suất theo quý và chứa các chỉ số tài chính cùng với giá cổ phiếu đã được xử lý tương ứng, sẵn sàng cho việc phân tích và xây dựng mô hình ở các bước tiếp theo.

Quá trình này bao gồm các bước chính sau:

2.1. Tạo bảng dữ liệu mới từ hai bảng đã cho

2.1.1 Chuyển đổi định dạng dữ liệu tài chính

Dữ liệu tài chính ban đầu có các chỉ số ở dạng hàng và các quý ở dạng cột. Để có thể kết hợp với dữ liệu giá theo chuỗi thời gian, chúng ta cần chuyển đổi bảng dữ liệu này sao cho mỗi hàng đại diện cho một quý duy nhất và mỗi cột là một chỉ số tài chính.

```
financial_long = financial.set_index('chi tieu')
financial_long = financial_long.transpose()
financial_long = financial_long.reset_index()
financial_long = financial_long.rename(columns={'index': 'quarter_str'})
```

2.1.2. Chuẩn hóa và chuyển đổi cột thời gian

Cột thời gian (quarter_str) đang ở định dạng chuỗi ký tự (ví dụ: 'q4/2024'). Chúng ta cần chuyển đổi nó sang định dạng datetime chuẩn để có thể thực hiện các phép toán thời gian. Hàm `parse_quarter` được tạo để phân tích chuỗi này và trả về ngày cuối cùng của quý tương ứng.

```
def parse_quarter(q_str):
    try:
        quarter, year = q_str.split('/')
        quarter_num = int(quarter[1])
        month = quarter_num * 3
        day = pd.Timestamp(year=int(year), month=month, day=1).days_in_month
        return pd.to_datetime(f'{year}-{month}-{day}')
    except:
        return None

financial_long['quarter'] = financial_long['quarter_str'].apply(parse_quarter)
financial_long = financial_long.dropna(subset=['quarter']).reset_index(drop=True)
```

2.1.3. Tổng hợp dữ liệu giá cổ phiếu theo quý

Theo yêu cầu của đề bài, giá cổ phiếu của một quý không phải là giá của một ngày duy nhất mà là giá trị trung bình của giá đóng cửa trong một cửa sổ thời gian **±14 ngày** xung quanh ngày kết thúc quý (tức là một khoảng 29 ngày). Điều này giúp làm mượt dữ liệu giá và giảm thiểu các biến động bất thường trong ngắn hạn.

Chúng ta lặp qua mỗi ngày cuối quý, xác định cửa sổ thời gian tương ứng, lọc ra các phiên giao dịch trong khoảng đó và tính giá đóng cửa trung bình.

```
price = price.sort_values('ngay').reset_index(drop=True)
quarter_end_dates = financial_long['quarter'].unique()
quarterly_prices = []

for q_date in quarter_end_dates:
    start_date = q_date - pd.Timedelta(days=14)
    end_date = q_date + pd.Timedelta(days=14)
    mask = (price['ngay'] >= start_date) & (price['ngay'] <= end_date)
    price_window = price.loc[mask]

    if not price_window.empty:
        avg_price = price_window['gia dong cua'].mean()
    else:
        avg_price = np.nan

    quarterly_prices.append({
        'quarter': q_date,
        'gia dong cua': avg_price
    })
```

```
price_quarterly = pd.DataFrame(quarterly_prices)
```

2.1.4. Kết hợp dữ liệu tài chính và dữ liệu giá

Sau khi có hai bảng dữ liệu theo cùng một tần suất quý, chúng ta sử dụng hàm pd.merge để kết hợp chúng thành một DataFrame duy nhất (`final_df`). Phép hợp nhất này được thực hiện dựa trên cột quarter chung, đảm bảo mỗi dòng trong bảng cuối cùng chứa cả thông tin tài chính và giá cổ phiếu của cùng một quý.

```
final_df = pd.merge(financial_long, price_quarterly, on='quarter', how='inner')
```

