1. Data Preprocessing
คำอธิบาย:
เริ่มต้นด้วยการจัดการคุณภาพของข้อมูล เช่น:

เติมค่าที่หายไป (missing values)
กำจัดค่าผิดปกติ (outliers)
ปรับขนาดข้อมูล (normalization) เพื่อให้เหมาะสมกับการฝึกโมเดล
การตีความ:
ขั้นตอนนี้ช่วยให้ข้อมูลมีความสะอาดและมีรูปแบบที่เหมาะสมสำหรับการเรียนรู้ของโมเดล โดยเฉพาะ normalization ที่สำคัญมากสำหรับโมเดลที่ใช้ gradient-based optimization เช่น neural networks

🔹 2. Data Partitioning
คำอธิบาย:
แบ่งข้อมูลย้อนหลังออกเป็นชุด training, validation และ testing โดยใช้วิธี rolling window เพื่อจำลองสถานการณ์การเทรดจริง

การตีความ:
การใช้ rolling window ช่วยให้โมเดลสามารถเรียนรู้จากข้อมูลในอดีตและทดสอบกับข้อมูลในอนาคตอย่างต่อเนื่อง ซึ่งสะท้อนลักษณะของตลาดที่เปลี่ยนแปลงตามเวลา

🔹 3. Model Training and Evaluation
คำอธิบาย:
ฝึกโมเดลหลายแบบ พร้อมปรับแต่ง hyperparameters และใช้เทคนิค validation เพื่อเพิ่มความแม่นยำในการพยากรณ์

การตีความ:
การใช้หลายโมเดล (เช่น regression, LSTM, Transformer ฯลฯ) และการปรับ hyperparameters ช่วยให้ได้ผลลัพธ์ที่ดีที่สุดจากแต่ละโมเดล และสามารถเปรียบเทียบประสิทธิภาพได้อย่างเป็นระบบ

🔹 4. Feature Importance Analysis
คำอธิบาย:
ใช้ SHAP values และ permutation importance เพื่อวัดความสำคัญของแต่ละตัวชี้วัดทางเทคนิคในแต่ละโมเดล

การตีความ:
SHAP ให้ข้อมูลเชิงลึกว่าแต่ละ feature มีผลต่อการตัดสินใจของโมเดลอย่างไร ส่วน permutation importance วัดผลกระทบของการสับค่าของ feature ต่อประสิทธิภาพของโมเดล

🔹 5. Consensus Indicator Selection
คำอธิบาย:
รวมผลการวิเคราะห์จากหลายโมเดลเพื่อเลือกตัวชี้วัดที่มีผลกระทบสูงและปรากฏอย่างสม่ำเสมอในหลายโมเดล

การตีความ:

ขั้นตอนนี้ช่วยให้ได้ชุดตัวชี้วัดที่มีความน่าเชื่อถือสูงและมีความสำคัญต่อการพยากรณ์ในหลายบริบทของโมเดล

✅ สรุปภาพรวม
Framework นี้เป็นแนวทางที่มีโครงสร้างชัดเจนและครอบคลุมทุกขั้นตอน ตั้งแต่การเตรียมข้อมูลไปจนถึงการเลือกตัวชี้วัดที่ดีที่สุด โดยเน้นการประเมินแบบหลายมิติและการจำลองสถานการณ์จริง
เพื่อคัดเลือกตัวชี้วัดทางเทคนิคที่มีผลต่อการพยากรณ์ราคาหุ้น S&P 500 โดยใช้ข้อมูล OHLCV (Open, High, Low, Close, Volume) และประเมินประสิทธิภาพของโมเดลหลายประเภท โดยแต่ละขั้นตอนมีเครื่องมือและเหตุผลรองรับดังนี้:

🔷 1. Preprocess Data
✅ ขั้นตอน:
Linear Interpolation: เติมค่าที่ขาดหายไปอย่างต่อเนื่อง
IQR-based Filtering: กำจัด outliers โดยใช้ Interquartile Range
Min-Max Scaling: ปรับขนาดข้อมูลให้อยู่ในช่วง [0,1]
PCA (Principal Component Analysis): ลดมิติข้อมูลโดยคงไว้ 95% ของความแปรปรวน
🛠️ เครื่องมือที่ใช้:
pandas สำหรับจัดการ missing values
scikit-learn สำหรับ scaling และ PCA
numpy สำหรับการคำนวณ IQR
📌 ทำไมต้องใช้:
เพื่อให้ข้อมูลสะอาดและเหมาะสมกับการฝึกโมเดล
ลดความซับซ้อนของข้อมูลโดยไม่สูญเสียสาระสำคัญ
ป้องกันโมเดลจากการเรียนรู้ noise หรือข้อมูลผิดปกติ
🔷 2. Split Data with Rolling Windows
✅ ขั้นตอน:
แบ่งข้อมูลเป็น training (70%), validation (15%), testing (15%)
ใช้ rolling window ขนาด 60 วัน เลื่อนทีละ 1 วัน
🛠️ เครื่องมือที่ใช้:
Custom Python function หรือ TimeSeriesSplit จาก scikit-learn
ใช้ pandas เพื่อจัดการช่วงเวลา
📌 ทำไมต้องใช้:
จำลองสถานการณ์การเทรดจริงที่ข้อมูลใหม่เข้ามาเรื่อย ๆ
ป้องกัน data leakage และให้โมเดลเรียนรู้แบบ time-aware
🔷 3. Model Training and Evaluation
✅ ขั้นตอน:
ใช้โมเดล: SVR, XGBoost, Random Forest (RF), LSTM
ปรับ hyperparameters ด้วย Bayesian Optimization (50 รอบ)
ฝึกโมเดลทั้งแบบ lagged (Close(t-1)) และ non-lagged (Close(t))
ใช้ walk-forward cross-validation
🛠️ เครื่องมือที่ใช้:
scikit-learn สำหรับ SVR, RF
xgboost สำหรับ XGBoost
TensorFlow หรือ PyTorch สำหรับ LSTM

optuna หรือ bayes_opt สำหรับ Bayesian Optimization
📌 ทำไมต้องใช้:
เพื่อเปรียบเทียบโมเดลทั้งแบบ traditional และ deep learning
Bayesian Optimization มีประสิทธิภาพสูงในการค้นหา hyperparameters ที่ดีที่สุด
Walk-forward CV เหมาะกับข้อมูล time series ที่ไม่สามารถสับแบบ random ได้
🔷 4. Feature Importance Analysis
✅ ขั้นตอน:
ใช้ Permutation Importance เพื่อจัดอันดับความสำคัญของตัวชี้วัด
🛠️ เครื่องมือที่ใช้:
eli5 หรือ scikit-learn สำหรับ permutation importance
SHAP (สามารถใช้เพิ่มเติมได้) เพื่อวิเคราะห์เชิงลึก
📌 ทำไมต้องใช้:
เพื่อเข้าใจว่า indicator ใดมีผลต่อการพยากรณ์มากที่สุด
ช่วยในการคัดเลือก feature ที่มีประโยชน์จริง
🔷 5. Consensus Indicator Selection
✅ ขั้นตอน:
รวมผลจากหลายโมเดล
เลือก indicators ที่ปรากฏใน ≥ 3 โมเดล
🛠️ เครื่องมือที่ใช้:
Python dictionary หรือ DataFrame สำหรับ aggregation
numpy/pandas สำหรับการนับและจัดกลุ่ม
📌 ทำไมต้องใช้:
เพื่อให้ได้ชุด indicators ที่มีความน่าเชื่อถือสูง
ลดความเสี่ยงจากการเลือก feature ที่เฉพาะเจาะจงกับโมเดลใดโมเดลหนึ่ง
🎯 ผลลัพธ์ที่ได้
Selected Key Indicators: ตัวชี้วัดที่มีผลต่อการพยากรณ์มากที่สุด
Model Performance Metrics: เช่น RMSE, MAE, R^2, Accuracy ฯลฯ
3.1. Experiment Setup Details
✅ ข้อมูลพื้นฐาน
ช่วงเวลา: 01/01/1950 – 12/07/2023 (รวม 18,137 วันซื้อขาย)
Target Variable: ราคาปิดรายวันของ S&P 500
Feature Set: เริ่มต้น 88 indicators → ลดเหลือ 35 indicators หลัง PCA
🔹 ขั้นตอนที่ 1: การเตรียมข้อมูล (Preprocessing)
🛠️ เครื่องมือ:
Linear Interpolation: เติมค่าที่หายไปอย่างต่อเนื่อง
IQR Filtering: กำจัด outliers โดยใช้ช่วงระหว่าง Q1 และ Q3
Min-Max Scaling: ปรับขนาดข้อมูลให้อยู่ในช่วง [0,1]
PCA (Principal Component Analysis): ลดมิติข้อมูลโดยคงไว้ 95% ของความแปรปรวน
📌 เหตุผล:
ลด noise และความซับซ้อนของข้อมูล
ปรับข้อมูลให้อยู่ในรูปแบบที่เหมาะสมกับโมเดล ML/DL
PCA ช่วยลด multicollinearity และเพิ่มความเร็วในการฝึกโมเดล
🔹 ขั้นตอนที่ 2: การแบ่งข้อมูล (Data Splitting)
🛠️ วิธีการ:

