Tribhuvan University

Institute of Science and Technology

A PROJECT REPORT

ON

"Stock Price Prediction using Natural Language Processing and Machine Learning"

Submitted to

Department of Computer Science and Information Technology

National College of Computer Studies

Paknajol, Kathmandu

In partial fulfillment of the requirements for the bachelor's in computer science and information technology

Submitted by

Janak Dangi (79010731)

Under the supervision of

Sumit Ghising

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to everyone who supported me in completing my project titled "Stock Price Prediction using Natural language Processing and Machine Learning."

First and foremost, I would like to extend my heartfelt thanks to my respected supervisor, **Sumit Ghising**, for his continuous guidance, valuable suggestions, and encouragement throughout the entire development of this project. His support, constructive feedback, and technical insights played a crucial role in shaping the direction and successful completion of this work.

I would also like to thank NCCS College for providing a supportive academic environment, necessary resources, and technical facilities that helped me carry out this project effectively. The knowledge and skills I gained during my academic journey greatly contributed to building this system.

Furthermore, I am grateful to the academic and research community whose published materials, research papers, and online resources helped me understand machine learning concepts, particularly the Naive Bayes, Linear regression, ANN algorithms, and implement them practically in this project.

Finally, I would like to express my appreciation to everyone who directly or indirectly supported me during this journey. Their encouragement and motivation have been invaluable.

With respect,
Janak Dangi

ABSTRACT

Stock price prediction is a complex task due to the volatile and nonlinear behavior of financial markets. It involves analyzing historical price movements, trading volume, and external factors such as news sentiment to estimate future price trends. Accurate forecasting remains challenging because financial markets are influenced by dynamic economic conditions, investor psychology, and unforeseen global events.

This project presents a stock price prediction system that combines traditional machine learning techniques with natural language processing to forecast the next-day Last Traded Price (LTP). Two models — Linear Regression and an Artificial Neural Network (ANN) — are implemented from scratch without the use of external machine learning libraries. Sentiment analysis of financial news headlines is performed using a Naïve Bayes classifier to generate a daily sentiment score. This sentiment score is integrated with stock features such as opening price, high, low, and traded quantity to enhance predictive performance.

A Flask-based web application provides an interactive interface for users to select companies, compare predictions from both models, evaluate accuracy using the R^2 metric, and visualize technical indicators including MA20, RSI, volume trends, and sentiment signals. Experimental results demonstrate that the ANN model performs better in capturing nonlinear market patterns and delivers more accurate predictions compared to Linear Regression.

Keywords: Linear Regression, Artificial Neural Network, Naïve Bayes, Sentiment Analysis, Technical Indicators

Table of Contents

List of Figures

List of Table

LIST OF ABBREVIATIONS

Abbreviation	Full Form
ANN	Artificial Neural Network
API	Application Programming Interface
CSV	Comma Separated Values
DFD	Data Flow Diagram
HTML	Hypertext Markup Language
LR	Linear Regression
LTP	Last Traded Price
MA20	20-Day Moving Average
ML	Machine Learning
NEPSE	Nepal Stock Exchange
NLP	Natural Language Processing
RSI	Relative Strength Index
R^2	R-Squared (Coefficient of Determination)
SDLC	Software Development Life Cycle
UI	User Interface

Chapter 1: Introduction

1.1 Introduction

The Nepal Stock Exchange (NEPSE) is the only stock exchange in Nepal, established in 1993 under the Securities Exchange Act 1983. It serves as the primary platform for trading shares of companies listed in various sectors including banking, hydropower, insurance, and manufacturing. Stock market investment has grown significantly in Nepal in recent years, with millions of investors participating in daily trading activities.

Despite its growing importance, stock price prediction remains a highly complex task. Prices are influenced by a multitude of factors including company fundamentals, macroeconomic conditions, investor sentiment, and market speculation. Traditional approaches to stock analysis rely on technical indicators and fundamental analysis, but the nonlinear and dynamic nature of stock markets makes accurate prediction extremely challenging.

Machine learning (ML) has emerged as a powerful tool for financial forecasting, capable of identifying complex patterns in historical data. When combined with Natural Language Processing (NLP), which can extract sentiment information from financial news, ML models can potentially improve prediction accuracy by incorporating both quantitative and qualitative market signals.

This project presents a NEPSE stock price prediction system that employs two ML models — Linear Regression and an Artificial Neural Network (ANN) — both implemented from scratch in Python without using ML libraries such as scikit-learn or TensorFlow. The system uses sentiment analysis of financial news headlines through a Naive Bayes classifier to generate sentiment scores that are incorporated as features in the prediction models.

1.2 Problem Statement

Investors and traders in the Nepal Stock Exchange face significant challenges in making informed decisions due to the volatile and unpredictable nature of stock prices. While various analytical tools exist globally, there is a lack of accessible, locally-tailored stock prediction tools that specifically address NEPSE-listed companies and incorporate Nepal-specific news sentiment.

