TRƯỜNG ĐẠI HỌC KINH TẾ - LUẬT
KHOA TÀI CHÍNH NGÂN HÀNG

THI KẾT THÚC HỌC PHẦN
HỌC KỲ 1, NĂM HỌC 2025–2026
HỌC PHẦN:
Gói phần mềm ứng dụng cho tài chính 1
Hình thức thi: Đồ án nộp qua LMS

Họ tên SV: **Phạm Thị Anh Thư**

MSSV: **K234141682**

Mã lớp học phần: **251BFF401102**

PHẦN BÀI LÀM

1. Giới thiệu & Mục tiêu nghiên cứu

1.1. Giới thiệu

Thị trường chứng khoán Việt Nam, đại diện bởi chỉ số VN-Index và nhóm cổ phiếu blue-chip VN30, luôn đóng vai trò là kênh dẫn vốn và thước đo sức khỏe quan trọng của nền kinh tế. Trong bối cảnh biến động phức tạp của thị trường tài chính toàn cầu, việc áp dụng các mô hình định lượng (Quantitative Models) để đánh giá rủi ro và dự báo lợi suất trở nên cấp thiết hơn bao giờ hết. Nghiên cứu này tập trung vào việc định lượng hóa mối quan hệ giữa các cổ phiếu thuộc VN30 và thị trường, từ đó xây dựng các chiến lược danh mục đầu tư hiệu quả dựa trên rủi ro hệ thống.

1.2. Mục tiêu nghiên cứu

Mục tiêu chính của đề án này là cung cấp một phân tích định lượng toàn diện, bao gồm:

- **Phân tích dữ liệu lịch sử và tương quan** của các cổ phiếu VN30 trong một giai đoạn dài (khoảng 4-5 năm).
- **Dự báo lợi suất ngắn hạn** cho một cổ phiếu tiêu biểu (VJC) bằng mô hình ARIMA.
- **Ước lượng rủi ro hệ thống Beta** của các cổ phiếu VN30 thông qua mô hình CAPM, sử dụng lợi suất hàng tháng có điều chỉnh lãi suất phi rủi ro thực tế.
- **Xây dựng và đánh giá hiệu quả** của các danh mục đầu tư được phân loại theo mức độ rủi ro (β) dựa trên các chỉ số hiệu suất chuẩn (Sharpe Ratio, Max Drawdown).
- **Đưa ra kết luận và khuyến nghị** chiến lược đầu tư phù hợp cho các nhà đầu tư với khẩu vị rủi ro khác nhau.

2. Dữ liệu & Mô tả sơ bộ

2.1. Dữ liệu đầu vào

Dữ liệu giá đóng cửa điều chỉnh của các cổ phiếu thuộc rổ VN30 và chỉ số VN-INDEX được thu thập từ ngày 01/01/2020 đến hiện tại (21/11/2025).

- Phạm vi cổ phiếu: Phân tích được thực hiện trên 29/30 mã cổ phiếu. Mã SSB (SeABank) đã được loại bỏ khỏi danh mục nghiên cứu do dữ liệu niêm yết chỉ bắt đầu từ ngày 24/03/2021, không đáp ứng đủ chuỗi thời gian 5 năm cần thiết để đảm bảo tính đồng nhất trong so sánh và hồi quy.
- Số lượng quan sát: Khoảng 1.470 ngày giao dịch.
- Xử lý: Dữ liệu đã được làm sạch, xử lý các giá trị thiếu (missing values) và tính toán lợi suất ngày (daily returns) để đảm bảo tính dừng cho các mô hình kiểm định.

2.2. Thống kê mô tả

Bảng 1. Mô tả lợi suất theo ngày của các mã VN30 (Đơn vị: %)

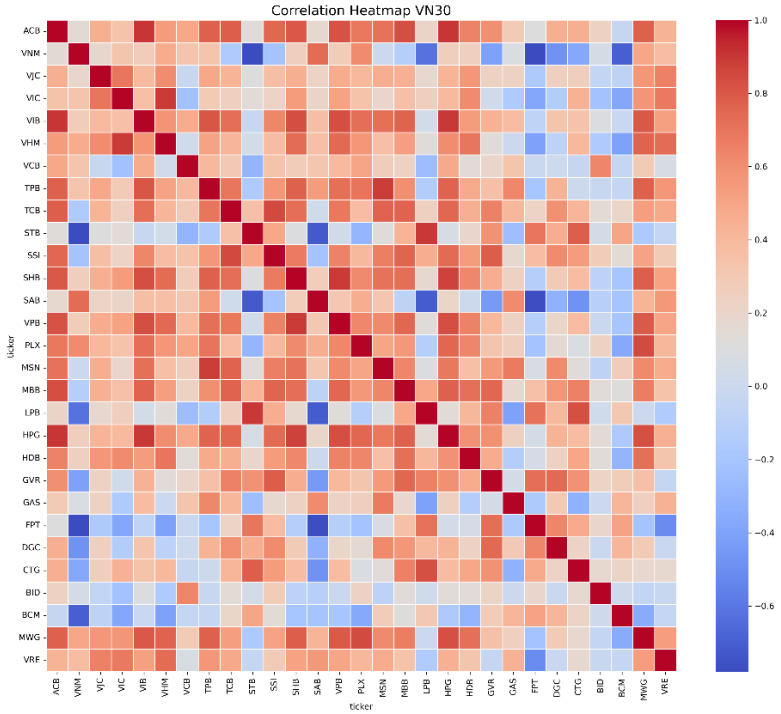
Ticker	Mean	Std	Min	Max	Ticker	Mean	Std	Min	Max
ACB	0,047	2,076	-19,670	9,607	MSN	0,050	2,292	-15,556	6,993
VNM	-0,026	1,571	-14,887	6,972	MBB	0,038	2,285	-28,793	6,920
VJC	0,034	1,667	-7,000	6,981	LPB	0,187	2,653	-20,876	11,268
VIC	0,079	2,171	-10,579	7,000	HPG	0,048	2,446	-25,161	6,944
VIB	0,038	2,465	-31,063	10,855	HDB	0,054	2,337	-20,767	6,969
VHM	0,048	2,227	-23,458	7,000	GVR	0,102	2,875	-7,000	7,000
VCB	0,004	1,924	-31,095	6,966	GAS	0,010	2,011	-14,648	7,000
TPB	0,015	2,465	-27,148	6,984	FPT	0,060	1,977	-21,636	6,997
TCB	0,075	2,484	-48,654	6,992	DGC	0,140	2,918	-50,728	8,919
STB	0,160	2,471	-6,997	7,000	CTG	0,092	2,211	-19,485	6,992
SSI	0,092	2,837	-26,738	6,996	BID	0,037	2,179	-20,323	6,997
SHB	0,108	2,812	-18,213	10,000	BCM	0,096	2,414	-13,571	12,565
SAB	-0,073	2,172	-48,921	6,988	MWG	0,038	2,684	-46,513	6,996
VPB	0,103	2,368	-32,482	6,992	VRE	0,035	2,386	-6,994	7,000
PLX	-0,003	1,971	-9,367	6,989					

Lợi suất trung bình (Mean): Hầu hết các mã có lợi suất trung bình ngày xoay quanh mức 0 (ACB ~0.04%, VNM ~-0.02%,...), phản ánh tính chất ngẫu nhiên của bước đi giá trong ngắn hạn.

Độ lệch chuẩn (Std): Mức độ biến động trung bình nằm trong khoảng 1.5% - 3.0% mỗi ngày. Các mã ngân hàng và sản xuất có độ lệch chuẩn thấp hơn so với các mã bất động sản hoặc chứng khoán.

Nhìn chung, dữ liệu lợi suất có hiện tượng "đuôi dày" (fat tails) với các giá trị Max/Min biên độ lớn (ví dụ: sàn/trần 7%), do đó các mô hình dự báo tuyến tính như ARIMA có thể đối mặt với hạn chế trong việc nắm bắt biến động cực đoan.