Training Set: Jan 1950 – Jun 2012
Testing Set: Jun 2012 – Jun 2023
สัดส่วน: ประมาณ 6:1
Rolling Window: ขนาด 60 วัน, เลื่อนทีละ 1 วัน
📌 เหตุผล:
การแบ่งแบบ time-based ป้องกัน data leakage
Rolling window ช่วยให้โมเดลเรียนรู้จากข้อมูลล่าสุดเสมอ เหมือนการเทรดจริง
เพิ่มความแม่นยำในการประเมินโมเดลในสภาพแวดล้อมที่เปลี่ยนแปลง
🔹 ขั้นตอนที่ 3: การฝึกและประเมินโมเดล (Model Training & Evaluation)
🧠 โมเดลที่ใช้:
SVR (Support Vector Regression)
XGBoost (Extreme Gradient Boosting)
RF (Random Forest)
LSTM (Long Short-Term Memory)
🛠️ เทคนิคที่ใช้:
Bayesian Optimization (50 iterations): ปรับ hyperparameters อย่างมีประสิทธิภาพ
Lagged vs Non-Lagged Targets: เปรียบเทียบการใช้ Close(t-1) กับ Close(t)
Walk-forward Cross-validation: ประเมินโมเดลแบบ time-aware
📌 เหตุผล:
ใช้โมเดลหลากหลายเพื่อเปรียบเทียบความสามารถในการจับ pattern
Bayesian Optimization มีประสิทธิภาพสูงกว่ Grid/Random Search
LSTM เหมาะกับข้อมูล time series ที่มี dependency ตามเวลา
🔹 Hyperparameter Configuration
🔧 XGBoost:
n_estimators = 200
max_depth = 6
learning_rate = 0.01
🔧 LSTM:
2 layers × 64 units
dropout = 0.2
epochs = 100
📌 เหตุผล:
XGBoost: ปรับสมดุลระหว่างความซับซ้อนและความเสถียร
LSTM: โครงสร้างลึกพอที่จะจับ pattern ระยะยาว, dropout ป้องกัน overfitting
🔹 ขั้นตอนที่ 4: Feature Importance Analysis
🛠️ เครื่องมือ:
Permutation Importance: วัดผลกระทบของการสับค่าของ feature ต่อ performance
(สามารถใช้ SHAP เพิ่มเติมเพื่อวิเคราะห์เชิงลึก)
📌 เหตุผล:
ช่วยให้เข้าใจว่า indicators ใดมีผลต่อการพยากรณ์มากที่สุด
เป็นพื้นฐานสำหรับการคัดเลือก indicators ที่มีคุณภาพ
🔹 ขั้นตอนที่ 5: Consensus Indicator Selection
✅ วิธีการ:
รวมผลจากหลายโมเดล

เลือก indicators ที่ปรากฏใน ≥ 3 โมเดล
📌 เหตุผล:
ลด bias จากโมเดลใดโมเดลหนึ่ง
ได้ชุด indicators ที่มีความน่าเชื่อถือสูงและมีผลต่อการพยากรณ์จริง
🎯 ผลลัพธ์ที่คาดหวัง
Selected Key Indicators: ตัวชี้วัดที่มีผลต่อราคาปิดของ S&P 500
Model Performance Metrics: เช่น RMSE, MAE, R^2 ฯลฯ

3.2. Preprocessing Phase Expansion
✅ 1. การจัดการ Missing Values
พบ Missing Values 2.1% โดยเฉพาะใน indicators เช่น:
Average True Range (ATR)
Relative Vigor Index (RVI)
ช่วงที่เกิด: ต้นทศวรรษ 2010s
วิธีแก้ไข: ใช้ Linear Interpolation เพื่อเติมค่าที่ขาดหายไป
🛠️ เครื่องมือที่ใช้:
pandas.DataFrame.interpolate(method='linear')
📌 เหตุผล:
Linear interpolation เหมาะกับข้อมูล time series เพราะรักษาความต่อเนื่องของแนวโน้ม
ป้องกันการสูญเสียข้อมูลสำคัญที่อาจส่งผลต่อการเรียนรู้ของโมเดล
✅ 2. การจัดการ Outliers
พบ Outliers 1.5% โดยมีสาเหตุจากเหตุการณ์ตลาดสำคัญ เช่น:
2015 Swiss Franc (CHF) Crisis
2020 COVID-19 Market Crash
วิธีแก้ไข: ใช้ Interquartile Range (IQR) Filtering
🛠️ เครื่องมือที่ใช้:

📌 เหตุผล:
IQR filtering เป็นวิธี robust ที่ไม่อ่อนไหวต่อการกระจายของข้อมูล
ช่วยลดผลกระทบของเหตุการณ์สุดโต่งที่อาจทำให้โมเดลเรียนรู้ผิดพลาด
✅ 3. การลดจำนวน Features
🔹 3.1. ตัด Indicators ที่มี Variance ต่ำ
จำนวนที่ตัดออก: 12 ตัว เช่น:
Chaikin Oscillator
เหตุผล: Indicators เหล่านี้มีการเปลี่ยนแปลงน้อย → ไม่ให้ข้อมูลใหม่แก่โมเดล
🔹 3.2. ตัด Indicators ที่มีความสัมพันธ์สูง
จำนวนที่ตัดออก: 8 ตัว
เกณฑ์: Pearson's correlation coefficient $|r| > 0.95$
เหตุผล: ป้องกัน Multicollinearity ซึ่งอาจทำให้โมเดลเรียนรู้ซ้ำซ้อน
🛠️ เครื่องมือที่ใช้:
pandas.DataFrame.var() สำหรับ variance
pandas.DataFrame.corr(method='pearson') สำหรับ correlation matrix
📌 เหตุผล:
ลดความซับซ้อนของข้อมูล

เพิ่มความสามารถในการตีความผลลัพธ์ของโมเดล
ปรับปรุงความเสถียรของการฝึกโมเดล โดยลดการเรียนรู้ซ้ำซ้อน
✅ 4. การลดมิติด้วย PCA
เป้าหมาย: คงไว้ 95% ของความแปรปรวน
ผลลัพธ์: ลดจาก 88 → 35 indicators
🛠️ เครื่องมือที่ใช้:

📌 เหตุผล:
ลดมิติข้อมูลโดยไม่สูญเสียสาระสำคัญ
เพิ่มความเร็วในการฝึกโมเดล
ลดความเสี่ยงจาก overfitting
🎯 สรุปผลของ Preprocessing
ข้อมูลมีความต่อเนื่องและสะอาด
ลด noise และความซ้ำซ้อน
ได้ชุด indicators ที่มีคุณภาพสูงและเหมาะสมกับการพย
🔷 3.3. Algorithm Selection Justification
✅ หลักการเลือกอัลกอริธึม
การเลือกอัลกอริธึมในงานวิจัยนี้มีเป้าหมายเพื่อ:

รองรับความซับซ้อนของข้อมูล Time Series ทางการเงิน
เปรียบเทียบโมเดลที่มีลักษณะต่างกันทั้งเชิงสถิติ, เชิงโครงสร้าง, และเชิงลึก
ลดความลำเอียงจากการพึ่งพาโมเดลใดโมเดลหนึ่งมากเกินไป (model agnostic)
🔹 1. Support Vector Regression (SVR)
🧠 ลักษณะเด่น:
เป็น baseline model ที่ดีสำหรับการจับความสัมพันธ์เชิงเส้น
มีความสามารถในการใช้ kernel (เช่น RBF) เพื่อจัดการกับความไม่เป็นเชิงเส้น
📌 เหตุผลที่เลือก:
ใช้เป็นจุดเริ่มต้นในการเปรียบเทียบกับโมเดลที่ซับซ้อนกว่า
มีความเสถียรและไม่ไวต่อ outliers มากนัก
เหมาะกับข้อมูลที่มี noise ปานกลาง
🔹 2. XGBoost (Extreme Gradient Boosting)
🧠 ลักษณะเด่น:
เป็นโมเดลแบบ ensemble ที่ใช้ boosting เพื่อปรับปรุงข้อผิดพลาดของโมเดลก่อนหน้า
มีการควบคุม overfitting ที่ดีผ่าน regularization
📌 เหตุผลที่เลือก:
มีความสามารถในการจัดการกับความไม่เป็นเชิงเส้นและความสัมพันธ์ซับซ้อน
ทนต่อ noise และ missing values ได้ดี
มีประสิทธิภาพสูงในการพยากรณ์ข้อมูลทางการเงิน (อ้างอิงจาก Zhao et al., 2024)
🔹 3. Random Forest (RF)
🧠 ลักษณะเด่น:
เป็นโมเดลแบบ ensemble ที่ใช้การสุ่มเลือก feature และข้อมูลเพื่อสร้างหลาย decision trees
มีความสามารถในการจัดการกับข้อมูลที่มีความแปรปรวนสูง
📌 เหตุผลที่เลือก:
Robust ต่อ outliers และ noise