2.1.5. Tính toán lại các chỉ số biến động giá

Vì giá cổ phiếu đã được tổng hợp lại theo quý, các cột `thay doi gia` và `% thay doi` ban đầu không còn chính xác. Chúng ta cần tính toán lại các giá trị này dựa trên giá đóng cửa trung bình của quý mới.

```
final_df = final_df.sort_values('quarter', ascending=True).reset_index(drop=True)
final_df['thay doi gia'] = final_df['gia dong cua'].diff()
final_df['% thay doi'] = final_df['gia dong cua'].pct_change()

first_cols = ['quarter', 'gia dong cua', 'thay doi gia', '% thay doi']
other_cols = [col for col in final_df.columns if col not in first_cols and col != 'quarter_str']
final_df = final_df[first_cols + other_cols]

final_df = final_df.sort_values('quarter', ascending=False).reset_index(drop=True)
```

2.2. Làm sạch và chuẩn hóa dữ liệu

2.2.1. Chuẩn hóa tên cột

Tên các cột chỉ số tài chính chứa dấu tiếng Việt và các ký tự đặc biệt. Để thuận tiện cho việc truy cập và xử lý sau này, chúng ta chuẩn hóa tất cả tên cột về dạng chữ thường không dấu, không có ký tự đặc biệt bằng hàm `clean_col_name`.

```
def clean_col_name(col_name):
    s = unicodedata.normalize('NFKD', col_name).encode('ascii', 'ignore').decode('utf-8')
    s = s.lower()
    s = re.sub(r'[^a-z0-9]+', ' ', s)
    s = s.strip(' ')
    return s

final_df.columns = [clean_col_name(col) for col in final_df.columns]
```

2.2.2. Xử lý các giá trị bị thiếu

Một số chỉ số tài chính có thể không được báo cáo trong một vài quý, dẫn đến dữ liệu bị thiếu. Các cột có tỷ lệ thiếu quá cao (ở đây là 90%) sẽ không mang lại nhiều giá trị cho phân tích và có thể gây lỗi. Vì vậy, chúng ta sẽ xác định và loại bỏ những cột này.

```
mode = "threshold"
threshold = 0.90
final_df = final_df.replace(r"^\s*$", np.nan, regex=True)
missing_frac = final_df.isna().mean()
cols_to_drop = missing_frac[missing_frac >= threshold].index.tolist()

if cols_to_drop:
    final_df = final_df.drop(columns=cols_to_drop, errors="ignore")
```

2.2.3. Chuyển đổi kiểu dữ liệu sang dạng số

Dữ liệu tài chính thường chứa các ký tự định dạng như dấu phẩy (,) để ngăn cách hàng nghìn hoặc ký tự phần trăm (%). Các ký tự này khiến cột dữ liệu được đọc vào dưới dạng chuỗi (object) thay vì dạng số, làm cho việc tính toán không thể thực hiện được. Chúng ta xây dựng một hàm để "ép" các cột này về đúng định dạng số (float), xử lý cả dấu phẩy và ký tự phần trăm.

```
def force_numeric_fixed(df):
    df2 = df.copy()
    for col in df2.columns:
        if col in ['quarter', 'quarter_str']:
            continue
        s = df2[col].astype(str)
        s = s.str.replace(",", "", regex=False).str.replace(" ", "", regex=False)
        is_percent = s.str.endswith("%")
        s = s.str.replace("%", "", regex=False)
        s = pd.to_numeric(s, errors="coerce")
        s[is_percent] = s[is_percent] / 100.0
        df2[col] = s
    return df2

final_df = force_numeric_fixed(final_df)
```

2.3. Kết quả cuối cùng

Sau khi thực hiện tất cả các bước trên, chúng ta thu được một DataFrame `final_df` hoàn chỉnh, sạch sẽ và có cấu trúc. Dữ liệu này đã sẵn sàng cho các bước phân tích thống kê, trực quan hóa và xây dựng mô hình ở phần 3.