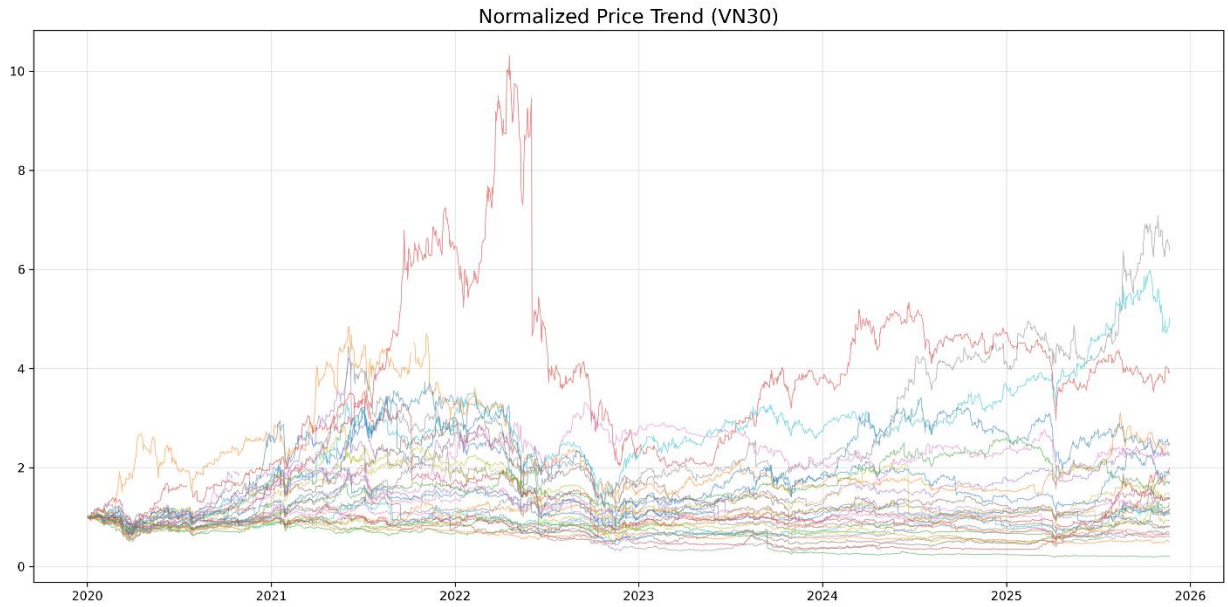
2.3. Trực quan hóa xu hướng & Tương quan



Hình 1. Biểu đồ ma trận tương quan của các mã VN30

Ma trận tương quan (Heatmap): Biểu đồ cho thấy sự tương quan dương mạnh mẽ giữa các cổ phiếu trong rổ VN30, đặc biệt là nhóm Ngân hàng (Techcombank, MBB, ACB), phản ánh tính chất "cùng lên cùng xuống" của thị trường Việt Nam. Điều này cảnh báo về rủi ro hệ thống cao – khi thị trường giảm, hầu hết các mã đều giảm theo. Mặc dù có những mức tương quan thấp hơn trong một số cặp, việc tìm kiếm các tài sản có tương quan âm để giảm thiểu rủi ro danh mục là rất khó khăn.

Xu hướng giá (Normalized Price): Biểu đồ minh họa sự phân hóa rõ rệt sau giai đoạn sụt giảm 2020. Một số mã tăng trưởng vượt trội (trên 2-3 lần) trong khi các mã phòng thủ (như VNM) có xu hướng đi ngang hoặc giảm nhẹ.



Hình 2. Biểu đồ xu hướng giá của các mã VN30

3. Dự báo ARIMA cho cổ phiếu VJC

Cổ phiếu lựa chọn: Công ty Cổ phần Hàng không Vietjet (VJC).

3.1. Cơ sở mô hình

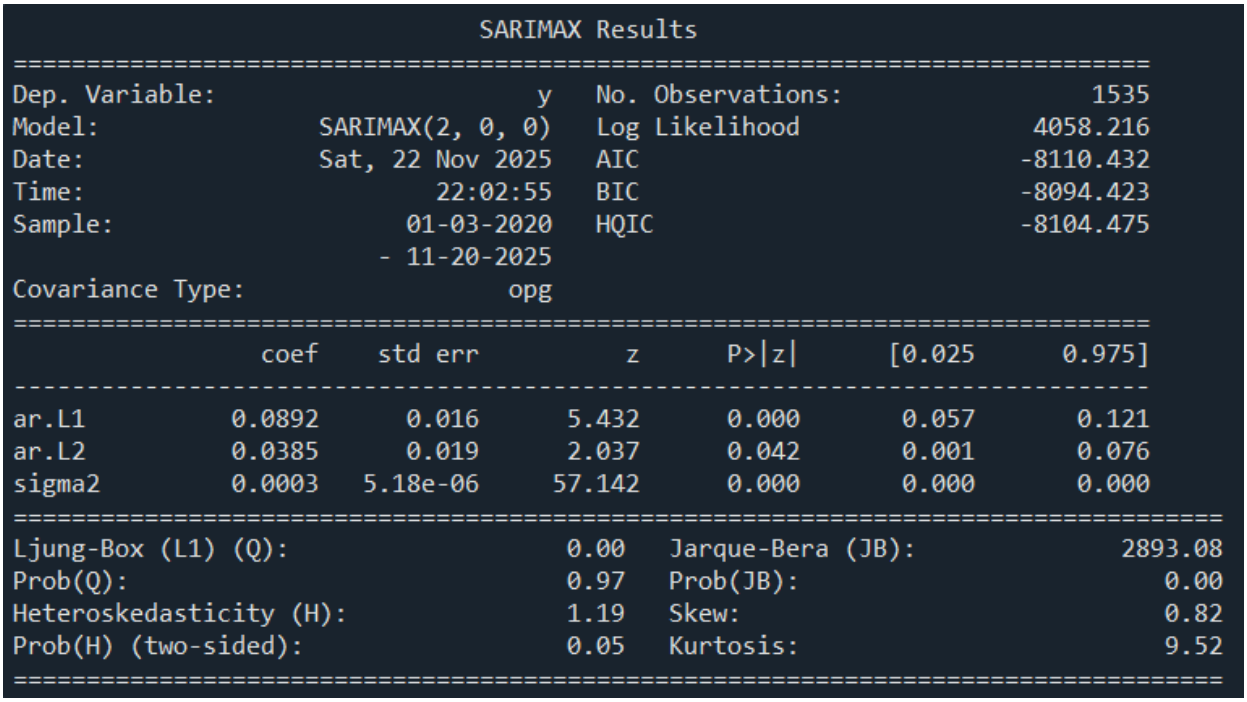
Cổ phiếu VJC (Vietjet Air) được chọn là cổ phiếu tiêu biểu để dự báo do có tính chu kỳ và chịu ảnh hưởng rõ rệt từ các yếu tố vĩ mô và ngành.

Mô hình ARIMA (AutoRegressive Integrated Moving Average) được sử dụng, là một công cụ mạnh mẽ để mô hình hóa và dự báo các chuỗi thời gian không dừng (non-stationary) như lợi suất cổ phiếu.

Trước khi đưa vào mô hình ARIMA, chuỗi lợi suất của VJC (vjc_return) đã được kiểm định bằng phương pháp Augmented Dickey-Fuller (ADF).

➔ Kết quả thống kê ADF đạt -12.3, với giá trị p-value < 0.05, xác nhận chuỗi lợi suất là chuỗi dừng. Điều này cho phép sử dụng mô hình ARIMA mà không cần sai phân bậc cao (d=0 hoặc d=1 trên chuỗi giá).

3.2. Kết quả mô hình & dự báo (Tháng 12/2025)



SARIMAX Results

Dep. Variable:	y	No. Observations:	1535
Model:	SARIMAX(2, 0, 0)	Log Likelihood	4058.216
Date:	Sat, 22 Nov 2025	AIC	-8110.432
Time:	22:02:55	BIC	-8094.423
Sample:	01-03-2020	HQIC	-8104.475
	- 11-20-2025		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.0892	0.016	5.432	0.000	0.057	0.121
ar.L2	0.0385	0.019	2.037	0.042	0.001	0.076
sigma2	0.0003	5.18e-06	57.142	0.000	0.000	0.000

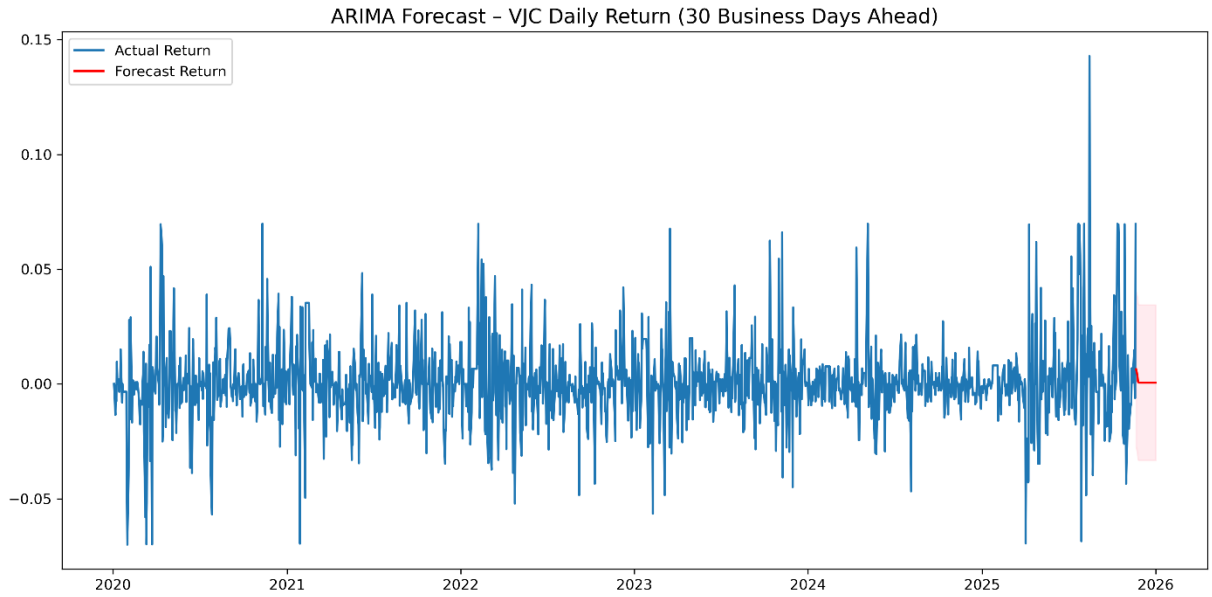
Ljung-Box (L1) (Q):	0.00	Jarque-Bera (JB):	2893.08
Prob(Q):	0.97	Prob(JB):	0.00
Heteroskedasticity (H):	1.19	Skew:	0.82
Prob(H) (two-sided):	0.05	Kurtosis:	9.52