ไม่ต้องปรับ hyperparameters มากเท่า XGBoost
ให้ผลลัพธ์ที่ตีความได้ง่ายผ่าน feature importance
🔹 4. Long Short-Term Memory (LSTM)
🧠 ลักษณะเด่น:
เป็น neural network ที่ออกแบบมาเพื่อจัดการกับลำดับข้อมูล (sequence)
มีหน่วยความจำระยะยาวและระยะสั้น → เหมาะกับข้อมูล Time Series
📌 เหตุผลที่เลือก:
สามารถจับ temporal dependencies ได้ดี เช่น momentum, volatility clustering
เหมาะกับการพยากรณ์ราคาหุ้นที่มีลักษณะเปลี่ยนแปลงตามเวลา
ได้รับการสนับสนุนจากงานวิจัยล่าสุด (Nazareth & Reddy, 2023)
🔹 5. ความหลากหลายของโมเดล (Model-Agnostic Design)
📌 เหตุผล:
การใช้โมเดลหลายประเภทช่วยลดความลำเอียงจากการพึ่งพาโมเดลเดียว
ทำให้ผลการวิเคราะห์มีความน่าเชื่อถือมากขึ้น
สนับสนุนการเลือก indicators ที่มีผลจริงในหลายบริบทของโมเดล (Pérez-Hernández & Arévalo-de-Pablos, 2024)
🎯 สรุปภาพรวม

Algorithm	จุดเด่น	เหตุผลที่เลือก
SVR	Linear + Kernel	Baseline, เสถียร
XGBoost	Boosting, Regularization	จัดการ non-linear ได้ดี
RF	Ensemble, Robust	ทน noise, ตีความง่าย
LSTM	Temporal Memory	เหมาะกับ Time Series

🔷 3.4. ข้อจำกัดของแนวทางที่เสนอ (Proposed Approach Limitations)
✅ 1. Temporal Bias จากช่วงเวลาการฝึกโมเดล
รายละเอียด: โมเดลถูกฝึกด้วยข้อมูลตั้งแต่ปี 2010–2023 ซึ่งอาจไม่ครอบคลุมช่วงที่ตลาดมีโครงสร้างเปลี่ยนแปลง เช่น:
High-inflation regimes
Policy shifts หรือ macroeconomic shocks
📌 ผลกระทบ:
โมเดลอาจเรียนรู้ pattern ที่เฉพาะเจาะจงกับช่วงเวลานั้น
ไม่สามารถ generalize ไปยังช่วงเวลาที่มีลักษณะตลาดต่างออกไป เช่น ยุคก่อนปี 2008 หรือช่วง stagflation
🛠️ แนวทางแก้ไขที่อาจใช้:
เพิ่มข้อมูลจากช่วงก่อนปี 2010 เพื่อให้โมเดลเห็นความหลากหลายของ regime
ใช้ regime-switching models หรือ context-aware features เช่น inflation dummy, volatility regime
✅ 2. ความเสี่ยงจากการใช้ PCA ในการลดมิติ
รายละเอียด: PCA ถูกใช้เพื่อลด feature จาก 88 → 35 โดยคงไว้ 95% ของความแปรปรวน
ข้อจำกัด: Indicators ที่มี variance ต่ำอาจถูกตัดออก แม้จะมีความสำคัญเชิงกลยุทธ์ เช่น:
Chaikin Oscillator
บาง momentum indicators
📌 ผลกระทบ:
สูญเสีย granularity ที่อาจมีผลต่อการพยากรณ์ในช่วงตลาดเฉพาะ
ลดความสามารถในการตีความผลลัพธ์ของโมเดล

🛠️ แนวทางแก้ไขที่อาจใช้:
ใช้ feature selection แบบ hybrid เช่น SHAP + PCA
พิจารณาใช้ domain knowledge เพื่อคง indicators ที่มีความสำคัญแม้ variance ต่ำ
✅ 3. ความซับซ้อนของโมเดล LSTM
รายละเอียด: LSTM มีโครงสร้างซับซ้อน ต้องการการปรับ hyperparameters อย่างละเอียด เช่น:

จำนวน layers
จำนวน units
dropout rate
learning rate
หากไม่มีการ regularize อย่างเหมาะสม เช่น dropout หรือ early stopping → เสี่ยงต่อ overfitting

📌 ผลกระทบ:
โมเดลอาจเรียนรู้ noise แทน pattern จริง
ประสิทธิภาพลดลงเมื่อใช้กับข้อมูลใหม่ที่ไม่เคยเห็นมาก่อน
🛠️ แนวทางแก้ไขที่อาจใช้:
ใช้ Bayesian Optimization ร่วมกับ cross-validation
เพิ่ม dropout, batch normalization, และ early stopping
เปรียบเทียบกับโมเดลที่ง่ายกว่า เช่น GRU หรือ Temporal Convolutional Networks (TCN)
🎯 สรุปภาพรวมข้อจำกัด
หมวด ข้อจำกัด ผลกระทบ แนวทางแก้ไข
Temporal Bias ข้อมูลฝึกจำกัดช่วงเวลา ไม่ครอบคลุม regime ต่าง ๆ เพิ่มช่วงเวลา, ใช้ regime-aware features
PCA Granularity Loss ตัด indicators ที่ variance ต่ำ สูญเสียข้อมูลเชิงกลยุทธ์ ใช้ hybrid selection, domain knowledge
LSTM Complexity ต้องปรับ hyperparameters มาก เสี่ยง overfitting ใช้ regularization, เปรียบเทียบกับโมเดลอื่น

🔷 3.5. Technical Indicators
✅ หลักการพื้นฐานของการวิเคราะห์ทางเทคนิค
การวิเคราะห์ทางเทคนิคตั้งอยู่บนสมมติฐานว่า ราคาหุ้นเคลื่อนไหวตามแนวโน้ม (trend)
แนวโน้มเหล่านี้เกิดจากพฤติกรรมการซื้อขายที่มีรูปแบบของนักลงทุน (Fang, Qin, & Jacobsen, 2014)
ต่างจากปัจจัยพื้นฐาน เช่น กำไรหรือรายได้ → ตัวชี้วัดทางเทคนิคเป็นการประมาณเชิงสถิติจากราคาและปริมาณ
✅ บทบาทของ Indicators ในการพยากรณ์
ใช้ข้อมูลในอดีตเพื่อระบุรูปแบบราคา เช่น:

แนวโน้ม (Trend)
โมเมนตัม (Momentum)
ความผันผวน (Volatility)
ปริมาณการซื้อขาย (Volume)
(Ahn et al., 2003; Bâra & Oprea, 2024; Jiang et al., 2020; Salim & Djunaidy, 2024)
นักเทรดระยะสั้นใช้เพื่อจับจังหวะการเคลื่อนไหว

นักลงทุนระยะยาวใช้เพื่อหาจังหวะซื้อขายที่เหมาะสม

✅ การจัดกลุ่ม Indicators ทั้ง 4 ประเภทหลัก (van der Hagen, 2021)
กลุ่ม ตัวอย่างแนวคิดหลัก
Trend / Moving Averages SMA, EMA, MACD ราคามีแนวโน้มต่อเนื่อง
Volatility ATR, Bollinger Bands วัดความผันผวนของราคา
Momentum RSI, Stochastic Oscillator วัดแรงขับเคลื่อนของราคา
Volume OBV, Chaikin Money Flow วัดแรงสนับสนุนจากปริมาณการซื้อขาย
✅ การสร้าง Indicators ในงานวิจัย
ใช้ข้อมูล OHLCV จาก Yahoo Finance
สร้างทั้งหมด 88 ตัวชี้วัด โดยใช้ Python library: TA-Lib
ผ่านขั้นตอน PCA เพื่อคัดเลือก → เหลือ 35 ตัวชี้วัดสุดท้าย
🛠️ เครื่องมือที่ใช้:

Indicators ที่มีหลาย output เช่น MACD (มี MACD line, Signal line, Histogram) → ถูกเน้นในตาราง (Table 2)
✅ การคัดเลือก Indicators ที่สำคัญ
ใช้ Permutation Importance และ SHAP เพื่อวัดความสำคัญ
Indicators ที่ถูกเลือกมากที่สุดจะมีสูตรแสดงไว้ในตารางประกอบ
🎯 สรุปภาพรวม
การวิเคราะห์ทางเทคนิคช่วยให้เข้าใจพฤติกรรมราคาหุ้นจากมุมมองเชิงสถิติ
การจัดกลุ่ม indicators ช่วยให้โมเดลเรียนรู้จากหลายมิติของตลาด
การใช้ TA-Lib และ PCA ทำให้ได้ชุดข้อมูลที่มีคุณภาพและเหมาะสมกับการพยากรณ์

3.6. Data Normalization
✅ 1. ความจำเป็นของการ Preprocessing ก่อนใช้โมเดล ML
ก่อนนำข้อมูลเข้าโมเดล ML จำเป็นต้องผ่านขั้นตอน Data Cleaning และ Normalization
ในขั้นตอน Data Cleaning:
ลบฟีเจอร์ที่ไม่จำเป็น เช่น Adjusted Closing Price เนื่องจากข้อมูลนี้ซ้ำซ้อนกับ Closing Price
ลดความซับซ้อนของข้อมูลเพื่อให้โมเดลเรียนรู้ได้ดีขึ้น
✅ 2. ความสำคัญของการ Normalization
Normalization คือการปรับขนาดข้อมูลให้อยู่ในช่วงที่เหมาะสม เช่น [0, 1]
ช่วยให้:
Gradient Descent ในโมเดล ML/DL ทำงานได้เร็วขึ้น (Jin et al., 2023)
ป้องกันการ bias จาก scale ที่ต่างกันของ indicators (Chung & Shin, 2018)
รักษาความสัมพันธ์เชิงโครงสร้างของข้อมูลโดยไม่บิดเบือน
✅ 3. วิธีที่ใช้ในงานวิจัยนี้: Min-Max Normalization
ใช้สูตร Min-Max normalization ตามสมการ (1):
x
normalized
=
x
−
x

min
x
max
−
x
min
x
normalized

=
x
max

−x
min

x−x
min

โดยที่:
x
x คือค่าของ feature
x
min
x
min

,
x
max
x
max

คือค่าต่ำสุดและสูงสุดของ feature นั้น
🛠️ เครื่องมือที่ใช้:

✅ 4. ผลกระทบเชิงบวกของ Normalization
ลดความเสี่ยงจากการที่โมเดลให้ความสำคัญกับ indicators ที่มี scale ใหญ่เกินไป
ทำให้การฝึกโมเดลมีความเสถียรและรวดเร็ว
ช่วยให้การเปรียบเทียบ indicators เป็นธรรมมากขึ้น

🎯 สรุปภาพรวม

ขั้นตอน	รายละเอียด	ผลกระทบ
Data Cleaning	ลบฟีเจอร์ซ้ำซ้อน เช่น Adjusted Close	ลด noise และความซับซ้อน
Normalization	ใช้ Min-Max scaling	เพิ่มความเร็วและเสถียรของการฝึกโมเดล
ผลลัพธ์	ข้อมูลพร้อมสำหรับ ML/DL	ปรับปรุงประสิทธิภาพการพยากรณ์

🔷 3.7. Feature Selection with PCA

✅ วัตถุประสงค์ของการใช้ PCA

ลดจำนวนตัวชี้วัดทางเทคนิคจาก 88 ตัว → เหลือ 35 ตัว

คงไว้ 95% ของความแปรปรวน เพื่อรักษาข้อมูลสำคัญ

เพิ่ม ความเร็วในการประมวลผล และลด ความซับซ้อนของโมเดล

แก้ปัญหา Multicollinearity ที่เกิดจาก indicators ที่มีความสัมพันธ์สูง

✅ ขั้นตอนการดำเนินการ PCA

🔹 1. จัดการ Missing Values

พบ Missing Values 2.1% ในช่วงต้นปี 2010s

ใช้ Linear Interpolation เพื่อเติมค่าที่ขาดหายไป

🔹 2. Standardization

ก่อนทำ PCA ต้องปรับข้อมูลให้มีค่าเฉลี่ย = 0 และส่วนเบี่ยงเบนมาตรฐาน = 1

ใช้สูตร:

$$z = x - \mu$$

$$z = \sigma$$

$$x - \mu$$

เพื่อให้ PCA ไม่ถูก bias จาก scale ของ indicators

🔹 3. การสร้าง Correlation Matrix

สร้าง เมทริกซ์ความสัมพันธ์ (Pearson’s r) ระหว่าง indicators

จากนั้น Diagonalize เมทริกซ์เพื่อหาทิศทางใหม่ที่ไม่สัมพันธ์กัน (Principal Components)

🔹 4. การเลือก Components

เลือกเฉพาะ components ที่รวมกันแล้วอธิบายได้ ≥ 95% ของความแปรปรวน

ผลลัพธ์คือ 35 components ที่ใช้แทน indicators เดิม

✅ ข้อดีของ PCA ในบริบทนี้

ลดความซ้ำซ้อนของข้อมูล

ปรับปรุงความเสถียรของโมเดล ML/DL

ลดเวลาในการฝึกโมเดล

ป้องกันการ overfitting จาก indicators ที่มีความสัมพันธ์กันสูง

⚠️ ข้อจำกัดของ PCA

อาจ สูญเสีย granularity จาก indicators ที่มี variance ต่ำแต่มีความสำคัญเชิงกลยุทธ์ เช่น:

Chaikin Oscillator
PCA ไม่สามารถรักษาความหมายดั้งเดิมของ indicators ได้ → ยากต่อการตีความเชิงธุรกิจ
🛠️ เครื่องมือที่ใช้:

🎯 สรุปภาพรวม
ขั้นตอน รายละเอียด ผลกระทบ
Linear Interpolation เติมค่าที่ขาดหายไป รักษาความต่อเนื่องของข้อมูล
Standardization ปรับ scale ของ indicators ป้องกัน bias ใน PCA
PCA ลดมิติข้อมูล เพิ่มประสิทธิภาพโมเดล, ลด multicollinearity
ข้อจำกัด สูญเสีย indicators ที่ variance ต่ำ อาจพลาดข้อมูลเชิงกลยุทธ์
🔷 3.8.1. Random Forest Regression
✅ หลักการของ Random Forest
เป็น ensemble learning method ที่รวมผลจากหลาย decision trees
ใช้เทคนิค bagging: สุ่มตัวอย่างจาก training set แบบมีการแทนที่ (bootstrap sampling)
แต่ละต้นไม้เรียนรู้จากชุดข้อมูลที่แตกต่างกัน → ลดความลำเอียงและความแปรปรวน
✅ การประเมินผลของแต่ละต้นไม้
ใช้ Out-Of-Bag (OOB) samples ซึ่งเป็นข้อมูลที่ไม่ได้ถูกใช้ในการสร้างต้นไม้แต่ละต้น
วัดความแม่นยำด้วย Mean Squared Error (MSE) จาก OOB → ให้การประเมินที่ไม่ลำเอียง
✅ สูตรการพยากรณ์โดยรวมของ Random Forest
สมมติว่าเรามีต้นไม้
T
1
,
T
2
,
.
.
.
,
T
K
T
1

,T
2

,...,T
K

และ predictor vector
x
x:

$$\hat{y} = \frac{1}{K} \sum_{k=1}^{K} T_k(x)$$

$$\hat{y} = \sum_{k=1}^{K} \frac{1}{K} T_k$$

$$\frac{1}{K} T_k(x)$$

โดยที่ $T_k(x)$ คือค่าที่ต้นไม้ที่ k พยากรณ์จาก input

x
x
การเฉลี่ยผลลัพธ์จากทุกต้นไม้ช่วยลดความแปรปรวนและเพิ่มความแม่นยำ
✅ การตั้งค่าของโมเดลในงานวิจัยนี้
จำนวนต้นไม้: 500 trees
จำนวนตัวแปรที่เลือกต่อการ split:
m
m variables (ค่าเฉพาะไม่ได้ระบุ แต่มักใช้
p
p

หรือ
log

2
(
p
)
log
2

(p) โดย
p
p คือจำนวน features)
ใช้ OOB error เพื่อปรับแต่ง hyperparameters
✅ ผลลัพธ์ของโมเดล
วัดผลด้วย Mean Absolute Error (MAE):
ได้ค่า MAE = 1.39 (เมื่อไม่ใช้ lagged variables)
แสดงถึงความแม่นยำที่ดีในการพยากรณ์ราคาปิดของ S&P 500
✅ ข้อดีของ Random Forest ในบริบท Time Series
Robust ต่อ noise และ outliers
สามารถจับ interactions ที่ซับซ้อน ระหว่าง indicators
ไม่ต้องปรับ hyperparameters มากเท่าโมเดลอื่น เช่น XGBoost หรือ LSTM
ให้ผลลัพธ์ที่ ตีความได้ง่าย ผ่าน feature importance
🎯 สรุปภาพรวม
หัวข้อ รายละเอียด
ประเภทโมเดล Ensemble Regression (Bagging)
จำนวนต้นไม้ 500
เทคนิคการประเมิน Out-of-Bag (OOB) Error
Metric ที่ใช้ MAE = 1.39
จุดเด่น Robust, ตีความง่าย, ลด variance
จุดประสงค์ ใช้เป็น baseline ที่มีความแม่นยำสูงและไม่ overfit

3.8.2. XGBoost Model

✅ หลักการของ XGBoost
XGBoost (Extreme Gradient Boosting) เป็นโมเดลแบบ gradient boosting ที่มีประสิทธิภาพสูง
ใช้การรวมผลจากหลายต้นไม้ (CART: Classification and Regression Trees)
มีความสามารถในการ parallelization และ decentralization → ทำให้เรียนรู้เร็วและแม่นยำกว่ารุ่นก่อนหน้า (Han, Kim, & Enke, 2023)
✅ โครงสร้างของ CART ใน XGBoost
CART ใช้ Gini Index แทน Entropy ในการเลือกตัวแปร
ทำการ binary split โดยแบ่งข้อมูลออกเป็น 2 กลุ่มตามตัวแปรที่เลือก
แต่ละต้นไม้จะเรียนรู้จาก subset ของข้อมูล → สร้าง leaf nodes ที่มีค่าพยากรณ์เฉพาะ
✅ สูตรการพยากรณ์ของ XGBoost
สมมติว่าเรามีต้นไม้
T
1
$,$
T
2
$,$
$.$
$.$
$.$
$,$
T
K
T
1

$,T$
2

$,...,T$
K

และ predictor vector
x
$x:$

y
$\hat{}$
$=$
$\sum$
k
$=$
1
K

f
k
(
x
)
y
^

=
k=1
∑
K

f
k

(x)
โดยที่
f
k
f
k

คือฟังก์ชันของต้นไม้ที่
k
k และ
y
^
y
^

คือค่าพยากรณ์รวม
✅ Objective Function ของ XGBoost
L
(
ϕ
)
=
∑
i
=
1
n
l

(
y
i
,
y
^
i
)
+
∑
k
=
1
K
Ω
(
f
k
)
L(ϕ)=
i=1
∑
n

l(y
i

,
y
^

i

)+
k=1
∑
K

Ω(f
k

)
l

(
y
i
,
y
^
i
)
l(y
i

,
y
^

i

): loss function เช่น MSE หรือ MAE
Ω
(
f
k
)
Ω(f
k

): regularization term → ป้องกัน overfitting
Regularization term ประกอบด้วย:
จำนวน leaf nodes (
T
T)
ค่า norm ของ weights (
//
w
//
2
// w //
2
)
ค่าคงที่ (
γ
γ) ที่ควบคุมความซับซ้อน
✅ การตั้งค่าของโมเดลในงานวิจัยนี้