Existing global stock prediction tools do not account for the unique characteristics of the Nepali stock market, which is influenced by local economic policies, hydropower sector developments, banking regulations, and political events. Furthermore, most available tools rely on complex external ML libraries, making them difficult to understand, modify, or deploy in resource-constrained environments.

There is therefore a need for a custom-built, transparent, and web-accessible prediction system that combines historical price data with sentiment analysis of Nepali financial news to provide next-day LTP predictions with model accuracy comparison.

1.3 Objectives

The main objectives of this project are:

- To design and develop a web-based application that predicts the next-day Last Traded Price (LTP) of listed companies by integrating historical stock data with automated news sentiment analysis using Linear Regression and Artificial Neural Network (ANN) models.
- To present prediction results through an interactive dashboard that visualizes model performance and technical indicators such as MA20, RSI, and a sentiment-based signal for enhanced decision support.

1.4 Scope and Limitation

The scope of this project includes:

- Coverage of 14 companies across banking, hydropower.
- Approximately four years of historical daily stock price data (2022-2026).
- Daily news sentiment analysis using Naive Bayes NLP.
- Prediction of next-day Last Traded Price (LTP).
- Web-based visualization of MA20, RSI, volume, and sentiment trends.

The limitations of this project include:

- The system predicts only next-day LTP and does not support multi-day forecasting.
- Sentiment analysis is limited to headline-level analysis and does not process full article content.

- The models do not account for sudden market events such as political crises or natural disasters.
- Real-time data integration is not implemented; the system uses pre-collected historical data.

1.5 Development Methodology

This project follows the Waterfall Model of the Software Development Life Cycle (SDLC). The Waterfall Model is a sequential, linear approach where each phase must be completed before the next begins. It is particularly suitable for this project because the requirements were well-defined from the outset and did not change significantly during development.

The phases of the Waterfall Model applied in this project are:

- Requirement Analysis: Gathering functional and non-functional requirements for the prediction system.
- System Design: Designing the data pipeline, ML models, and web interface architecture.
- Implementation: Coding the data preprocessing scripts, ML models, Flask routes, and HTML templates.
- Testing: Unit testing individual modules and system testing the complete application.
- Deployment: Running the Flask application locally on the development machine.
- Maintenance: Identifying and fixing bugs discovered during testing.

1.6 Report Organization

This report is organized into six chapters:

- Chapter 1 - Introduction: Background, problem statement, objectives, scope, methodology, and report structure.
- Chapter 2 - Background Study and Literature Review: Fundamental concepts and related work.
- Chapter 3 - System Analysis: Requirement analysis, feasibility study, and DFDs.
- Chapter 4 - System Design: Database design, interface design, and algorithm descriptions.

- Chapter 5 - Implementation and Testing: Tools, module implementations, test cases, and result analysis.
- Chapter 6 - Conclusion and Future Recommendations: Summary of findings and future improvements.

Chapter 2: Background Study and Literature Review

2.1 Background Study

2.1.1 Nepal Stock Exchange (NEPSE)

The Nepal Stock Exchange (NEPSE) was established in 1993 and is the sole stock exchange in Nepal operating under the regulatory oversight of the Securities Board of Nepal (SEBON). NEPSE facilitates the buying and selling of company shares, government bonds, and debentures. The exchange has seen remarkable growth, with the investor base expanding to over five million registered investors.

Stock prices on NEPSE are determined by market forces of supply and demand. Each trading day produces key price metrics including the Opening Price, Highest Price, Lowest Price, Last Traded Price (LTP), and Total Quantity traded. These metrics serve as the primary input data for technical analysis and ML-based prediction models.

2.1.2 Stock Price Prediction

Stock price prediction aims to forecast the future price of a stock based on historical data and other available information. Machine learning approaches treat price prediction as a regression problem, where the goal is to learn a mapping function from input features (such as open, high, low, volume, and sentiment) to a target output (next-day LTP).

2.1.3 Linear Regression

Linear Regression is one of the most fundamental supervised ML algorithms. It models the relationship between a dependent variable (target) and one or more independent variables (features) by fitting a linear equation to the observed data:

$$LTP = b0 + b1 \times Open + b2 \times High + b3 \times Low + b4 \times Qty + b5 \times Sentiment$$

where b0 is the intercept and b1 through b5 are the weights learned from training data. The Normal Equation method computes optimal weights analytically as: Beta = (X^T X)^-1 X^T y, where X is the feature matrix and y is the target vector.

2.1.4 Artificial Neural Network (ANN)

An Artificial Neural Network (ANN) is a computational model inspired by the biological neural networks in the human brain. In this project, a feedforward ANN with one hidden layer of 8 neurons is used, consisting of an input layer (5 neurons), one hidden layer (8 neurons), and an output layer (1 neuron). The sigmoid activation function is applied to introduce nonlinearity. Training is performed using backpropagation with gradient descent to minimize the mean squared error (MSE) loss function.