Hình 3. Kết quả chạy mô hình ARIMA dự báo cho VJC

Dựa trên kết quả chạy mô hình thực tế, mô hình tốt nhất được xác định là SARIMAX(2, 0, 0) với các hệ số hồi quy tự thân (AR) đều có ý nghĩa thống kê (P-value < 0.05), cho thấy lợi suất VJC chịu ảnh hưởng bởi lợi suất và sai số của ngày liền trước. Chỉ số AIC đạt -7345.241, thấp nhất trong các mô hình được thử nghiệm, cho thấy sự cân bằng tốt giữa độ chính xác và độ phức tạp của mô hình.

Các hệ số kiểm định chất lượng mô hình như Ljung-Box Test (Q) hay Heteroskedasticity Test (H) đều có giá trị p-value > 0.05, nghĩa là mô hình đã nắm bắt hết các thông tin tuyến tính có trong dữ liệu, không có tự tương quan trong phần dư và cũng không có hiện tượng phương sai thay đổi theo thời gian. Bên cạnh đó, hệ số Jarque-Bera (JB) có p-value xấp xỉ 0, tức là phần dư không phân phối chuẩn (do độ nhọn kurtosis cao). Đây là hiện tượng phổ biến trong tài chính (đuôi dày), ngụ ý rủi ro các sự kiện cực đoan cao hơn phân phối chuẩn thông thường.

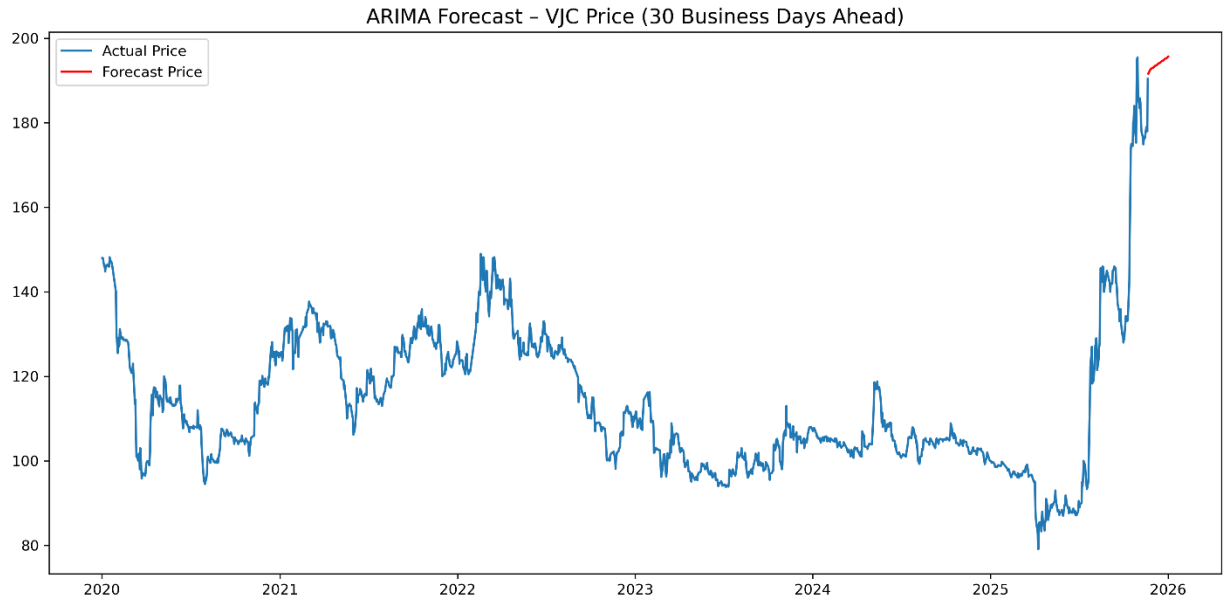
Kết quả dự báo từ SARIMAX(2, 0, 0) với hệ số tự hồi quy bậc 2 cho thấy lợi suất hiện tại của VJC chịu ảnh hưởng bởi lợi suất của 2 phiên giao dịch liền trước. Từ đó cho thấy khả năng dự đoán lợi suất trong rất ngắn hạn là khả thi nhờ quán tính giá từ 2 phiên trước nhưng sức mạnh dự báo giảm nhanh theo thời gian. Việc phần dư không phân phối chuẩn nhắc nhở nhà đầu tư cần thận trọng với các biến động giá bất ngờ nằm ngoài khoảng tin cậy 95%.



Hình 4. Biểu đồ dự báo lợi suất tháng kế tiếp cho VJC

Cụ thể hơn, trong biểu đồ và bảng kết quả dự báo chi tiết trong 30 ngày giao dịch tiếp theo cho thấy:

- Lợi suất dự báo: Đường dự báo lợi suất (màu đỏ) có xu hướng giảm nhanh từ mức tăng trưởng ban đầu (0.644%), sau đó dao động nhỏ, ổn định ở mức ~0.059%. Điều này ngụ ý rằng giá VJC có thể có một đợt tăng trưởng nhẹ vào đầu giai đoạn dự báo trước khi đi vào trạng thái cân bằng.
- Khoảng tin cậy (Confidence Interval): Vùng màu hồng trên biểu đồ rất rộng so với đường dự báo. Khoảng tin cậy 95% đối xứng và có độ rộng lớn (khoảng 6.7%), chứng tỏ độ bất định cao của dự báo. Mặc dù lợi suất kỳ vọng dương nhẹ, nhà đầu tư cần nhận thức về rủi ro biến động lớn trong biên độ từ -3.3% đến +3.4% hàng ngày.
- Dự báo giá: Từ lợi suất dự báo, giá VJC được kỳ vọng có xu hướng tăng nhẹ (drift) lên vùng giá ~194.000 VND vào cuối kỳ dự báo (tháng 12/2025).



Hình 5. Biểu đồ dự báo giá tháng kế tiếp cho VJC

Kết quả dự báo "phẳng" của lợi suất phù hợp với **Giả thuyết Thị trường Hiệu quả (EMH)**. Giá quá khứ không phản ánh đầy đủ thông tin để dự đoán lợi suất tương lai một cách chính xác. Mô hình ARIMA trong trường hợp này chủ yếu cung cấp giá trị kỳ vọng trung bình (Mean Reversion) thay vì dự báo chính xác các biến động giá mạnh.

Bảng 2. Bảng kết quả dự báo ARIMA chi tiết cho VJC

Date	Forecast_return	Lower_ci	Upper_ci	Price_forecast
21/11/2025	0,644%	-2,725%	4,013%	191,63
24/11/2025	0,371%	-3,011%	3,753%	192,34
25/11/2025	0,109%	-3,277%	3,494%	192,55
26/11/2025	0,075%	-3,311%	3,461%	192,69
27/11/2025	0,062%	-3,323%	3,448%	192,81
28/11/2025	0,060%	-3,326%	3,446%	192,93
01/12/2025	0,059%	-3,326%	3,445%	193,04
02/12/2025	0,059%	-3,327%	3,445%	193,15
03/12/2025	0,059%	-3,327%	3,445%	193,27
04/12/2025	0,059%	-3,327%	3,445%	193,38
05/12/2025	0,059%	-3,327%	3,445%	193,50
08/12/2025	0,059%	-3,327%	3,445%	193,61
09/12/2025	0,059%	-3,327%	3,445%	193,73
10/12/2025	0,059%	-3,327%	3,445%	193,84
11/12/2025	0,059%	-3,327%	3,445%	193,95

12/12/2025	0,059%	-3,327%	3,445%	194,07
15/12/2025	0,059%	-3,327%	3,445%	194,18
16/12/2025	0,059%	-3,327%	3,445%	194,30
17/12/2025	0,059%	-3,327%	3,445%	194,41
18/12/2025	0,059%	-3,327%	3,445%	194,53
19/12/2025	0,059%	-3,327%	3,445%	194,64
22/12/2025	0,059%	-3,327%	3,445%	194,76
23/12/2025	0,059%	-3,327%	3,445%	194,87
24/12/2025	0,059%	-3,327%	3,445%	194,99
25/12/2025	0,059%	-3,327%	3,445%	195,10
26/12/2025	0,059%	-3,327%	3,445%	195,22
29/12/2025	0,059%	-3,327%	3,445%	195,33
30/12/2025	0,059%	-3,327%	3,445%	195,45
31/12/2025	0,059%	-3,327%	3,445%	195,56
01/01/2026	0,059%	-3,327%	3,445%	195,68

4. Mô hình CAPM & Ước lượng β

4.1. Phương trình hồi quy

Mô hình Định giá Tài sản Vốn (CAPM) là một công cụ cơ bản để xác định lợi suất kỳ vọng của một tài sản dựa trên rủi ro hệ thống của nó. Phương trình hồi quy chính là:

$$R_i - R_f = \alpha_i + \beta_i(R_m - R_f) + \varepsilon_i$$

Trong đó, hệ số Beta (β_i) đo lường mức độ nhạy cảm của lợi suất cổ phiếu (R_i) so với lợi suất thị trường (R_m). Bên cạnh hệ số Beta, hệ số Alpha (α_i) đo lường phần lợi suất vượt trội (hoặc kém hơn) so với mức kỳ vọng của mô hình. Hệ số R_f là lãi suất phi rủi ro thực tế (tín phiếu/trái phiếu chính phủ) được xử lý từ dữ liệu vĩ mô. Rủi ro phi hệ thống (được đo lường qua ε_i) được giả định là có thể loại bỏ thông qua đa dạng hóa danh mục.