Estimators: 200 ต้นไม้
Max Depth: 6 → ควบคุมความซับซ้อน
Learning Rate: 0.01 → ปรับการเรียนรู้ให้ช้าแต่เสถียร
Regularization Term: ใช้ค่า
γ
γ เพื่อควบคุมความซับซ้อนของต้นไม้
✅ ข้อดีของ XGBoost ในบริบท Time Series
จัดการกับ non-linear relationships ได้ดี
มีความสามารถในการ generalize ไปยังข้อมูลใหม่
ปรับแต่งได้ละเอียดผ่าน hyperparameters และ regularization
เหมาะกับข้อมูลที่มี noise และความแปรปรวนสูง เช่น ตลาดหุ้น
🎯 สรุปภาพรวม
หัวข้อ รายละเอียด
ประเภทโมเดล Gradient Boosting (CART-based)
จำนวนต้นไม้ 200
ความลึกสูงสุด 6
Learning Rate 0.01
Regularization ใช้
γ
γ เพื่อควบคุมความซับซ้อน
จุดเด่น แม่นยำสูง, ปรับแต่งได้, ทน noise
จุดประสงค์ พยากรณ์แนวโน้มราคาหุ้น S&P 500

3.8.3. Support Vector Regression (SVR)
✅ หลักการของ SVR
SVR สร้างฟังก์ชันการถดถอยในรูปแบบ:
f
(
x
)
=
w
T
x
+
b
f(x)=w
T
x+b
โดยที่:

w
w: เวกเตอร์น้ำหนัก
b

b: ค่า bias
x
x: เวกเตอร์ของ input features
จุดมุ่งหมายคือการ ลดความคลาดเคลื่อน ระหว่างค่าที่พยากรณ์กับค่าจริง โดยอนุญาตให้มี margin of tolerance (ϵ-insensitive zone)

✅ Optimization Objective
ฟังก์ชันเป้าหมาย:
min

1
2
//
w
//
2
+
C
∑
i
=
1
n
(
ξ
i
+
ξ
i
*
)
min
2
1

// w //
2
+C
i=1
∑
n

(ξ
i

+ξ
i
*

)
โดยมี constraints:

y
i
−
f
(
x
i
)
≤
ϵ
+
ξ
i
y
i

−f(x
i

)≤ϵ+ξ
i

f
(
x
i
)
−
y
i
≤
ϵ
+
ξ
i

*
f(x
i

)−y
i

≤ϵ+ξ
i
*

ξ
i
,
ξ
i
*
≥
0
ξ
i

,ξ
i
*

≥0
C: ค่าที่ควบคุม trade-off ระหว่างความแม่นยำกับความซับซ้อนของโมเดล

ϵ: ขอบเขตความคลาดเคลื่อนที่ยอมรับได้

✅ การจัดการ Non-linearity ด้วย Kernel
ใช้ Radial Basis Function (RBF) kernel:
K
(
x
i
,
x
j
)
=
exp

(
−
γ
//
x
i
−
x
j
//
2
)
K(x
i

,x
j

)=exp(−γ // x
i

−x
j

//
2
)
ช่วยให้ SVR สามารถเรียนรู้ความสัมพันธ์ที่ไม่เป็นเชิงเส้นได้
✅ การตั้งค่าในงานวิจัยนี้
ใช้ RBF kernel โดยตั้งค่า:
γ
γ: ควบคุมความไวของ kernel ต่อข้อมูลแต่ละจุด
C
C: ปรับสมดุลระหว่างความแม่นยำและความเรียบง่ายของโมเดล
ใช้ Grid Search เพื่อหาค่า hyperparameters ที่ดีที่สุด
✅ ข้อดีของ SVR
มีความสามารถในการจัดการกับข้อมูลที่มี noise
เหมาะกับชุดข้อมูลที่มีขนาดไม่ใหญ่มาก
มีความยืดหยุ่นสูงเมื่อใช้ร่วมกับ kernel
🔷 3.8.4. Long Short-Term Memory (LSTM)
✅ พื้นฐานของ LSTM
LSTM เป็นโมเดลในตระกูล Recurrent Neural Networks (RNNs) ที่ออกแบบมาเพื่อจัดการกับ ลำดับข้อมูล (sequential data) เช่น Time Series

มีความสามารถในการ จดจำข้อมูลในอดีตระยะยาว ผ่านโครงสร้างที่เรียกว่า memory cell และ gates
✅ องค์ประกอบของ LSTM Node
Forget Gate: ตัดสินใจว่าจะลืมข้อมูลใดจากสถานะก่อนหน้า
Input Gate: ตัดสินใจว่าจะเพิ่มข้อมูลใหม่ใดเข้าไปใน memory
Output Gate: ตัดสินใจว่าจะส่งข้อมูลใดออกไปเป็น output
(ในเอกสารของคุณมีการอ้างถึง Fig. 2 ซึ่งน่าจะเป็นภาพแสดงโครงสร้าง LSTM node)

✅ ข้อดีของ LSTM สำหรับ Time Series
จัดการกับ temporal dependencies ได้ดี
เหมาะกับข้อมูลที่มี seasonality, momentum, หรือ volatility clustering
ใช้ได้ทั้งแบบ univariate และ multivariate time series
✅ การตั้งค่าในงานวิจัยนี้ (จากข้อมูลก่อนหน้า):
2 layers × 64 units
Dropout = 0.2
Epochs = 100
ใช้ validation set เพื่อตรวจสอบการ overfitting
🎯 สรุปเปรียบเทียบ SVR vs LSTM
คุณสมบัติ SVR LSTM
ประเภทโมเดล Kernel-based Regression Deep Learning (RNN)
ความสามารถเชิงเวลา จำกัด สูงมาก
ความซับซ้อน ต่ำ-ปานกลาง สูง
ความสามารถในการจับ non-linearity สูง (ผ่าน kernel) สูง (ผ่านโครงสร้างลำดับ)
ความเหมาะสมกับ Time Series ปานกลาง สูงมาก
การปรับแต่ง Grid Search ต้องใช้การปรับหลาย hyperparameters
โครงสร้างของ LSTM Node และการทำงานของแต่ละ Gate
🧠 ภาพรวมของ LSTM Node
แต่ละ Node ประกอบด้วยชุดของ cells ที่ทำหน้าที่เก็บข้อมูลจากลำธารข้อมูล (data stream) ที่ผ่านเข้ามา
เส้นบนของ cell ทำหน้าที่เป็น สายพานข้อมูล (transport line) ที่ส่งข้อมูลจากอดีตไปยังปัจจุบัน
ความเป็นอิสระของแต่ละ cell ช่วยให้โมเดลสามารถเลือกกรองหรือเพิ่มค่าจาก cell หนึ่งไปยังอีก cell ได้อย่างยืดหยุ่น
Sigmoid Layer ทำหน้าที่เป็น gate ที่ควบคุมการไหลของข้อมูล โดยให้ค่า 0 หรือ 1 เพื่อ “ปิด” หรือ “เปิด” การส่งผ่านข้อมูล
🔑 การทำงานของแต่ละ Gate
1. 🩹 Memory Gate
เลือกว่าจะเก็บข้อมูลใหม่ใดไว้ใน cell
ประกอบด้วยสองขั้นตอน:
Input Gate (Sigmoid Layer): เลือกว่าค่าใดจะถูกเปลี่ยนแปลง
Candidate Layer (tanh หรือ Linear): สร้างเวกเตอร์ของค่าผู้สมัครใหม่ที่จะถูกเพิ่มเข้าไปใน cell state
2. 🧹 Forget Gate
ตัดสินใจว่าจะ “ลืม” ข้อมูลเก่าใน cell state หรือไม่
ค่า output อยู่ระหว่าง 0 ถึง 1:
1 = เก็บข้อมูลไว้ทั้งหมด
0 = ลบข้อมูลทิ้งทั้งหมด
3. 📤 Output Gate

กำหนดว่า cell จะส่งค่าอะไรออกมา
พิจารณาจาก:
cell state ปัจจุบัน
ข้อมูลใหม่ที่ถูกกรองแล้ว
🧮 สูตรทั่วไปของแต่ละ Gate
(หมายเลขในวงเล็บ เช่น (5), (6), (7), (8) หมายถึงสูตรในงานวิจัยต้นฉบับ)