Dưới đây là 5 dòng và 14 cột đầu tiên của bảng dữ liệu mới sau khi đã xử lý:

quarter	gia dong cua	thay doi gia	thay doi	doanh thu thuan	tang truong doanh thu	...
2024-12-31	63,089.0053	-5,936.2642	-0.0061	15,477,873,125,441.0000	-0.0091	...
1	60,946.0701	-5,529.9281	0.0072	15,537,337,313,473.0000	-0.0064	...
2024-06-30	63,416.1420	-1,265.7336	-0.0196	16,656,787,772,473.0000	0.0961	...
3	64,681.0759	533.0565	0.0083	14,112,411,317,058.0000	0.0139	...
2023-12-31	64,148.0194	-7,060.6902	-0.0992	15,618,710,944,490.0000	0.0365	...

▼ 3. Lựa chọn các yếu tố nguy cơ (risk factors) tiềm năng cho mô hình tài chính

Mục tiêu của phần này là **x**ác định các yếu tố (chỉ số tài chính nội tại) có mối quan hệ đáng kể với **giá cổ phiếu trung bình theo quý** bằng **hệ số tương quan Pearson**.

3.1 Tóm tắt phương pháp

1. Dùng `final_df` (tạo ở phần 2) chứa: `quarter`, `gia dong cua` (giá trung bình quý), các chỉ số tài chính chuyển về dạng số. (Cách lấy giá quý: avg ngày cuối quý ±14 ngày).
2. Tính ma trận tương quan `corr = final_df.corr(numeric_only=True)` giữa `gia dong cua` và các chỉ số.
3. Vẽ **heatmap** của ma trận tương quan (bằng Seaborn).
4. Liệt kê **top 5** chỉ số tương quan dương mạnh nhất và **top 5** tương quan âm mạnh nhất so với `gia dong cua`.