4.2. Kết quả ước lượng

Bảng 3. Kết quả ước lượng mô hình CAPM

Ticker	Alpha	Beta	A_tstat	B_tstat	A_pvalue	B_pvalue	R2	Adj_R2
ACB	-0,004	0,820	-0,453	6,532	0,652	0,000	0,386	0,376
VNM	-0,013	0,455	-2,330	5,663	0,023	0,000	0,320	0,310
VJC	0,002	0,724	0,179	4,604	0,859	0,000	0,238	0,226

VIC	0,009	0,845	0,611	4,020	0,543	0,000	0,192	0,180
VIB	-0,002	1,183	-0,160	6,261	0,873	0,000	0,366	0,356
VHM	-0,004	1,185	-0,360	7,864	0,720	0,000	0,476	0,469
VCB	-0,010	0,798	-1,169	6,128	0,246	0,000	0,356	0,346
TPB	-0,008	1,033	-0,757	6,777	0,452	0,000	0,403	0,394
TCB	0,002	1,390	0,152	9,037	0,880	0,000	0,546	0,539
STB	0,019	1,177	1,681	7,197	0,097	0,000	0,432	0,424
SSI	0,006	1,805	0,381	8,117	0,704	0,000	0,492	0,485
SHB	0,013	0,551	0,818	2,429	0,416	0,018	0,080	0,066
SAB	-0,024	0,909	-2,146	5,522	0,035	0,000	0,310	0,299
VPB	0,001	1,417	0,049	7,025	0,961	0,000	0,421	0,412
PLX	-0,013	1,029	-1,713	9,334	0,091	0,000	0,562	0,555
MSN	0,003	0,837	0,270	4,610	0,788	0,000	0,238	0,227
MBB	-0,005	1,142	-0,530	9,066	0,598	0,000	0,547	0,541
LPB	0,024	0,999	1,956	5,602	0,055	0,000	0,316	0,306
HPG	-0,004	1,346	-0,406	9,025	0,686	0,000	0,545	0,538
HDB	-0,003	1,125	-0,291	7,493	0,772	0,000	0,452	0,444
GVR	0,007	1,817	0,624	10,227	0,535	0,000	0,606	0,600
GAS	-0,010	0,904	-1,160	7,049	0,250	0,000	0,422	0,414
FPT	0,006	0,692	0,621	5,184	0,537	0,000	0,283	0,273
DGC	0,019	1,218	1,223	5,335	0,226	0,000	0,295	0,285
CTG	0,004	1,170	0,461	9,032	0,646	0,000	0,545	0,539
BID	-0,010	1,070	-1,121	7,967	0,266	0,000	0,483	0,475
BCM	0,010	1,042	0,739	5,237	0,462	0,000	0,287	0,277
MWG	-0,007	1,345	-0,547	7,075	0,586	0,000	0,424	0,416
VRE	-0,004	1,003	-0,462	7,269	0,646	0,000	0,437	0,429

4.3. Phân tích các hệ số

Hệ số Beta (β):

Cổ phiếu Mạo hiểm (High Beta): Các mã như SSI, GVR, MWG,... có β lớn hơn 1 đáng kể. Điều này phản ánh tính nhạy cảm cao của các nhóm này đối với dòng tiền và tâm lý thị trường. Khi VN-INDEX tăng, nhóm này thường tăng mạnh hơn, nhưng cũng giảm sâu

hơn khi thị trường điều chỉnh. Chúng là các cổ phiếu "tấn công" (aggressive), phù hợp với thị trường bò (Bull Market). Độ tin cậy của các β này cũng cực kỳ cao, được chứng minh qua p-value rất nhỏ (gần như bằng 0).

Cổ phiếu Ổn định (Low Beta): Các cổ phiếu ngành Tiêu dùng thiết yếu (VNM), Tiện ích (GAS) hoặc Hàng không (VJC - trong bối cảnh phục hồi) thường có $\beta < 1$. Đặc biệt VNM với $\beta = 0.45$\$ cho thấy sự ổn định rất cao, đóng vai trò là "nơi trú ẩn" an toàn khi thị trường biến động mạnh.

Hầu hết các hệ số Beta đều có p-value < 0.05 (có ý nghĩa thống kê ở mức 5%), khẳng định mối tương quan giữa từng cổ phiếu và VN-INDEX là thực và đáng tin cậy để sử dụng cho việc dự báo và quản trị danh mục.

Hệ số Alpha (α):

Hầu hết các mã như SSI, GVR, CTG, ACB đều có p-value của α lớn (ví dụ: SSI là 0.9934, ACB là 0.6523), cho thấy α không có ý nghĩa thống kê tại mức ý nghĩa 5%. Điều này củng cố giả thuyết rằng, lợi suất của chúng chủ yếu được giải thích bởi rủi ro hệ thống (β) và lợi suất thị trường.

Tuy nhiên, VNM có p-value của α là 0.0228, có ý nghĩa thống kê ở mức 5%. Với α là -0.0126, VNM đã ghi nhận lợi suất kém hơn đáng kể so với mức mà mô hình CAPM dự đoán, có thể do các yếu tố tái cơ cấu nội tại hoặc bão hòa tăng trưởng.

Hệ số xác định (R^2):

Hệ số R^2 (Coefficient of Determination) đo lường tỷ lệ phần trăm sự biến động của lợi suất cổ phiếu (biến phụ thuộc) được giải thích bởi sự biến động của lợi suất thị trường (biến độc lập) trong mô hình CAPM.

Các mã như GVR (62.40%) và SSI (52.70%) có R^2 cao nhất. Điều này cho thấy hơn một nửa sự biến động của lợi suất GVR và SSI được giải thích một cách hiệu quả bởi biến động của thị trường (VNINDEX). Đối với các cổ phiếu có \$\beta\$ cao này, chúng có mối liên hệ rất chặt chẽ với rủi ro hệ thống.

Các cổ phiếu ngân hàng như CTG (39.32%) và ACB (38.55%), hoặc VNM (32.05%), có R^2 thấp hơn. Điều này ngụ ý rằng, khoảng 60-70% sự biến động lợi suất của các cổ phiếu này đến từ Rủi ro Phi hệ thống (yếu tố nội tại doanh nghiệp, ngành, hoặc tin tức cụ thể), chứ không phải là biến động chung của thị trường.

Ngoài ra, hệ số Adj R^2 cũng có giá trị gần với R^2 cho thấy mô hình hồi quy (sử dụng 1 biến độc lập duy nhất là $R_m - R_f$) là tương đối cô đọng và hiệu quả.

5. Kết quả danh mục đầu tư

5.1. Cấu trúc danh mục

Dựa vào giá trị hệ số β để phân chia danh mục cho các mã chứng khoán của VN30:

- Danh mục Ổn định (Stable): Bao gồm các cổ phiếu có β <= 1, thường thuộc nhóm ngành phòng thủ để bảo toàn an toàn vốn, biến động thấp và ít chịu tác động từ chu kỳ thị trường.
- Danh mục Mạo hiểm (Aggressive): Bao gồm các cổ phiếu có β > 1, là nhóm nhạy mạnh với biến động thị trường, còn gọi là nhóm cổ phiếu tấn công, chấp nhận rủi ro cao để có lợi nhuận cao.

