University of Essex

Course Name: MSc Data Science and its Application
Module Name: CF969-7-SP/PT : Machine Learning for Finance (Spring term)

CF969- SP ZU Assignment 2

Applying ML Models for Stock Price Prediction

Student Name: Md Ashraful Alam Gazi
Registration Number: 2401510

Date of Submission: 17/04/2025

Applying ML Models for Stock Price Prediction

Registration Number: 2401510

April 17, 2025

1 Introduction

In finance, being able to predict stock returns is very important for investors, analysts, and fund managers. Good predictions can help manage portfolios better, reduce risk, and increase profits. Traditional financial models often assume things like linearity and market efficiency. However, financial markets are always changing and are often non-linear, which makes them a great fit for machine learning methods.

This report explores the application of four distinct machine learning models to predict the next-day returns of selected financial sector stocks. The models include, Linear Regression (Ordinary Least Squares), Support Vector Machine (SVM), Random Forest Regressor, and Feedforward Neural Network.

The main goal of this study is to compare how well each of the selected models can predict stock returns and how interpretable their results are when applied to real stock market data. Each model is tested on the same dataset and evaluated using the same performance metrics, allowing for a fair and practical comparison of their strengths and weaknesses.

To make accurate predictions, the models are trained using technical indicators calculated from historical stock price data. These indicators include 10-day and 50-day moving averages, rolling volatility, momentum, and daily returns. All of these features are commonly used in trading strategies and offer valuable signals for understanding short-term movements in the market.

2 Dataset and Preprocessing

2.1 Dataset Overview

The dataset consists of daily historical closing prices for five major U.S. financial sector stocks: Citigroup (C), PayPal (PYPL), Visa (V), Charles Schwab (SCHW), and Mastercard (MA). The data spans a five-year period from January 1, 2019 to January 1, 2024 and was sourced using the Yahoo Finance API via the `yfinance` Python library.

These five stocks were selected based on a balanced mix of beta values, including both high-risk (e.g., PYPL) and low-risk (e.g., V) assets, to ensure model performance could be observed across a variety of volatility and sensitivity levels.

2.2 Feature Engineering and Dataset Preparation

To build the feature set, technical indicators were calculated for each stock, including daily returns, 10-day and 50-day moving averages, 10-day rolling volatility, and momentum. These are common in technical analysis and help capture short-term price behavior. The prediction target, or label, is the stock's next-day return, making this a supervised regression task.

Before training, all features were standardized using Z-score normalization (`StandardScaler` from `scikit-learn`) to ensure they are on the same scale—important for models like SVM and Neural Networks. The dataset was then split into 80% for training and 20% for testing, allowing the models to learn from one portion and be evaluated fairly on new data.

3 Model Implementation

3.1 Linear Regression

Linear regression was implemented using the Ordinary Least Squares (OLS) method from the `statsmodels` library. This model serves as a baseline due to its simplicity and interpretability. Each stock's return was modeled independently using the 25 engineered features.

A key strength of OLS is its ability to provide coefficient estimates along with p-values, allowing for assessment of statistical significance. In this study, only a small subset of features (e.g., `C_MA50`, `V_Return`, `MA_Return`) showed significance at the 5% level, suggesting that most technical indicators may not have strong linear relationships with next-day returns. This further justified the need to test non-linear models.

3.2 Support Vector Machine

Support Vector Regression (SVR) was implemented using the `SVR` class from `scikit-learn`, with the RBF (Radial Basis Function) kernel. This kernel is suitable for capturing complex, non-linear relationships between features and target variables.

Basic hyperparameters were used across all five models:

- `C = 1.0` (regularization strength)
- `gamma = 'scale'` (auto kernel coefficient)
- `epsilon = 0.001` (insensitivity zone for loss)

Each SVR was trained per stock independently, and predictions were stored for later evaluation.

3.3 Random Forest

Random Forest regression was implemented using `RandomForestRegressor` from `scikit-learn`. Each model used:

- `n_estimators = 100`
- `max_depth = None`

- `random_state = 42`

This ensemble method aggregates multiple decision trees to reduce variance and improve generalization. A valuable feature of this model is the ability to compute feature importances, which helps identify which technical indicators contributed most to the predictions.

3.4 Neural Network

A simple feedforward neural network was built using the Keras API in TensorFlow. Each network had:

- Input layer with 25 features
- Two hidden layers: one with 64 neurons, followed by one with 32 neurons (ReLU activation)
- Output layer with a single neuron for return prediction

The model was compiled with:

- Loss: Mean Squared Error (MSE)
- Optimizer: Adam (learning_rate=0.001)
- Epochs: 50
- Batch size: 32

While powerful, neural networks require more careful tuning and are more sensitive to data size and quality.

4 Model Evaluation

All models were evaluated using three regression metrics:

- Mean Squared Error (MSE): penalizes larger errors more heavily
- Mean Absolute Error (MAE): measures average error magnitude
- R^2 Score: proportion of variance in the target explained by the model

4.1 Test Set Performance

Table 1: Test Set Performance: MSE, MAE, and R^2

Stock	**Model**	**MSE**	**MAE**	**R^2**
C	Linear Regression	0.000282	0.012648	-0.0869
	SVM	0.000523	0.017961	-1.0159
	Random Forest	0.000533	0.015646	-1.0514
	Neural Network	0.005634	0.049811	-20.6955
PYPL	Linear Regression	0.000563	0.017610	-0.0662
	SVM	0.001023	0.024916	-0.9375
	Random Forest	0.000708	0.019608	-0.3417
	Neural Network	0.006881	0.056691	-12.0387
V	Linear Regression	0.000103	0.007906	-0.1222
	SVM	0.000279	0.013087	-2.0502
	Random Forest	0.000208	0.010503	-1.2778
	Neural Network	0.005391	0.055641	-57.9299
SCHW	Linear Regression	0.000733	0.018178	-0.0219
	SVM	0.001028	0.023307	-0.4338
	Random Forest	0.000777	0.018601	-0.0846
	Neural Network	0.004337	0.048408	-5.0494
MA	Linear Regression	0.000115	0.008176	-0.0678
	SVM	0.000381	0.014788	-2.5533
	Random Forest	0.000243	0.011549	-1.2684
	Neural Network	0.003418	0.043766	-30.8719