3.2 Tính hệ số tương quan

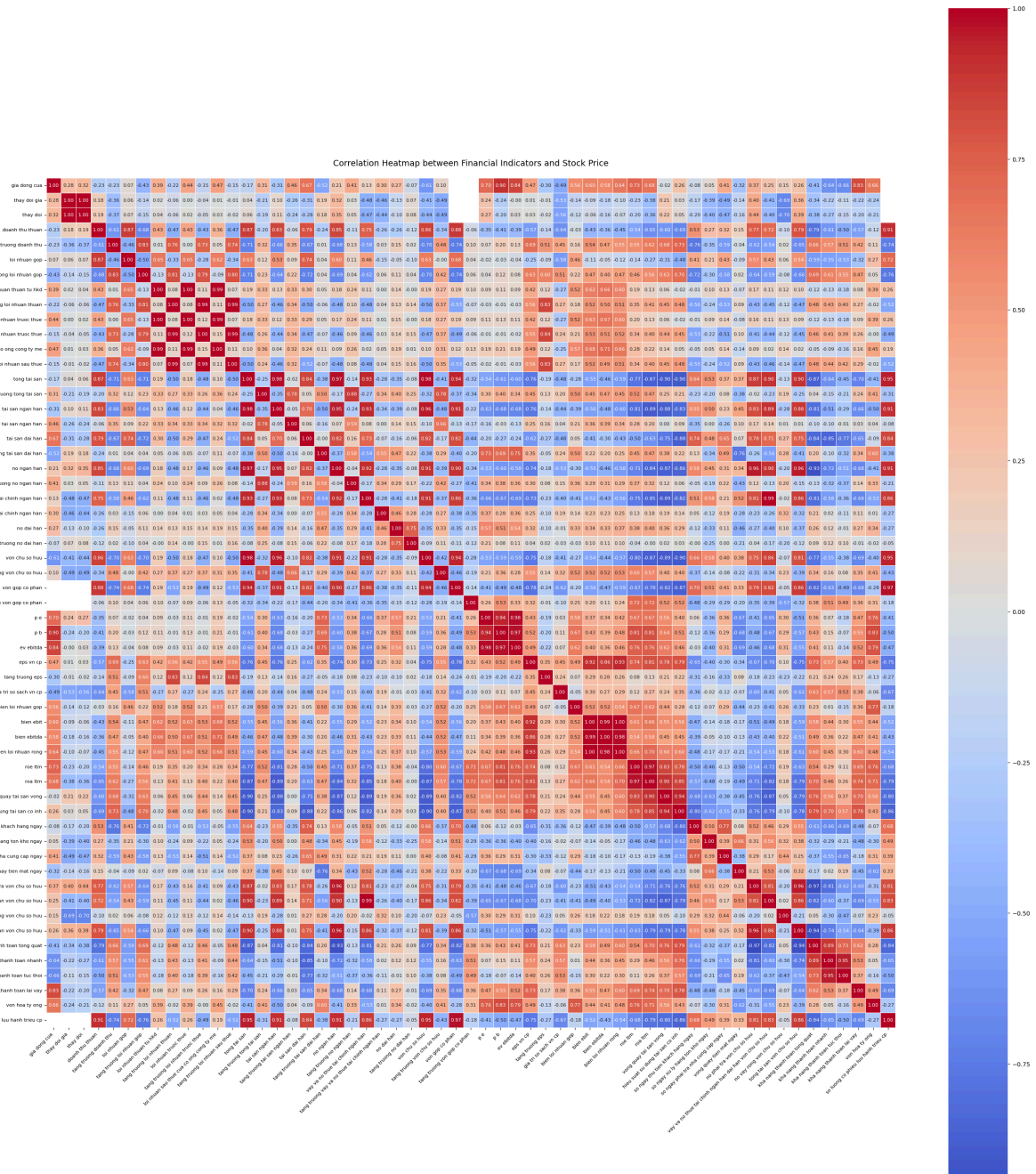
3.2.1. Vẽ correlation heatmap

Từ DataFrame đã tạo ở phần 2, vẽ correlation heatmap:

```
corr = final_df.corr(numeric_only=True)

n_cols = corr.shape[1]
fig_width = max(12, n_cols * 0.5)
fig_height = fig_width

plt.figure(figsize=(fig_width, fig_height))
sns.heatmap(
    corr,
    annot=True,
    fmt=".2f",
```

```
    cmap="coolwarm",
    cbar=True,
    annot_kws={"size": 8},
    linewidths=0.5,
    square=True
)

plt.title("Correlation heatmap between financial indicators and stock price", fontsize=14, pad=20)
plt.xticks(rotation=45, ha='right', fontsize=8)
plt.yticks(rotation=0, fontsize=8)
plt.tight_layout()
plt.show()
```

Ta thu được kết quả như sau:



Correlation heatmap between financial indicators and stock price

3.2.2 Liệt kê 5 cột tương quan dương và 5 cột tương quan âm mạnh nhất

```
corr_price = corr['gia dong cua'].sort_values(ascending=False)
top_positive = corr_price[corr_price.index != 'gia dong cua'].head(5)
top_negative = corr_price[corr_price.index != 'gia dong cua'].tail(5)

print("\nTop 5 positive correlations with stock price:")
display(top_positive)

print("\nTop 5 negative correlations with stock price:")
display(top_negative)
```

Ta được kết quả như sau:

▼ 5 tương quan **dương** mạnh nhất:

	gia dong cua
p b	0.8981
ev ebitda	0.8450
kha nang thanh toan lai vay	0.8250
roe ltm	0.7269
p e	0.7019

▼ 5 tương quan **âm** mạnh nhất:

	gia dong cua
kha nang thanh toan nhanh	-0.6393
kha nang thanh toan tuc thoi	-0.6577
von gop co phan	NaN
tang truong von gop co phan	NaN
so luong co phieu luu hanh trieu cp	NaN

Tại sao lại có tận 3 cột NaN? Chẳng lẽ dữ liệu chúng ta chuẩn hoá có vấn đề, hoặc phương pháp của chúng ta sai?