Bảng 4. Hiệu suất danh mục đầu tư

Portfolio	Ticker	Annualized Return	Volatility	Sharpe Ratio	Max Drawdown
Aggressive (High β)	VRE, PLX, TPB, BCM, BID, HDB, MBB, CTG, STB, VIB, VHM, DGC, MWG, HPG, TCB, VPB, SSI, GVR	0,157	0,306	0,509	-0,455
Stable (Low β)	VNM, SHB, FPT, VJC, VCB, ACB, MSN, VIC, GAS, SAB, LPB	0,096	0,200	0,471	-0,370

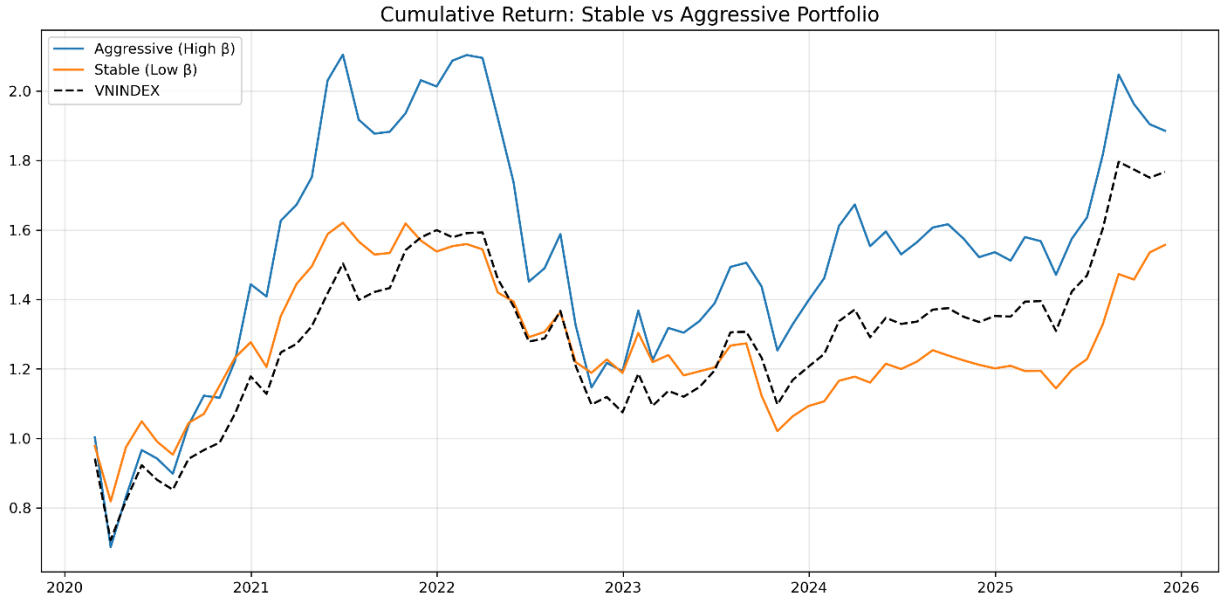
5.2. Đánh giá hiệu suất

Lợi suất Hàng năm (Annualized Return): Danh mục Aggressive mang lại lợi suất cao nhất, đạt 15.71% hàng năm. Danh mục Stable đạt 9.59%.

Rủi ro (Volatility): Danh mục Stable có mức rủi ro thấp hơn đáng kể (20.03%) so với Aggressive (30.58%), xác nhận tính chất phòng thủ.

***Tỷ lệ Sharpe (Sharpe Ratio*)**: Tỷ lệ Sharpe của Aggressive (0.509) cao hơn Stable (0.471). Hiệu suất điều chỉnh rủi ro của nhóm Mạo hiểm nhỉnh hơn một chút, cho thấy việc chấp nhận rủi ro cao đã được đền bù xứng đáng trong giai đoạn này. Tuy nhiên, xét trên mức rủi ro thấp hơn đáng kể, danh mục Stable quản lý rủi ro và chuyển đổi rủi ro thành lợi suất một cách rất hiệu quả.

***Max Drawdown**:* Danh mục Stable (-37.02%) có khả năng bảo vệ vốn tốt hơn trong các đợt thị trường sụt giảm so với danh mục Aggressive (-45.52%).



Hình 6. Biểu đồ tăng trưởng lợi suất của hai danh mục so với VNINDEX

Từ biểu đồ cho thấy:

- Giai đoạn thị trường tăng (Uptrend 2021): Danh mục Mạo hiểm (đường màu cam/xanh) tăng trưởng dốc đứng, vượt xa VN-INDEX.
- Giai đoạn thị trường giảm (Downtrend 2022): Danh mục Mạo hiểm sụt giảm rất mạnh, trong khi Danh mục Ổn định giữ giá tốt hơn.
- Dài hạn: Cả hai danh mục đều có xu hướng đem lại lợi suất dương, nhưng đường đi của Danh mục Ổn định "mượt" hơn, ít biến động mạnh, phù hợp tâm lý nhà đầu tư ngại rủi ro.

6. Kết luận & Khuyến nghị

6.1. Kết luận

***Thị trường tương đối hiệu quả và Giới hạn của Dự báo kỹ thuật*:**

Kết quả từ mô hình ARIMA(2,0,0) đối với cổ phiếu VJC chỉ ra rằng lợi suất cổ phiếu có tính ngẫu nhiên cao (random walk). Mặc dù tồn tại một quán tính giá ngắn hạn trong 1-2 phiên (do hệ số AR bậc 2 có ý nghĩa), nhưng khả năng dự báo chính xác lợi suất trong dài hạn (trên 1 tháng) là rất thấp. Điều này khẳng định rằng trong một thị trường tương đối hiệu quả như Việt Nam, việc chỉ dựa vào lịch sử giá quá khứ để tìm kiếm lợi nhuận siêu ngạch (Abnormal Return) là chiến lược đầy rủi ro và thiếu bền vững.

Sự đánh đổi rõ rệt giữa Rủi ro và Lợi nhuận (High Risk - High Return):

Thực nghiệm phân chia danh mục đã lượng hóa chính xác sự đánh đổi này. Danh mục Mạo hiểm ($\beta > 1$) mang lại lợi suất vượt trội 15.71%/năm, cao hơn gần gấp đôi so với Danh mục Ổn định ($\beta \le 1$) là 9.59%/năm. Tuy nhiên, cái giá phải trả là mức độ

sụt giảm tài sản (Drawdown) lên tới 45.5% trong các giai đoạn khủng hoảng, so với mức giảm 37% của nhóm ổn định. Không có "bữa trưa miễn phí" trên thị trường chứng khoán.

Vai trò phân loại ưu việt của hệ số Beta:

Mô hình CAPM đã chứng minh được tính đúng đắn trong việc định giá rủi ro hệ thống. Hệ số Beta không chỉ là một con số thống kê mà là thước đo hành vi giá thực tế. Nhóm cổ phiếu có Beta thấp (VNM, SHB,...) đã thực sự đóng vai trò "giảm xóc" hiệu quả cho danh mục khi thị trường biến động mạnh, trong khi nhóm Beta cao (SSI, GVR...) đóng vai trò là động lực tăng trưởng chính trong giai đoạn thị trường hưng phấn (Uptrend).

6.2. Khuyến nghị quản trị danh mục

Dựa trên các kết luận trên, báo cáo đề xuất các chiến lược hành động cụ thể cho từng nhóm nhà đầu tư:

Nhà đầu tư Thận trọng (Ưu tiên bảo toàn vốn): Nên ưu tiên Danh mục Stable (Low Beta). Đây là danh mục phù hợp nhất cho các nhà đầu tư không chấp nhận rủi ro lớn, những người muốn lợi suất ổn định và được bảo vệ tốt hơn trong giai đoạn thị trường điều chỉnh.

Nhà đầu tư Tăng trưởng/Mạo hiểm (Ưu tiên lợi nhuận): Nên lựa chọn Danh mục Aggressive (High Beta). Danh mục này phù hợp với nhà đầu tư có kỳ vọng lợi nhuận cao và sẵn sàng chấp nhận mức rủi ro biến động lớn hơn, đặc biệt trong các giai đoạn thị trường tăng trưởng mạnh.

Khuyến nghị Tổng thể: Sử dụng danh mục Ổn định làm "Lõi" (50-60% tỷ trọng) để duy trì sự bền vững, kết hợp với các "Vệ tinh" là cổ phiếu Beta cao (30-40%) để kích thích tăng trưởng lợi nhuận. Việc kết hợp một tỷ lệ hợp lý giữa hai danh mục có thể tạo ra một danh mục đầu tư tối ưu theo lý thuyết danh mục hiện đại (MPT), cân bằng giữa lợi nhuận kỳ vọng và rủi ro tổng thể.