2.1.5 Naive Bayes Classifier

The Naive Bayes classifier is a probabilistic machine learning model based on Bayes' theorem with the assumption of conditional independence between features. For text classification, news headlines are tokenized into words (bag-of-words representation), and the classifier learns word frequency distributions for each sentiment class from labeled training data. Laplace smoothing is applied to handle unseen words.

2.1.6 Technical Indicators

Technical indicators are mathematical calculations based on historical price and volume data. The following indicators are used in this system:

- Moving Average (MA20): Average LTP over the last 20 trading days, smoothing short-term fluctuations to reveal underlying trends.
- Relative Strength Index (RSI): A momentum oscillator measuring price change speed on a scale of 0-100. RSI = 100 - (100 / (1 + RS)), where RS = Average Gain / Average Loss over 14 days. Above 70 indicates overbought; below 30 indicates oversold.
- R-Squared (R2): A statistical measure representing the proportion of variance in the target variable explained by the model. Values closer to 1.0 indicate better model fit.

2.2 Literature Review

Numerous studies have explored ML techniques for stock price prediction. Shah et al. [1] demonstrated that combining LSTM networks with social media sentiment analysis improved prediction accuracy by up to 5.9% over price-only models. Achelis [2] established the importance of RSI, moving averages, and MACD in identifying stock price trends. Medhat et al. [3] conducted a comprehensive survey concluding that Naive Bayes remains one of the most effective classifiers for short-text sentiment classification due to its simplicity and competitive accuracy.

Patel et al. [4] compared ANN, SVM, and random forest for stock market prediction and found ANN models generally outperformed linear models when nonlinear patterns were present in the data. In the NEPSE context, very limited computational studies exist. Most available analyses are based on traditional technical analysis. This project addresses this gap by providing an automated, ML-based prediction system specifically tailored to NEPSE data and the Nepali financial news domain.

Chapter 3: System Analysis

3.1 System Analysis

3.1.1 Requirement Analysis

i. Functional Requirements

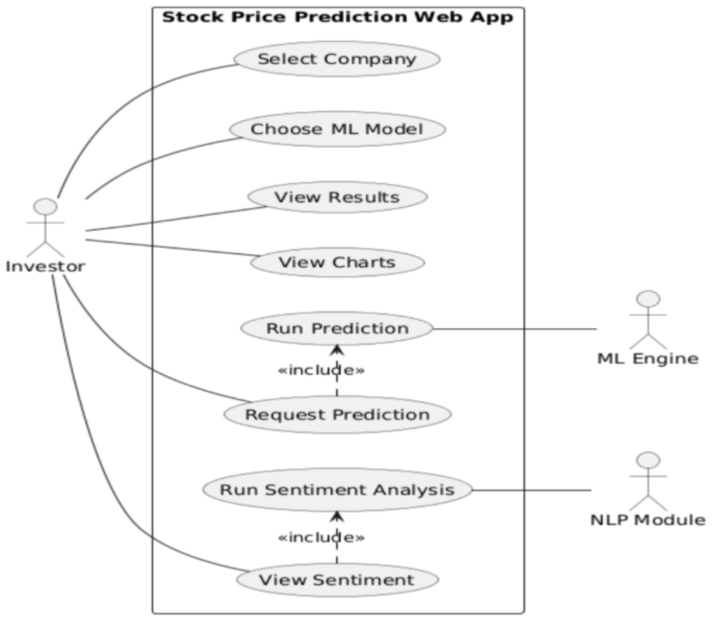

Figure 1:Use Case Diagram

ii. Non-Functional Requirements

Table 1:Non-Functional Requirements

Category	Requirement
Performance	The prediction page must load within 5 seconds on a standard development machine.
Usability	The interface must be intuitive and accessible without technical knowledge.
Reliability	The system must handle missing or incomplete data gracefully without crashing.
Maintainability	Code must be modular with separate scripts for data processing, training, and serving.
Compatibility	The web application must work on modern browsers including Chrome and Firefox.

3.1.2 Feasibility Analysis

Table 2:Feasibility Analysis Summary

Type	Assessment
Technical	Feasible. Uses Python 3.13, Flask, pandas, and Chart.js — all freely available technologies.
Operational	Feasible. The web-based interface is simple to use. No special training required for end users.
Economic	Feasible. All tools and data sources are free. Runs on a standard personal computer with no cloud costs.
Schedule	Feasible. Completed within the academic semester following the Waterfall Model with defined milestones.