4.2 Cross-Validation Results

Table 2: Cross-Validation MSE Results (Mean $\pm$ Std)

Stock	**Linear Reg.**	**SVM**	**Random Forest**	**Neural Network**
C	0.001469 ± 0.001742	0.001252 ± 0.000736	0.000853 ± 0.000787	0.071184 ± 0.108693
PYPL	0.001265 ± 0.000646	0.001358 ± 0.000318	0.001028 ± 0.000382	0.061704 ± 0.093098
V	0.000606 ± 0.000482	0.000629 ± 0.000261	0.000441 ± 0.000294	0.100032 ± 0.173842
SCHW	0.001106 ± 0.001127	0.001038 ± 0.000387	0.000689 ± 0.000420	0.021223 ± 0.018609
MA	0.000870 ± 0.000854	0.000859 ± 0.000285	0.000570 ± 0.000339	0.028496 ± 0.032871

Across all five stocks, Linear Regression and Random Forest consistently delivered the lowest MSE and MAE values, although none of the models achieved positive R^2 scores. This reinforces the challenge of short-term return prediction using only technical indicators.

Neural Networks significantly underperformed, likely due to overfitting and lack of sequential awareness. SVMs also struggled to generalize well, reflecting their sensitivity to hyperparameters and noisy inputs.

5 Interpretation and Discussion

This section presents insights from the test set and cross-validation results across four machine learning models: Linear Regression, Support Vector Machine (SVM), Random Forest, and Neural Network. The models were evaluated on five financial stocks: Citigroup (C), PayPal (PYPL), Visa (V), Charles Schwab (SCHW), and Mastercard (MA), using technical indicators as predictive features.

5.1 Model Performance Insights

Linear Regression demonstrated the most stable and consistent performance. Although its R^2 scores were slightly negative across all stocks (e.g., -0.0869 for C, -0.0678 for MA), it consistently achieved the lowest or near-lowest values for both MSE and MAE in test set evaluation. Cross-validation results further supported its reliability, with mean MSEs ranging from 0.000606 (V) to 0.001469 (C) and moderate standard deviations.

Random Forest performed comparably to Linear Regression, often slightly outperforming it in cross-validation (e.g., 0.000441 ± 0.000294 for V). On the test set, its errors were generally close to those of the linear model, but R^2 scores remained negative, such as -1.0514 for C and -1.2684 for MA, indicating modest predictive capability.

Support Vector Machine (SVM) performed worse across all stocks. It exhibited significantly higher MSE and MAE values, and the R^2 scores indicated poor generalization (e.g., -2.0502 for V and -2.5533 for MA). Despite having competitive CV means (e.g., 0.001252 for C), its sensitivity to hyperparameters and inability to handle noise likely contributed to overfitting.

Neural Networks produced the highest error values and worst R^2 scores, with severe overfitting and unstable predictions. For example, on the test set, MSE reached 0.0056 for C and 0.0069 for PYPL, with extremely negative R^2 scores such as -57.9299 for V. Cross-validation confirmed these issues, with high mean errors (e.g., 0.100032 for V) and very large standard deviations, indicating poor generalization.

5.2 Feature Effectiveness and Model Limitations

Across models, features like the 50-day moving average and daily return had the most consistent predictive influence. In contrast, momentum and volatility indicators, while popular in technical trading, contributed little to improving accuracy in this task. This suggests that such features alone may not be sufficient for short-term forecasting.

The uniformly negative R^2 values across models reflect the broader limitation of using only technical indicators to predict next-day returns. These models lacked contextual inputs such as macroeconomic trends, news sentiment, or volume-based signals—factors known to affect market behavior.

Neural Networks, in particular, struggled due to their sensitivity to data size and structure. Without sequence modeling or external features, they failed to extract meaningful patterns from the data.

6 Conclusion

This project investigated the application of four machine learning models—Linear Regression, Support Vector Machine (SVM), Random Forest, and Neural Network—for predicting next-day stock returns of five major financial stocks: Citigroup (C), PayPal (PYPL), Visa (V), Charles Schwab (SCHW), and Mastercard (MA). All models were trained using technical indicators, including moving averages, volatility, momentum, and daily returns, and evaluated using test set performance and cross-validation.

The results showed that **Linear Regression consistently outperformed the other models** in both test set and cross-validation settings. Despite slightly negative R^2 values, it delivered the lowest Mean Squared Error (MSE) and Mean Absolute Error (MAE) for most stocks, demonstrating its robustness and stability in a high-noise financial context.

Random Forest also performed reasonably well and was competitive with the linear model, particularly in cross-validation. However, its slight improvements in accuracy did not justify the added complexity, and it still produced negative R^2 values on the test set.

SVM and Neural Networks, while more flexible and powerful in theory, failed to deliver meaningful improvements. Both models exhibited signs of overfitting, with Neural Networks performing particularly poorly. Their effectiveness was limited by the relatively small size of the dataset and the absence of sequential modeling or richer features beyond price-based indicators.