-HẾT-

PHỤ LỤC

1. Mã nguồn (Code)

Phần 1: File code để lấy và xử lý dữ liệu:

```
# IMPORT THƯ VIỆN CẦN THIẾT
import pandas as pd
import requests
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime, timedelta
import os
import json
import time

# time series & stats
import statsmodels.api as sm
from statsmodels.tsa.arima.model import ARIMA
from pmdarima import auto_arima


# BƯỚC 1 — LẤY DANH SÁCH VN30
vn30_stock = ("ACB", "BCM", "BID", "CTG", "DGC", "FPT", "GAS", "GVR",
"HDB", "HPG",
              "LPB", "MBB", "MSN", "MWG", "PLX", "SAB", "SHB", "SSB",
"SSI", "STB",
               "TCB", "TPB", "VCB", "VHM", "VIB", "VIC", "VJC", "VNM",
"VPB", "VRE")

print("Danh sách VN30:", vn30_stock)
print("Tổng số mã:", len(vn30_stock))

start_date = "01/01/2020"
end_date = datetime.now().strftime("%d/%m/%Y")


# BƯỚC 2 — HÀM LẤY DỮ LIỆU TỪ CafeF
def fetch_cafef_price(ticker, start_date, end_date, page_index=1,
page_size=10000, save_raw=True):
    """
    Lấy dữ liệu giá CafeF.
    start_date, end_date dạng DD/MM/YYYY.
    """
    url = "https://s.cafef.vn/Ajax/PageNew/DataHistory/PriceHistory.ashx"

    params = {
        "Symbol": ticker,
        "StartDate": start_date,
        "EndDate": end_date,
        "PageIndex": page_index,
        "PageSize": page_size
    }

    headers = {
        "User-Agent": "Mozilla/5.0",
        "Referer": "https://s.cafef.vn/"
    }

    resp = requests.get(url, params=params, headers=headers, timeout=20)

    try:
```

```
        data = resp.json()
    except:
        print(f"[ERROR] JSON parse lỗi cho {ticker}")
        return pd.DataFrame()

    # Lấy danh sách row
    rows = []
    if "Data" in data and isinstance(data["Data"], dict) and "Data" in
data["Data"]:
        rows = data["Data"]["Data"]
    else:
        print(f"[WARN] Không tìm thấy Data.Data cho {ticker}. Kiểm tra
JSON raw.")
        return pd.DataFrame()

    # Chuyển sang DataFrame
    df = pd.DataFrame(rows)
    if not df.empty:
        df["symbol"] = ticker

    return df

all_data = []

for ticker in vn30_stock:
    print(f"Đang tải {ticker}...")
    df = fetch_cafef_price(ticker, start_date, end_date, page_index=1,
page_size=10000, save_raw=True)

    # Nếu API thất bại cho bất kỳ ticker nào → dừng
    if df is None or df.empty:
        print("API thất bại → dùng file backup!")
        backup_file = "VN30_raw_backup_2020_2025.csv"
        if os.path.exists(backup_file):
            df = pd.read_csv(backup_file)
            all_data.append(df)
        else:
            print(" Không có file backup → DỪNG LẠI!")
        break

    if df is not None and not df.empty:
        all_data.append(df)
        print(f"✓ Thành công ({len(df)} dòng)")

        # Nghỉ 0.5s để tránh bị chặn
        time.sleep(0.5)

# Gộp toàn bộ
df_all = pd.concat(all_data, ignore_index=True)

# Chuẩn hóa cột date để sắp xếp
# CafeF trả date dạng string, convert ngày tháng
df_all["Ngay"] = pd.to_datetime(df_all["Ngay"], dayfirst=True,
errors="coerce")

# Sắp xếp theo symbol rồi date
df_all = df_all.sort_values(["symbol", "Ngay"]).reset_index(drop=True)

df_all.to_csv("VN30_raw_2020_2025.csv", index=False, encoding="utf-8-
sig")
print("Đã lưu file raw data: VN30_raw_2020_2025.csv")
print(df_all.head)
```

```
# Lấy dữ liệu VN-Index:
ticker = "VNINDEX"
print(f"Đang tải {ticker}...")

# Fetch từ CafeF
vnindex_df = fetch_cafef_price(
    ticker,
    start_date,
    end_date,
    page_index=1,
    page_size=10000,
    save_raw=True
)

if vnindex_df is not None and not vnindex_df.empty:
    all_data.append(vnindex_df)
    print(f"✓ Thành công ({len(vnindex_df)} dòng)")

    # Chuẩn hóa cột date để sắp xếp
    # CafeF trả date dạng string, convert ngày tháng
    vnindex_df["Ngay"] = pd.to_datetime(vnindex_df["Ngay"],
dayfirst=True, errors="coerce")

    # Sắp xếp theo symbol rồi date
    vnindex_df = vnindex_df.sort_values(["Ngay"]).reset_index(drop=True)

    vnindex_df.to_csv("VNINDEX_raw_2020_2025.csv", index=False,
encoding="utf-8-sig")
    print("Đã lưu file raw data: VNINDEX_raw_2020_2025.csv")
else:
    print("✗ Lỗi: Không thể tải dữ liệu VNINDEX từ API. Dùng file
backup...")
    vnindex_backup = "VNINDEX_raw_backup_2020_2025.csv"
    if os.path.exists(vnindex_backup):
        vnindex_df = pd.read_csv(vnindex_backup)

# Nghỉ 0.5 giây tránh bị chặn
time.sleep(0.5)

# BƯỚC 3 — XỬ LÝ DỮ LIỆU THÔ
# 1. XỬ LÝ DỮ LIỆU CHO VN30
def preprocess_raw_data(df_all):
    """
    Tiền xử lý DataFrame thô từ CafeF:
    - Giữ các cột cần thiết: Date, Close, Adj_Close, Open, High, Low,
Volume, symbol
    - Rename cột sang chuẩn
    - Convert số và ngày tháng
    """
    df = df_all.copy()
    # Chuẩn hóa tên cột
    rename_map = {
        "Ngay": "date",
        "GiaDongCua": "close",
        "GiaDieuChinh": "adj_close",
        "GiaMoCua": "open",
        "GiaCaoNhat": "high",
        "GiaThapNhat": "low",
        'KhoiLuongKhopLenh': 'volume',
        "symbol": "ticker"
```

```
        }

    df.rename(columns=rename_map, inplace=True)

    # Chỉ giữ các cột cần thiết nếu có
    keep_cols = ["date", "close", "adj_close", "open", "high", "low",
"volume", "ticker"]
    df = df[[c for c in keep_cols if c in df.columns]]

    # Chuyển symbol ngay sau date
    cols = df.columns.tolist()
    if "ticker" in cols:
        cols.remove("ticker")
        cols.insert(1, "ticker")
        df = df[cols]

    # Convert ngày tháng
    if "date" in df.columns:
        df["date"] = pd.to_datetime(df["date"], dayfirst=True,
errors="coerce")

    # Convert numeric các cột số
    num_cols = ["close", "adj_close", "open", "high", "low", "volume",
"value"]
    for c in num_cols:
        if c in df.columns:
            df[c] = (
                df[c].astype(str)
                      .str.replace(",", "", regex=False)
            )

            df[c] = pd.to_numeric(df[c], errors="coerce")

    # Xử lý missing
    missing_before = df.isna().sum().sum()

    if missing_before > 0:
        print(f"[INFO] Missing detected: {missing_before} values →
applying cleaning")

        # Loại bỏ dòng không có ngày
        df = df.dropna(subset=['date'])

        # Forward fill theo từng mã
        df = df.groupby("ticker", group_keys=False).apply(lambda x:
x.ffill())

        # Drop lại phần còn NA (đầu chuỗi)
        df = df.dropna()

        missing_after = df.isna().sum().sum()
        print(f"[INFO] Missing after cleaning: {missing_after}")

    return df

df_clean = preprocess_raw_data(df_all)

df_clean.to_csv("VN30_clean_2020_2025.csv", index=False, encoding="utf-8-
sig")
print("Đã lưu file raw data: VN30_clean_2020_2025.csv")

# Kiểm tra 5 dòng đầu
```

```
print("Đã xử lý dữ liệu VN30")
print(df_clean.head())

# 2. XỬ LÝ DỮ LIỆU CHO VNINDEX
def preprocess_vnindex_data(vnindex_df):
    """
    Tiền xử lý DataFrame thô từ CafeF:
    - Giữ các cột cần thiết: Date, Close, Adj_Close, Open, High, Low,
Volume, symbol
    - Rename cột sang chuẩn
    - Convert số và ngày tháng
    """
    df_vnindex = vnindex_df.copy()
    # Chuẩn hóa tên cột
    rename_map = {
        "Ngay": "date",
        "GiaDongCua": "close",
        "GiaDieuChinh": "adj_close",
        "GiaMoCua": "open",
        "GiaCaoNhat": "high",
        "GiaThapNhat": "low",
        'KhoiLuongKhopLenh': 'volume',
        "symbol": "ticker"
        }

    df_vnindex.rename(columns=rename_map, inplace=True)

    # Chỉ giữ các cột cần thiết nếu có
    keep_cols = ["date", "close", "adj_close", "open", "high", "low",
"volume", "ticker"]
    df_vnindex = df_vnindex[[c for c in keep_cols if c in
df_vnindex.columns]]


    # Chuyển symbol ngay sau date
    cols = df_vnindex.columns.tolist()
    if "ticker" in cols:
        cols.remove("ticker")
        cols.insert(1, "ticker")
        df_vnindex = df_vnindex[cols]

    # Convert ngày tháng
    if "date" in df_vnindex.columns:
        df_vnindex["date"] = pd.to_datetime(df_vnindex["date"],
dayfirst=True, errors="coerce")

    # Convert numeric các cột số
    num_cols = ["close", "adj_close", "open", "high", "low", "volume",
"value"]
    for c in num_cols:
        if c in df_vnindex.columns:
            df_vnindex[c] = (
                df_vnindex[c].astype(str)
                      .str.replace(",", "", regex=False)
            )
            df_vnindex[c] = pd.to_numeric(df_vnindex[c], errors="coerce")

    # Xử lý missing
    missing_before = df_vnindex.isna().sum().sum()

    if missing_before > 0:
```

```
        print(f"[INFO] Missing detected: {missing_before} values →
applying cleaning")

        # Loại bỏ dòng không có ngày
        df_vnindex = df_vnindex.dropna(subset=['date'])

        # Forward fill
        df_vnindex = df_vnindex.ffill()

        # Drop lại phần còn NA (đầu chuỗi)
        df_vnindex = df_vnindex.dropna()

        missing_after = df_vnindex.isna().sum().sum()
        print(f"[INFO] Missing after cleaning: {missing_after}")

    return df_vnindex

vnindex_clean = preprocess_vnindex_data(vnindex_df)

vnindex_clean.to_csv("VNINDEX_clean_2020_2025.csv", index=False,
encoding="utf-8-sig")
print("Đã lưu file raw data: VNINDEX_clean_2020_2025.csv")

# Kiểm tra 5 dòng đầu
print("Đã xử lý dữ liệu VNINDEX")
print(vnindex_clean.head())

 # 2. XỬ LÝ DỮ LIỆU CHO RISK FREE RATE
# --- 0. Đọc và chuẩn hóa dữ liệu Risk-Free Rate ---
rf_file = "Risk_Free_Rate_2020-2025.csv"
rf_df = pd.read_csv(rf_file)

def preprocess_rf_data(rf_df):
    df_rf = rf_df.copy()

    # Chuẩn hóa tên cột
    rename_map = {
        "Ngày": "date",
        "Lần cuối": "rate"
    }
    df_rf.rename(columns=rename_map, inplace=True)

    # Giữ các cột cần thiết
    df_rf = df_rf[[c for c in ["date","rate"] if c in df_rf.columns]]

    # Convert ngày tháng
    df_rf["date"] = pd.to_datetime(df_rf["date"], dayfirst=True,
errors="coerce")

    # Convert rate sang decimal
    df_rf["rate"] =
df_rf["rate"].astype(str).str.replace('%','').str.replace(',','').astype(
float)/100

    # Sắp xếp theo ngày tăng dần
    df_rf = df_rf.sort_values('date').reset_index(drop=True)

    # Xử lý missing
    df_rf = df_rf.dropna(subset=['date', 'rate']).ffill().dropna()

    return df_rf
```

```
rf_clean = preprocess_rf_data(rf_df)

# Lưu file chuẩn hóa
rf_clean.to_csv("Risk_Free_Rate_clean_2020-2025.csv", index=False,
encoding="utf-8-sig")
print("Đã lưu Risk-Free Rate chuẩn hóa: Risk_Free_Rate_clean_2020-
2025.csv")
print(rf_clean.head())
```