Check lại bằng đoạn code sau:

```
cols_to_check = ['gia dong cua', 'von gop co phan',
                 'tang truong von gop co phan',
                 'so luong co phieu luu hanh trieu cp']

for c in cols_to_check[1:]:
    valid = final_df[['gia dong cua', c]].dropna()
    print(f"\n→ {c}: {len(valid)} overlapping non-null rows")
    print(final_df[[c]].isna().sum(), "NaNs total in column")
```

Output:

```
→ von gop co phan: 16 overlapping non-null rows
von gop co phan    0
dtype: int64 NaNs total in column

→ tang truong von gop co phan: 2 overlapping non-null rows
tang truong von gop co phan    22
dtype: int64 NaNs total in column

→ so luong co phieu luu hanh trieu cp: 16 overlapping non-null rows
so luong co phieu luu hanh trieu cp    0
dtype: int64 NaNs total in column
```

Ta suy luận ra một cách giải thích như sau:

> 💡 Với cột `von gop co phan` và cột `so luong co phieu luu hanh trieu cp`, chúng có các giá trị là hằng số, lần lượt là `20899554450000` và `2089955445`, có nghĩa là không có sự biến thiên (variation).
> Chúng ta biết rằng hệ số tương quan không xác định nếu một trong hai biến có độ lệch chuẩn bằng 0 (dẫn đến lỗi chia cho 0).
> $r_{XY} = \frac{\mathrm{Cov}(X,Y)}{\sigma_X \cdot \sigma_Y} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\cdot\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$
> Thêm vào đó, các công ty hiếm khi thay đổi vốn điều lệ hàng quý, vì vậy điều này phần nào có thể đoán trước được.
> Còn với cột `tang truong von gop co phan`, chỉ với 2 hàng có giá trị (non-null) trùng lặp, pandas không thể tính toán được hệ số tương quan Pearson có ý nghĩa cho cột này.

3.3 Giá cổ phiếu của công ty này có thể có mối quan hệ phụ thuộc vào những chỉ số tài chính nào?

Các chỉ báo có tương quan dương mạnh

Chỉ báo	Tương quan	Giải thích
P/B (Tỷ lệ Giá trên Giá trị Sổ sách)	**+0.8981**	Tỷ lệ P/B cao cho thấy kỳ vọng thị trường lớn so với giá trị sổ sách. Khi giá cổ phiếu tăng, P/B tự nhiên tăng theo, phản ánh niềm tin của nhà đầu tư vào sự tăng trưởng và khả năng sinh lời của công ty.
EV/EBITDA	**+0.8450**	Đo lường định giá doanh nghiệp so với lợi nhuận hoạt động. Giá cổ phiếu cao hơn làm tăng giá trị doanh nghiệp, dẫn đến EV/EBITDA cao hơn, phản ánh khả năng sinh lời mạnh mẽ và sự lạc quan của nhà đầu tư.
Tỷ lệ Chi Trả Lãi Vay (Interest Coverage Ratio)	**+0.8250**	Các công ty có khả năng chi trả chi phí lãi vay mạnh mẽ được coi là ổn định về tài chính, điều này có xu hướng hỗ trợ giá cổ phiếu cao hơn.
ROE (Tỷ suất sinh lời trên vốn chủ sở hữu)	**+0.7269**	ROE cao cho thấy việc sử dụng vốn của cổ đông hiệu quả để tạo ra lợi nhuận, ảnh hưởng tích cực đến giá trị cổ phiếu.
P/E (Tỷ lệ Giá trên Lợi nhuận)	**+0.7019**	Tỷ lệ P/E tăng phản ánh kỳ vọng ngày càng tăng của nhà đầu tư về tăng trưởng lợi nhuận trong tương lai, thường liên quan đến định giá cổ phiếu cao hơn.