Gate แต่ละตัวมี weight (W) และ bias (b) ของตัวเอง
ใช้ sigmoid หรือ tanh เพื่อควบคุมการเปิด-ปิดของ gate
Output ของแต่ละ gate จะถูกใช้ในการอัปเดต hidden state (h□) และ cell state (C□)
🧪 การใช้งานในงานวิจัยนี้
ใช้ LSTM 2 ชั้น (2-layer architecture)
แต่ละชั้นมี 64 units
ใช้ dropout rate 0.2 เพื่อลด overfitting
ฝึกโมเดล 100 epochs โดยใช้ rolling window 60 วัน เพื่อจำลองสภาพการเทรดจริง
ใช้ Bayesian Optimization 50 รอบ เพื่อปรับพารามิเตอร์ให้เหมาะสม
อ้างอิงแนวทางจาก Goodfellow et al., 2016 เพื่อให้การประเมินโมเดลมีความน่าเชื่อถือ
📌 ตีความเชิงลึก
การใช้ sigmoid gate ทำให้ LSTM มีความสามารถในการ “จำ” หรือ “ลืม” ข้อมูลอย่างมีเงื่อนไข
การใช้ Bayesian Optimization ช่วยให้การปรับพารามิเตอร์มีประสิทธิภาพมากกว่าการสุ่มหรือ grid search
การใช้ rolling window สะท้อนการเรียนรู้แบบ time-aware ซึ่งเหมาะกับข้อมูลหุ้นที่มีลักษณะเปลี่ยนแปลงตามเวลา

🔍 โครงสร้างของ LSTM Node และการทำงานของแต่ละ Gate
🧠 ภาพรวมของ LSTM Node
แต่ละ Node ประกอบด้วยชุดของ cells ที่ทำหน้าที่เก็บข้อมูลจากลำธารข้อมูล (data stream) ที่ผ่านเข้ามา
เส้นบนของ cell ทำหน้าที่เป็น สายพานข้อมูล (transport line) ที่ส่งข้อมูลจากอดีตไปยังปัจจุบัน
ความเป็นอิสระของแต่ละ cell ช่วยให้โมเดลสามารถเลือกกรองหรือเพิ่มค่าจาก cell หนึ่งไปยังอีก cell ได้อย่างยืดหยุ่น
Sigmoid Layer ทำหน้าที่เป็น gate ที่ควบคุมการไหลของข้อมูล โดยให้ค่า 0 หรือ 1 เพื่อ “ปิด” หรือ “เปิด” การส่งผ่านข้อมูล
🔑 การทำงานของแต่ละ Gate
1. 🩹 Memory Gate
เลือกว่าจะเก็บข้อมูลใหม่ใดไว้ใน cell
ประกอบด้วยสองขั้นตอน:
Input Gate (Sigmoid Layer): เลือกว่าค่าใดจะถูกเปลี่ยนแปลง
Candidate Layer (tanh หรือ Linear): สร้างเวกเตอร์ของค่าผู้สมัครใหม่ที่จะถูกเพิ่มเข้าไปใน cell state
2. 🧹 Forget Gate
ตัดสินใจว่าจะ “ลืม” ข้อมูลเก่าใน cell state หรือไม่
ค่า output อยู่ระหว่าง 0 ถึง 1:
1 = เก็บข้อมูลไว้ทั้งหมด
0 = ลบข้อมูลทิ้งทั้งหมด
3. 📤 Output Gate

กำหนดว่า cell จะส่งค่าอะไรออกมา
พิจารณาจาก:
cell state ปัจจุบัน
ข้อมูลใหม่ที่ถูกกรองแล้ว
🧮 สูตรทั่วไปของแต่ละ Gate
(หมายเลขในวงเล็บ เช่น (5), (6), (7), (8) หมายถึงสูตรในงานวิจัยต้นฉบับ)

Gate แต่ละตัวมี weight (W) และ bias (b) ของตัวเอง
ใช้ sigmoid หรือ tanh เพื่อควบคุมการเปิด-ปิดของ gate
Output ของแต่ละ gate จะถูกใช้ในการอัปเดต hidden state (h□) และ cell state (C□)
🧪 การใช้งานในงานวิจัยนี้
ใช้ LSTM 2 ชั้น (2-layer architecture)
แต่ละชั้นมี 64 units
ใช้ dropout rate 0.2 เพื่อลด overfitting
ฝึกโมเดล 100 epochs โดยใช้ rolling window 60 วัน เพื่อจำลองสภาพการเทรดจริง
ใช้ Bayesian Optimization 50 รอบ เพื่อปรับพารามิเตอร์ให้เหมาะสม
อ้างอิงแนวทางจาก Goodfellow et al., 2016 เพื่อให้การประเมินโมเดลมีความน่าเชื่อถือ
📌 ตีความเชิงลึก
การใช้ sigmoid gate ทำให้ LSTM มีความสามารถในการ “จำ” หรือ “ลืม” ข้อมูลอย่างมีเงื่อนไข
การใช้ Bayesian Optimization ช่วยให้การปรับพารามิเตอร์มีประสิทธิภาพมากกว่าการสุ่มหรือ grid search
การใช้ rolling window สะท้อนการเรียนรู้แบบ time-aware ซึ่งเหมาะกับข้อมูลหุ้นที่มีลักษณะเปลี่ยนแปลงตามเวลา
📊 3.9 Evaluation Measures: การประเมินประสิทธิภาพโมเดลพยากรณ์หุ้น
🎯 วัตถุประสงค์
การประเมินผลมีความสำคัญอย่างยิ่งในการ:

ตรวจสอบความแม่นยำของโมเดล
เปรียบเทียบผลพยากรณ์กับค่าจริง
สะท้อนสภาพการเทรดจริงผ่านการใช้ walk-forward validation
🧮 ประเภทของ Metrics ที่ใช้
✅ สำหรับ Classification Models
ใช้ accuracy-based metrics เช่น Accuracy, Precision, Recall, F1-score
📉 สำหรับ Regression Models (ใช้ในงานวิจัยนี้)
ใช้ error-based metrics ที่เน้นการวัดความคลาดเคลื่อนระหว่างค่าพยากรณ์กับค่าจริง:

1. MAE (Mean Absolute Error)
คำนวณค่าเฉลี่ยของความคลาดเคลื่อนแบบสัมบูรณ์
จุดเด่น: ทนต่อ outliers ได้ดี
สูตร:
M
A
E
=
1

n
∑
i
=
1
n
|
y
i
−
y
^
i
|
MAE=
n
1

∑
i=1
n

|y
i

−
y
^

i

|
2. RMSE (Root Mean Square Error)
คำนวณรากที่สองของค่าเฉลี่ยกำลังสองของความคลาดเคลื่อน
จุดเด่น: เน้นการลงโทษความผิดพลาดที่มีขนาดใหญ่
สูตร:
R
M
S
E
=
1
n

∑
i
=
1
n
(
y
i
−
y
^
i
)
2
RMSE=
n
1

∑
i=1
n

(y
i

−
y
^

i

)
2

3. MAPE (Mean Absolute Percentage Error)
วัดความคลาดเคลื่อนในรูปแบบเปอร์เซ็นต์
จุดเด่น: ให้การตีความเชิงสัมพัทธ์
ข้อควรระวัง: ไม่แม่นยำเมื่อค่าจริงเข้าใกล้ศูนย์
สูตร:
M
A

P
E
=
100
n
∑
i
=
1
n
|
y
i
−
y
^
i
y
i
|
MAPE=
n
100

∑
i=1
n

y
i

y
i

−
y
^

i

🧪 การทดลองในงานวิจัย
ใช้ชุดข้อมูลทดสอบ 15% จากช่วงปี 2012–2023
ใช้ walk-forward validation เพื่อจำลองการเทรดจริง
อ้างอิงจากงานของ Obthong et al., 2020 และ Kumbure et al., 2022 เพื่อเลือก metric ที่เหมาะสมกับการพยากรณ์ทางการเงิน
📌 ตีความเชิงลึก
การใช้ หลาย metrics ช่วยให้เห็นภาพรวมของโมเดลทั้งในแง่ความแม่นยำและความเสถียร
RMSE เหมาะกับการวิเคราะห์ช่วงที่มีความผันผวนสูง (เช่นช่วง holiday effect)
MAPE เหมาะกับการเปรียบเทียบข้ามหุ้นหรือช่วงเวลา แต่ต้องระวังเมื่อราคาต่ำมาก
การใช้ walk-forward validation สะท้อนการใช้งานจริงมากกว่าการแบ่ง train/test แบบสุ่ม
🧾 4. Results and Discussion
📈 ภาพรวมของข้อมูล (Fig. 3)
กราฟแสดง ราคาปิดของดัชนี S&P 500 ตั้งแต่ 01/01/1950 ถึง 12/07/2023
เส้นสีน้ำเงิน = ราคาปิดจริง
จุดประสงค์: เพื่อวิเคราะห์และหา ตัวชี้วัดสำคัญ ที่สามารถพยากรณ์ราคาปิดในอนาคตได้
แม้จะมีความผันผวน แต่แนวโน้มโดยรวมของราคาคือ ขาขึ้น
🧪 ขั้นตอนการเตรียมข้อมูล (อ้างอิงจาก Section 3)
สร้าง technical indicators
ลบค่า missing/null
ทำ Normalization
ใช้ PCA เพื่อลดฟีเจอร์ซ้ำซ้อน
ทดลองโมเดล:
Random Forest Regression
XGBoost
Support Vector Regression (SVR)
Long Short-Term Memory (LSTM)
📊 4.1 Model Evaluation
📋 ตารางเปรียบเทียบผลลัพธ์ (Table 3)

Model	MAE	MAPE	RMSE	
SVR (no lag)	7.76	2.04	10.48	
SVR (with lag)	8.06	2.08	10.92	
LSTM (no lag)	0.014	0.008	0.045	
LSTM (with lag)		0.051	0.085	0.19
RF (no lag)	1.39	0.017	4.47	
RF (with lag)	2.08	0.010	5.21	
XGBoost (no lag)		4.89	0.084	9.24
XGBoost (with lag)		18.92	0.894	21.44