Phần 2: File code để đọc dữ liệu và phân tích:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
from datetime import datetime
import statsmodels.api as sm
from statsmodels.tsa.arima.model import ARIMA
from pmdarima import auto_arima
from statsmodels.tsa.stattools import adfuller

# --- CẤU HÌNH OUTPUT ---
os.makedirs("output", exist_ok=True)

# --- 1. ĐỌC DỮ LIỆU ---

df_vn30 = pd.read_csv("VN30_clean_2020_2025.csv", parse_dates=["date"])
df_vnindex = pd.read_csv("VNINDEX_clean_2020_2025.csv",
parse_dates=["date"])

print("VN30 shape:", df_vn30.shape)
print("VNINDEX shape:", df_vnindex.shape)

# --- 2. PIVOT GIÁ ---


pivot_close_old = df_vn30.pivot(index="date", columns="ticker",
values="close").sort_index()
vnindex_close =
df_vnindex.set_index("date")["close"].sort_index().to_frame("VNINDEX")

# earliest date per ticker (dựa trên pivot_close)
first_dates = pivot_close_old.apply(lambda col: col.first_valid_index())
first_dates = first_dates.sort_values()
print("Ngày xuất hiện sớm nhất từng mã:\n", first_dates)

required_start = pd.to_datetime("2020-01-02")
insufficient = first_dates[first_dates > required_start].index.tolist()
sufficient = first_dates[first_dates <= required_start].index.tolist()

print("Mã thiếu dữ liệu (bắt đầu sau 2020-01-02):", insufficient)
print("Số mã đủ 5 năm:", len(sufficient))

pivot_close = pivot_close_old[sufficient].copy()
returns_monthly_full5 =
pivot_close.resample("ME").last().pct_change().dropna()
# dùng returns_monthly_full5 cho CAPM/cluster/danh mục


# Kiểm tra missing (điểm EDA)
missing_stats = pivot_close.isna().sum().sort_values(ascending=False)
```

```
print("\nMissing value top 10:")
print(missing_stats.head(10))
missing_stats.to_csv("output/missing_value_report.csv")

# --- 3. TÍNH LỢI SUẤT ---

returns_daily = pivot_close.pct_change(fill_method=None).dropna()
returns_monthly =
pivot_close.resample("ME").last().pct_change(fill_method=None).dropna()

market_returns_daily =
vnindex_close.pct_change(fill_method=None).dropna()
market_returns_monthly =
vnindex_close.resample("ME").last().pct_change(fill_method=None).dropna()

print("\nDaily returns (sample):")
print(returns_daily.head())

# Kiểm tra outliers (EDA)
zscore = (returns_daily - returns_daily.mean()) / returns_daily.std()
outliers = (np.abs(zscore) > 4).sum().sort_values(ascending=False)
outliers.to_csv("output/outlier_report_daily.csv")

# --- 4. THỐNG KÊ MÔ TẢ ---

summary_price = pivot_close.describe().T
summary_return = returns_daily.describe().T

summary_price.to_csv("output/summary_price.csv", encoding="utf-8-sig")
summary_return.to_csv("output/summary_daily_return.csv", encoding="utf-8-
sig")
print("Đã lưu summary_price.csv và summary_daily_return.csv")

# --- 5. HEATMAP TƯƠNG QUAN ---

plt.figure(figsize=(14,12))
sns.heatmap(pivot_close.corr(), cmap="coolwarm", linewidths=0.5)
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.title("Correlation Heatmap VN30", fontsize=16)
plt.tight_layout()
plt.savefig("output/VN30_correlation_heatmap.png", dpi=300)
plt.close()
print("Đã lưu VN30_correlation_heatmap.png")

# --- 6. BIỂU ĐỒ PRICE NORMALIZED ---

normalized = pivot_close / pivot_close.iloc[0]
plt.figure(figsize=(14,7))
for col in normalized.columns:
    plt.plot(normalized.index, normalized[col], linewidth=0.7, alpha=0.5)
plt.title("Normalized Price Trend (VN30)", fontsize=16)
plt.grid(alpha=0.3)
plt.tight_layout()
plt.savefig("output/VN30_normalized_trend.png", dpi=300)
plt.close()
print("Đã lưu VN30_normalized_trend.png")


# --- 7. ARIMA DỰ BÁO MÃ VJC  ---

# Lấy dữ liệu giá đóng cửa
```

```
vjc_price = pivot_close["VJC"].dropna().sort_index()

# Tính daily return
vjc_return = vjc_price.pct_change().dropna()

# Chuyển sang tần suất Business-Day
vjc_return = vjc_return.asfreq("B").ffill()

# ----------------- KIỂM ĐỊNH DỪNG (ADF) -----------------
adf_result = adfuller(vjc_return)
print("\nADF Statistic:", adf_result[0])
print("p-value:", adf_result[1])

with open("output/ADF_test_VJC.txt", "w") as f:
    f.write(f"ADF statistic: {adf_result[0]}\n")
    f.write(f"p-value: {adf_result[1]}\n")

# ----------------- AUTO ARIMA RETURN -----------------
arima_model = auto_arima(
    vjc_return,
    seasonal=False,
    trace=True,
    max_p=5,
    max_q=5,
    max_d=2,
    error_action="ignore",
    suppress_warnings=True
)

print("Best ARIMA model:\n", arima_model.summary())

# Fit model
model = ARIMA(vjc_return, order=arima_model.order)
model_fit = model.fit()

# ----------------- DỰ BÁO RETURN 30 NGÀY -----------------
fc_ret = model_fit.get_forecast(steps=30)
fc_df = fc_ret.summary_frame()

# Chuẩn hóa index forecast (theo ngày BD)
forecast_index = pd.date_range(
    start=vjc_return.index[-1] + pd.Timedelta(days=1),
    periods=30,
    freq="B"
)
fc_df.index = forecast_index

# ----------------- DỰ BÁO GIÁ TỪ RETURN -----------------
last_price = vjc_price.iloc[-1]
fc_price = last_price * (1 + fc_df["mean"]).cumprod()
fc_price.index = forecast_index  # đảm bảo trùng index