Tóm tắt: Giá cổ phiếu của công ty cho thấy **mối quan hệ tích cực mạnh mẽ** với **các chỉ báo về khả năng sinh lời và định giá** - đặc biệt là P/B, P/E, EV/EBITDA, ROE, và khả năng chi trả lãi vay. Điều này cho thấy những cải thiện về khả năng sinh lời, tỷ suất sinh lời trên vốn và định giá của nhà đầu tư trực tiếp thúc đẩy sự gia tăng giá cổ phiếu của công ty.

Các chỉ báo có tương quan âm mạnh

Chỉ báo	Tương quan	Giải thích
Tỷ lệ Thanh Toán Nhanh (Quick Ratio)	**-0.6393**	Tỷ lệ thanh toán nhanh cao có thể ngụ ý thanh khoản dư thừa và tài sản lưu động chưa được sử dụng hết, điều này có thể ảnh hưởng tiêu cực đến định giá thị trường.
Tỷ lệ Tiền mặt (Thanh khoản tức thời)	**-0.6577**	Tỷ lệ tiền mặt rất cao cho thấy tiền mặt nhàn rỗi hoặc ban quản lý thận trọng, báo hiệu tiềm năng tăng trưởng ngắn hạn thấp hơn.
Vốn chủ sở hữu (Shareholders' Equity)	**-0.6113**	Vốn chủ sở hữu tăng do phát hành cổ phiếu mới có thể làm loãng giá trị cổ phiếu hiện tại, dẫn đến mối quan hệ tiêu cực với giá cổ phiếu.
Giá trị Sổ sách trên mỗi Cổ phiếu (Book Value per Share)	**-0.4938**	Thường di chuyển ngược chiều với giá thị trường khi giá cổ phiếu tăng nhanh hơn giá trị sổ sách.
Tăng trưởng Tài sản Dài hạn	**-0.5181**	Mở rộng tài sản nhanh chóng có thể tạm thời làm giảm lợi nhuận, gây áp lực giảm lên giá cổ phiếu.

Tóm tắt: Các chỉ báo về thanh khoản và cấu trúc vốn cho thấy **tương quan tiêu cực** với giá cổ phiếu. Điều này ngụ ý rằng khi công ty tích lũy thanh khoản quá mức hoặc mở rộng vốn chủ sở hữu quá nhanh mà không có tăng trưởng lợi nhuận tương xứng, thị trường sẽ phản ứng tiêu cực.

Các chỉ báo có mối quan hệ yếu hoặc không rõ ràng

Các chỉ báo như Tăng trưởng Doanh thu, Tăng trưởng EPS và Vòng quay Tài sản có tương quan thấp ($|r| < 0.3$), cho thấy **không có mối quan hệ tuyến tính mạnh mẽ** với giá cổ phiếu trong giai đoạn quan sát.

3.4 Kết luận

Giá cổ phiếu của công ty phụ thuộc **mạnh mẽ nhất vào các chỉ số sinh lời và định giá** - đáng chú ý là **P/B, EV/EBITDA, Tỷ lệ Chi Trả Lãi Vay, ROE và P/E**.

Những điều này phản ánh **niềm tin của nhà đầu tư, hiệu quả hoạt động và tỷ suất sinh lời trên vốn chủ sở hữu**, tất cả đều là những động lực chính thúc đẩy giá cổ phiếu tăng.

Ngược lại, **các chỉ báo về thanh khoản và mở rộng vốn** (ví dụ: Tỷ lệ Thanh toán nhanh, Tỷ lệ Tiền mặt, Vốn chủ sở hữu) cho thấy **mối quan hệ tiêu cực**, cho thấy thanh khoản dư thừa hoặc vốn hóa quá mức mà không có tăng trưởng thu nhập tương ứng có thể dẫn đến định giá thị trường thấp hơn.