🔍 วิเคราะห์ผลลัพธ์

✅ LSTM
ดีที่สุดในทุก metric โดยเฉพาะ ไม่มี lag
MAE ต่ำมาก (0.014) → พยากรณ์แม่นยำ
MAPE ต่ำสุด (0.008) → ความคลาดเคลื่อนสัมพัทธ์ต่ำ
RMSE ต่ำสุด (0.045) → ความผิดพลาดขนาดใหญ่แทบไม่มี
🟨 Random Forest
ดีกว่า SVR และ XGBoost
มีความแม่นยำพอสมควร โดยเฉพาะ MAPE ต่ำมาก
แต่ยังห่างจาก LSTM
🟥 SVR
MAE และ RMSE สูง → พยากรณ์คลาดเคลื่อนมาก
ผลลัพธ์แทบไม่ต่างกันระหว่างมี lag กับไม่มี lag
❌ XGBoost
แย่ที่สุด โดยเฉพาะเมื่อใช้ lag
MAE สูงถึง 18.92 และ MAPE เกือบ 1 (89.4%) → พยากรณ์ผิดพลาดรุนแรง
📌 ข้อสรุป
LSTM เหมาะสมที่สุด สำหรับการพยากรณ์ราคาปิดของ S&P 500
การใช้ lag ไม่ได้ช่วยเสมอไป และอาจทำให้โมเดลบางตัวแย่ลง (เช่น XGBoost)
การประเมินด้วย MAE, MAPE, RMSE ช่วยให้เห็นทั้งความแม่นยำและความเสถียรของโมเดล
📊 4.1.1 การเปรียบเทียบผลลัพธ์กับงานวิจัยก่อนหน้า
🎯 วัตถุประสงค์
เปรียบเทียบประสิทธิภาพของโมเดลพยากรณ์ราคาปิด S&P 500 กับงานวิจัย 3 ฉบับ:

Gao et al. (2017)
Sarıkoç and Celik (2024)
Hossain et al. (2018)
โดยใช้ 3 metric หลัก:

MAE (Mean Absolute Error)
RMSE (Root Mean Square Error)
MAPE (Mean Absolute Percentage Error)
📋 ตารางเปรียบเทียบผลลัพธ์ (Table 4)

Study / Model	MAE	MAPE	RMSE
Current Study (LSTM no lag)	0.014	0.008	0.045
Hossain et al. (LSTM+GRU)	0.023	0.0413	0.0023
Hossain et al. (PCA-ICA-LSTM)		0.0147	0.0205 0.00038
Sarıkoç & Celik (PCA-ICA-LSTM)		0.077	0.042 0.084
Gao et al. (LSTM)	14.7709		0.7240 20.4668

🔍 วิเคราะห์เชิงเปรียบเทียบ
✅ Current Study (LSTM without lag)
MAE และ MAPE ต่ำที่สุด → พยากรณ์แม่นยำมาก
RMSE สูงกว่าบาง hybrid model → อาจมีความผิดพลาดขนาดใหญ่เล็กน้อย
🧠 Hossain et al.
LSTM+GRU: แม้ MAE และ MAPE สูงกว่าเล็กน้อย แต่ RMSE ต่ำมาก → โมเดลมีความเสถียรสูง

PCA-ICA-LSTM: สมดุลดีมาก → RMSE ต่ำสุดในทุกงานวิจัย
🧪 Sarıkoç & Celik
ใช้ PCA-ICA-LSTM เช่นกัน แต่ผลลัพธ์ด้อยกว่าของ Hossain → อาจเกิดจากการเลือกฟีเจอร์หรือการตั้งค่าพารามิเตอร์
❌ Gao et al.
ใช้ LSTM แบบดั้งเดิม → ผลลัพธ์แย่ที่สุด
MAE สูงถึง 14.77 และ RMSE สูงถึง 20.46 → พยากรณ์ผิดพลาดรุนแรง
📌 ข้อสรุป
งานวิจัยปัจจุบันใช้ LSTM แบบปรับแต่ง ได้ผลลัพธ์ที่แม่นยำมาก โดยเฉพาะในแง่ MAE และ MAPE
Hybrid models เช่น LSTM+GRU และ PCA-ICA-LSTM มีแนวโน้มให้ RMSE ต่ำกว่า → เหมาะกับการพยากรณ์ในตลาดที่มีความผันผวน
การใช้ feature selection และ การรวมโมเดล ช่วยเพิ่มความแม่นยำอย่างมีนัยสำคัญ
งานของ Gao et al. แสดงให้เห็นว่า LSTM แบบดั้งเดิมอาจไม่เพียงพอในบริบทของตลาดหุ้น

4.2 Feature Importance: การวิเคราะห์ตัวชี้วัดทางเทคนิค
🎯 วัตถุประสงค์
เพื่อระบุว่า ตัวชี้วัดทางเทคนิค (technical indicators) ใดมีความสำคัญที่สุดในการพยากรณ์ดัชนี S&P 500
ฟีเจอร์พื้นฐาน เช่น close, high, open, low ถูกใช้ในทุกการทดลอง แต่ ไม่ถูกจัดอยู่ในกลุ่ม indicators ที่นำมาวิเคราะห์
📊 ผลลัพธ์จาก SVR (Support Vector Regression)
🔹 SVR (ไม่มี lag)
ตัวชี้วัดสำคัญที่สุด: MFI (Money Flow Index)
รองลงมา: TEMA, Volume, FWMA, OBV
🔸 SVR (มี lag)
ตัวชี้วัดสำคัญที่สุด: HLC3 (High+Low+Close)/3
รองลงมา: TEMA, Volume, FWMA, OBV
📌 หมายเหตุ:

MFI และ HLC3 อธิบายความแปรปรวนของโมเดล SVR ได้มากกว่า 80%
แสดงถึงความสำคัญของ indicator ที่สะท้อน แรงซื้อขาย และ ราคากลาง
📈 ผลลัพธ์จาก XGBoost
🔹 XGBoost (ไม่มี lag)
ตัวชี้วัดสำคัญที่สุด: Ichimoku
รองลงมา: OBV, TEMA, SMA14, HLC3
🔸 XGBoost (มี lag)
ตัวชี้วัดสำคัญที่สุด: HWMA (Hull Weighted Moving Average)
รองลงมา: OBV, TEMA, SMA14, HLC3
📌 ข้อสังเกต:

ตัวชี้วัดอย่าง OBV, TEMA, และ HLC3 ปรากฏในทั้ง SVR และ XGBoost → สะท้อนว่าเป็น indicators ที่มีความสำคัญสม่ำเสมอ
Ichimoku และ HWMA มีบทบาทเด่นใน XGBoost → อาจสะท้อนความสามารถของโมเดลในการจับ pattern ที่ซับซ้อน

📌 ข้อสรุปเชิงวิเคราะห์
ตัวชี้วัดที่มีความสำคัญสูงสุดแตกต่างกันตามโมเดลและการใช้ lag
Indicators ที่เกี่ยวข้องกับ momentum (MFI, OBV) และ moving average (TEMA, HWMA) มีบทบาทสำคัญ
การใช้ lag อาจเปลี่ยนลำดับความสำคัญของฟีเจอร์ → ควรพิจารณาอย่างรอบคอบในการออกแบบโมเดล

ผลลัพธ์จาก Random Forest (RF)
🔹 RF (ไม่มี lag)
ตัวชี้วัดสำคัญที่สุด: OBV (On-Balance Volume)
🔸 RF (มี lag)
ตัวชี้วัดสำคัญที่สุด: HLC3 (High + Low + Close) / 3
🔍 Indicators รองลงมา
SWMA (Smoothed Weighted Moving Average)
PVT (Price Volume Trend)
BBL (Bollinger Band Lower)
TEMA (Triple Exponential Moving Average)
🔁 การปรากฏซ้ำของ Indicators สำคัญ
ตัวชี้วัดที่ปรากฏในหลายโมเดล ได้แก่:

Indicator	SVR	XGBoost	RF
OBV ✅	✅	✅	
HLC3 ✅	✅	✅	
TEMA ✅	✅	✅	
FWMA ✅	✅	❌	
Ichimoku	❌	✅	✅
PVT ❌	❌	✅	

📌 ข้อสังเกต:

OBV, HLC3, TEMA เป็น indicators ที่มีความสำคัญสูงในทุกโมเดล → ควรพิจารณาเป็นฟีเจอร์หลักในการพยากรณ์
Ichimoku และ FWMA มีบทบาทเด่นใน XGBoost และ RF → อาจสะท้อนความสามารถในการจับ pattern ระยะกลางถึงยาว
PVT และ BBL ปรากฏเฉพาะใน RF → อาจเหมาะกับโมเดล tree-based ที่จับความสัมพันธ์แบบไม่เชิงเส้น
📌 ข้อสรุปเชิงวิเคราะห์
การวิเคราะห์ feature importance ช่วยให้เข้าใจว่าโมเดลแต่ละแบบ “มองเห็น” ความสำคัญของ indicators อย่างไร
การเลือก indicators ที่ปรากฏซ้ำในหลายโมเดล (เช่น OBV, HLC3, TEMA) อาจช่วยเพิ่มความแม่นยำและความเสถียรของโมเดล
การใช้ lag มีผลต่อการเปลี่ยนลำดับความสำคัญของ indicators → ควรทดสอบทั้งแบบมีและไม่มี lag