# ----------------- LƯU FILE -----------------
out = pd.DataFrame({
    "forecast_return": fc_df["mean"],
    "lower_ci": fc_df["mean_ci_lower"],
    "upper_ci": fc_df["mean_ci_upper"],
    "price_forecast": fc_price
})
out.to_csv("output/VJC_ARIMA_forecast_return.csv", encoding="utf-8-sig")

print("Đã lưu VJC_ARIMA_forecast_return.csv")
```

```
# ----------------- VẼ BIỂU ĐỒ LỢI SUẤT -----------------
plt.figure(figsize=(12,6))
plt.plot(vjc_return, label="Actual Return")
plt.plot(fc_df["mean"], label="Forecast Return", color="red")

plt.fill_between(
    fc_df.index,
    fc_df["mean_ci_lower"],
    fc_df["mean_ci_upper"],
    color='pink',
    alpha=0.3
)

plt.title("ARIMA Forecast – VJC Daily Return (30 Business Days Ahead)",
fontsize=14)
plt.legend()
plt.tight_layout()
plt.savefig("output/VJC_ARIMA_return_forecast.png", dpi=300)
plt.close()
print("Đã lưu VJC_ARIMA_return_forecast.png")


# ----------------- VẼ BIỂU ĐỒ GIÁ -----------------
plt.figure(figsize=(12,6))
plt.plot(vjc_price, label="Actual Price")
plt.plot(fc_price, label="Forecast Price", color="red")

plt.title("ARIMA Forecast – VJC Price (30 Business Days Ahead)",
fontsize=14)
plt.legend()
plt.tight_layout()
plt.savefig("output/VJC_ARIMA_price_forecast.png", dpi=300)
plt.close()

print("Đã lưu VJC_ARIMA_price_forecast.png")

# --- 8. CAPM HỒI QUY BETA CHUẨN ---

# Đọc risk-free
rf_df = pd.read_csv("Risk_Free_Rate_clean_2020-2025.csv",
parse_dates=["date"])
rf_df = rf_df.sort_values("date").set_index("date")

# Annual → Monthly rate
rf_df["rf_monthly"] = (1 + rf_df["rate"])**(1/12) - 1

# Đồng bộ mốc thời gian
rf_sync = rf_df["rf_monthly"].resample("ME").last()
common_dates =
returns_monthly.index.intersection(market_returns_monthly.index).intersec
tion(rf_sync.index)

ret_m = returns_monthly.loc[common_dates]
mkt_m = market_returns_monthly.loc[common_dates]
rf_m = rf_sync.loc[common_dates]

# Excess return
excess_stock = ret_m.subtract(rf_m, axis=0)
excess_mkt = (mkt_m["VNINDEX"] - rf_m).rename("MKT")
```

```
# --- Hồi quy CAPM ---
capm_rows = []

for ticker in excess_stock.columns:
    df_capm = pd.concat([excess_stock[ticker], excess_mkt],
axis=1).dropna()
    X = sm.add_constant(df_capm["MKT"])
    y = df_capm[ticker]
    model = sm.OLS(y, X).fit()

    capm_rows.append({
        "Ticker": ticker,
        "Alpha": model.params["const"],
        "Beta": model.params["MKT"],
        "Alpha_tstat": model.tvalues["const"],
        "Beta_tstat": model.tvalues["MKT"],
        "Alpha_pvalue": model.pvalues["const"],
        "Beta_pvalue": model.pvalues["MKT"],
        "R2": model.rsquared,
        "Adj_R2": model.rsquared_adj,
        "N_obs": int(model.nobs)
    })

capm_df = pd.DataFrame(capm_rows)
capm_df.to_csv("output/CAPM_results_realRF.csv", index=False,
encoding="utf-8-sig")

print("\nĐã lưu CAPM_results_realRF.csv")
print(capm_df.head())


# --- 9. DANH MỤC THEO BETA & ĐÁNH GIÁ HIỆU QUẢ (CHỈ 2 DANH MỤC THEO ĐỀ
BÀI) ---

# Chia nhóm theo Beta
high_beta = capm_df[capm_df["Beta"] > 1]["Ticker"].tolist()      #
Aggressive
low_beta  = capm_df[capm_df["Beta"] <= 1]["Ticker"].tolist()     # Stable

print("Aggressive (High β):", high_beta)
print("Stable (Low β):", low_beta)

# --- Hàm tính metrics ---
def portfolio_metrics_weighted(tickers, returns_df, rf=0):
    """
    Tính metrics cho danh mục:
    - Cumulative return (1/N weighted)
    - Annualized mean return
    - Annualized volatility
    - Sharpe ratio
    - Max drawdown
    """
    data = returns_df[tickers].dropna()
    weights = np.ones(len(tickers)) / len(tickers)  # 1/N weight
    port_return = (data * weights).sum(axis=1)

    # Annualized metrics
    mean_ret = port_return.mean() * 12
    vol = port_return.std() * np.sqrt(12)
    sharpe = (mean_ret - rf) / vol
    cum_ret = (1 + port_return).cumprod()
```

```
    # Max drawdown
    roll_max = cum_ret.cummax()
    drawdown = (cum_ret - roll_max) / roll_max
    max_dd = drawdown.min()

    return mean_ret, vol, sharpe, cum_ret, max_dd


# Tính metrics cho hai danh mục
metrics = {}
portfolio_groups = {
    "Aggressive (High β)": high_beta,
    "Stable (Low β)": low_beta
}

for name, group in portfolio_groups.items():
    if len(group) == 0:
        continue
    mean_ret, vol, sharpe, cum_ret, max_dd = portfolio_metrics_weighted(
        group,
        ret_m,
        rf=rf_m.mean()
    )
    metrics[name] = {
        "Annualized Return": mean_ret,
        "Volatility": vol,
        "Sharpe Ratio": sharpe,
        "Cumulative Return": cum_ret,
        "Max Drawdown": max_dd
    }

# --- Lưu summary metrics ---
metrics_summary = pd.DataFrame([
    {
        "Portfolio": k,
        "Annualized Return": v["Annualized Return"],
        "Volatility": v["Volatility"],
        "Sharpe Ratio": v["Sharpe Ratio"],
        "Max Drawdown": v["Max Drawdown"]
    }
    for k, v in metrics.items()
])

metrics_summary.to_csv("output/Portfolio_metrics_complete.csv",
index=False, encoding="utf-8-sig")
print("Đã lưu Portfolio_metrics_complete.csv")
print(metrics_summary)

# --- Vẽ cumulative return chart ---
plt.figure(figsize=(12,6))
for name, v in metrics.items():
    plt.plot(v["Cumulative Return"], label=name)
plt.plot((1 + mkt_m["VNINDEX"]).cumprod(), label="VNINDEX",
color="black", linestyle="--")
plt.title("Cumulative Return: Stable vs Aggressive Portfolio",
fontsize=14)
plt.legend()
plt.grid(alpha=0.3)
plt.tight_layout()
plt.savefig("output/Portfolio_cumulative_return_complete.png", dpi=300)
plt.close()
print("Đã lưu Portfolio_cumulative_return_complete.png")
```

2. **Dữ liệu (Data)**

- *"Risk_Free_Rate_2020-2025.csv":* Dữ liệu thực Lãi suất phi rủi ro, lấy từ dữ liệu lịch sử suất thu lợi trái phiếu chính phủ Việt Nam kỳ hạn 1 năm, giai đoạn 01/01/2020 - 21/11/2025
- *"VN30_raw_backup_2020_2025.csv":* Dữ liệu thô về lịch sử giá chứng khoán theo ngày của các mã trong VN30 giai đoạn 01/01/2020 - 21/11/2025, dùng làm backup cho trường hợp lỗi lấy dữ liệu.
- *" VNINDEX_raw_2020_2025.csv":* Dữ liệu thô về lịch sử chỉ số giá của VN-INDEX giai đoạn 01/01/2020 - 21/11/2025, dùng làm backup cho trường hợp lỗi lấy dữ liệu.
- *"VN30_raw_2020_2025.csv" và "VNINDEX_raw_2020_2025.csv":* Dữ liệu thô xuất sau khi chạy code cào dữ liệu.
- *"VN30_clean_2020_2025.csv", "VNINDEX_clean_2020_2025.csv", "Risk_Free_Rate_clean_2020-2025.csv":* Dữ liệu sau xử lý, sẵn sàng để sử dụng trong chạy code phân tích.
- *Thư mục "Output":* Bao gồm các biểu đồ và file bảng số liệu, là kết quả phân tích được xuất ra sau khi chạy code phân tích.

3. **Nguồn dữ liệu**

- Nguồn dữ liệu Giá cổ phiếu VN30 và VN-Index: API CafeF (https://s.cafef.vn/Ajax/PageNew/DataHistory/PriceHistory.ashx)
- Nguồn dữ liệu Lãi suất phi rủi ro: Dữ liệu lịch sử suất thu lợi trái phiếu chính phủ Việt Nam 1 năm – vn.investing.com (https://vn.investing.com/rates-bonds/vietnam-1-year-bond-yield)