ผลลัพธ์จาก LSTM (Fig. 7)
🔹 LSTM (ไม่มี lag และมี lag)
ตัวชี้วัดสำคัญที่สุด: HLC3
รองลงมา: OBV, SMA5, TEMA, BBL, SMA50

📌 ข้อสังเกต:

HLC3, OBV, TEMA, BBL ปรากฏในหลายโมเดล → เป็น indicators ที่มีความสำคัญสม่ำเสมอ
LSTM ให้ความสำคัญกับ indicators ที่สะท้อน แนวโน้มราคาและปริมาณการซื้อขาย
📋 Table 5: สรุปการเลือก Indicators ในแต่ละโมเดล
ตารางนี้แสดงว่า indicators ใดถูกเลือกในแต่ละโมเดล (มีและไม่มี lag):

Indicator	SVR	XGBoost	RF	LSTM		
HLC3 ✅	✅	✅	✅			
OBV ✅	✅	✅	✅			
TEMA ✅	✅	✅	✅			
FWMA ✅	✅	✅	✅			
BBL / BB	✅	✅	✅	✅		
Ichimoku	❌	✅	✅	✅		
PVT ❌	✅	✅	✅			
SMA ✅	✅	✅	✅			
SWMA / HWMA / HMA		บางส่วน		✅	✅	✅

📌 Indicators ที่โดดเด่นที่สุด:

HLC3: ปรากฏในทุกโมเดล → เป็นตัวแทนราคากลางที่มีความเสถียร
OBV: สะท้อนแรงซื้อขาย → มีบทบาทสำคัญในทุกโมเดล
TEMA, FWMA, BB: เป็น moving average และ volatility indicators ที่ช่วยจับแนวโน้มและความผันผวน
📌 ข้อสรุปเชิงวิเคราะห์
Indicators ที่ปรากฏในหลายโมเดลมีแนวโน้มให้ผลลัพธ์ที่แม่นยำและเสถียร
การใช้ indicators ที่หลากหลาย เช่น momentum, trend, volatility, volume ช่วยให้โมเดลเข้าใจพฤติกรรมตลาดได้ดีขึ้น
การใช้ lag มีผลต่อการเปลี่ยนลำดับความสำคัญของ indicators → ควรพิจารณาในการออกแบบฟีเจอร์

Momentum Indicators
RVI, KAMA, SWMA, HLC3
→ ใช้ในการวัดความเร็วของการเปลี่ยนแปลงราคา และช่วยระบุภาวะ overbought/oversold
🔸 Trend Indicators
MI, EMA, TEMA, FWMA, HMA, SWMA, HWMA, Ichimoku
→ ใช้ในการวิเคราะห์แนวโน้มและลดสัญญาณรบกวน
🔺 Volatility Indicators
ATR, Price Distance, BB (Bollinger Bands)
→ วัดการกระจายตัวของราคา ช่วยประเมินความเสี่ยงและการเกิด breakout
🔻 Volume Indicators
MFI, OBV, PVT
→ สะท้อนแรงซื้อขายและแนวโน้มการสะสม/กระจายของตลาด
การตีความเชิงลึกของโมเดลแต่ละประเภท
🔷 LSTM Models
เน้นการเรียนรู้จาก ลำดับเวลา (temporal dependencies)
ให้ความสำคัญกับ indicators ที่สะท้อนการเปลี่ยนแปลงรายวัน เช่น:

HLC3: ราคากลางของวัน
SMA5: ค่าเฉลี่ยเคลื่อนที่ 5 วัน
เหมาะกับการจับ pattern ที่เปลี่ยนแปลงตามเวลา เช่น holiday effect หรือ volatility หลังวันหยุด
🟩 Tree-Based Models (XGBoost, Random Forest)
โครงสร้างแบบ non-parametric ช่วยจับความสัมพันธ์แบบไม่เชิงเส้น
เน้น indicators ที่เกี่ยวกับ ปริมาณการซื้อขาย เช่น:
OBV: สะท้อนแรงสะสม/กระจายของตลาด
การตัดสินใจแบบลำดับชั้นช่วยให้เข้าใจสัญญาณจาก volume ได้ดี
🟦 Support Vector Regression (SVR)
ใช้ kernel-based optimization → ต้องการฟีเจอร์ที่มี noise ต่ำ
เน้น indicators ที่เป็น ค่าเฉลี่ยแนวโน้มแบบเรียบ เช่น:
TEMA, EMA23
เหมาะกับการสร้าง hyperplane ใน space ที่มีมิติสูง
📌 ข้อสังเกต:

ความแตกต่างในการเลือก indicators สะท้อนถึง inductive bias ของแต่ละโมเดล
การเลือก indicators ให้สอดคล้องกับจุดแข็งของโมเดล → ช่วยเพิ่มความแม่นยำและความเสถียร
⚠️ ข้อจำกัดของงานวิจัย
🗂️ Data Limitations
แม้ข้อมูลครอบคลุม 70 ปี แต่ตลาดมีการเปลี่ยนแปลงโครงสร้าง เช่น:
การเกิด algorithmic trading
เหตุการณ์สำคัญ เช่น COVID-19
Indicators จากอดีตอาจไม่สะท้อนสภาพตลาดปัจจุบัน
ยังไม่ได้ทดสอบกับดัชนีอื่น เช่น Dow Jones, Nasdaq, หรือ ตลาดต่างประเทศ
🌐 Market Dynamics
ตลาดหุ้นได้รับอิทธิพลจากหลายปัจจัย เช่น:
เศรษฐกิจ, การเมือง, ความรู้สึกของนักลงทุน
โมเดลอาจไม่สามารถจับทุกปัจจัยได้ → จำกัดความสามารถในการพยากรณ์
🧮 Model Limitations
ยังไม่ได้วิเคราะห์ข้อดี-ข้อเสียของแต่ละโมเดลอย่างละเอียด เช่น:
LSTM: แม่นยำแต่ต้องใช้ทรัพยากรมาก
SVR: ขึ้นอยู่กับการเลือก kernel ที่เหมาะสม
การเข้าใจ trade-off ของแต่ละโมเดลจะช่วยเลือกใช้งานได้เหมาะสม
🧠 Overfitting
มีการป้องกัน เช่น:
PCA เพื่อลดมิติ
Dropout 0.2 ใน LSTM
Regularization λ = 0.5 ใน XGBoost
แต่ข้อมูลการเงินมี noise สูง → ยังมีความเสี่ยง overfitting
การใช้ walk-forward validation ช่วยลด bias แต่ยังสมมติว่าตลาดมีความเสถียร ซึ่งอาจไม่จริงในช่วงวิกฤต
🧠 6. Conclusion: บทสรุปของงานวิจัย
🎯 เป้าหมายหลัก
ระบุ ตัวชี้วัดทางเทคนิค (technical indicators) ที่สำคัญที่สุดในการพยากรณ์ดัชนี S&P 500
ใช้โมเดลหลายประเภท ได้แก่:

XGBoost
Random Forest
Support Vector Regression (SVR)
Long Short-Term Memory (LSTM)
🧪 ข้อมูลและการทดลอง
ใช้ข้อมูลที่ผ่านการลดมิติด้วย PCA เหลือ 88 indicators
ทดลองโมเดลแบบ:
ไม่มี lag (ใช้ราคาปิดของวันปัจจุบัน)
มี lag (ใช้ราคาปิดของวันก่อนหน้า)
📊 ผลลัพธ์สำคัญ
🔹 Indicators ที่ดีที่สุดในแต่ละโมเดล

Model	No Lag	With Lag
SVR	MFI	HLC3
XGBoost	Ichimoku	HWMA
Random Forest	OBV	HLC3
LSTM	HLC3	HLC3

📌 Indicators ที่ปรากฏในทุกโมเดล:

HLC3, TEMA, FWMA, OBV, Bollinger Bands
📋 Indicators ที่ดีที่สุดในแต่ละหมวดหมู่
🔹 Momentum:
RVI, KAMA, SWMA, HLC3
🔸 Trend:
MI, EMA, TEMA, FWMA, HMA, SWMA, HWMA, Ichimoku Cloud
🔺 Volatility:
ATR, Price Distance, Bollinger Bands
🔻 Volume:
MFI, OBV, PVT
📌 การเลือก indicators ที่มีผลต่อ target โดยตรงช่วยลด error metrics ได้อย่างมีนัยสำคัญ

🔮 แนวทางการวิจัยในอนาคต
ขยายชุดข้อมูล:

เพิ่มแหล่งข้อมูล เช่น ข่าว, ดัชนีเศรษฐกิจ, sentiment analysis
ทดสอบกับตลาดอื่น:

เช่น Dow Jones, Nasdaq, FTSE 100 เพื่อดูความสามารถในการ generalize
พัฒนาโมเดล hybrid:

ผสมผสาน ML กับ technical analysis แบบดั้งเดิม เพื่อเพิ่มความแม่นยำ
วิเคราะห์ตามช่วงเวลา:

เปรียบเทียบ performance ในระยะ สั้น / กลาง / ยาว
เพิ่มความสามารถในการตีความโมเดล:

ใช้เทคนิคเช่น SHAP, LIME เพื่อให้ผู้ใช้เข้าใจว่า indicators ใดมีผลต่อการพยากรณ์