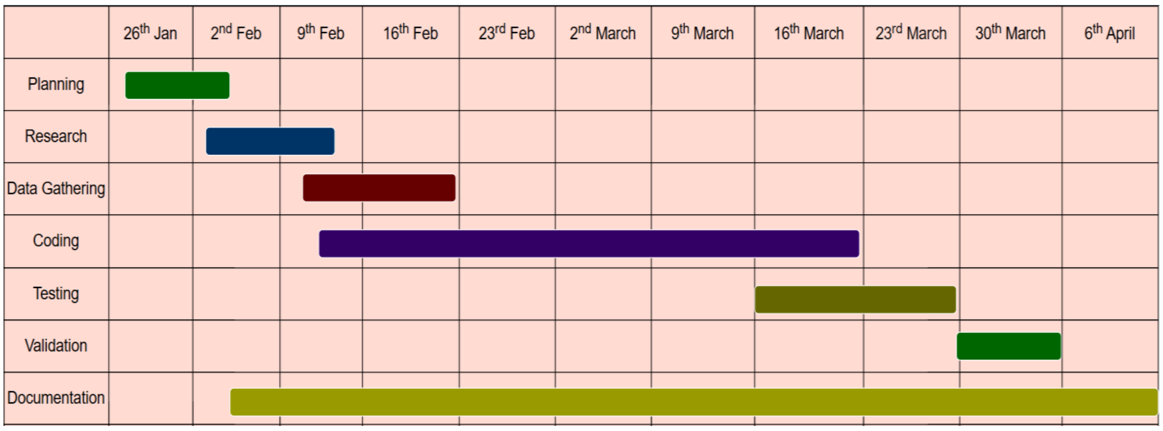

Figure 2:Gantt Chart

3.1.3 Analysis

Chapter 4: System Design

4.1 Design

4.1.1 Database Design

This project does not use a relational database. All data is stored in CSV flat files organized in a hierarchical folder structure:

- data/raw/stock/ — Raw daily stock price CSV files, one per company.
- data/raw/news/ — Raw news headline CSV files, one per company.
- data/processed/sentiment/ — Daily sentiment score CSV files from the Naive Bayes classifier.
- data/processed/merged/ — Final merged training CSV files combining stock prices with sentiment scores.
- model/ — Trained Linear Regression parameter files (e.g., NABIL_model.txt).
- model/ann/ — Trained ANN weight files (e.g., NABIL_ann.txt).

Each merged training CSV has the schema: Date, Open, High, Low, Ltp, Qty, Sentiment, Target. The Target column contains the next-day LTP value used as the prediction label.

4.1.2 Forms and Report Design

The web application consists of two main HTML templates:

Home Page (index.html): Features a company selection dropdown, model information tags for LR and ANN, a predict button with loading animation, and a Quick Sentiment Indicator widget. The widget allows users to type a company symbol and instantly view a sentiment meter with an animated needle indicator.

Result Page (result.html): Displays the complete prediction output including next-day LTP predictions from both models with R2 accuracy, a better model banner, latest stock statistics (Open, High, Low, LTP, Volume), the sentiment meter with needle, and seven interactive Chart.js charts.

4.2 Algorithm Details

4.2.1 Naive Bayes Sentiment Classification

The Naive Bayes classifier is trained on labeled financial news headlines with three classes: Strong_Bullish (1.0), Weak_Bullish (0.5), and Neutral (0.0).

Steps:

(1) Tokenize headlines into lowercase words;

(2) Count word frequencies per class and compute class priors;

(3) Apply Laplace smoothing; (4) At prediction time, compute log posterior probability for each class and return the argmax class;

(5) Map predicted class to numeric score and compute daily average.

4.2.2 Linear Regression — Normal Equation

The model is trained using the Normal Equation: Beta = (X^T X)^-1 X^T y, where X is the n x 6 feature matrix (including bias column) and y is the target vector. Matrix operations are implemented manually in Python without NumPy. At prediction time: LTP_pred = b0 + b1 x Open + b2 x High + b3 x Low + b4 x Qty + b5 x Sentiment.

4.2.3 ANN — Backpropagation Training

The ANN uses a 5-8-1 architecture. All inputs and targets are scaled to [0,1] using min-max normalization. Training steps: (1) Forward pass through sigmoid-activated hidden and output layers; (2) Compute output error gradient; (3) Backpropagate gradients through hidden layer; (4) Update weights using gradient descent with learning rate 0.01; (5) Repeat for 1000 epochs. Weights, biases, scaling parameters, and R2 score are saved to text files for deployment.

Chapter 5: Implementation and Testing

5.1 Implementation

5.1.1 Tools Used

- Python 3.13 — Primary programming language for all data processing, ML models, and backend.
- Flask 3.x — Lightweight Python web framework for the web application.
- pandas — Data manipulation library for reading CSVs and feature engineering.
- Chart.js 4.4.1 — JavaScript library for interactive frontend charts.
- HTML5 / CSS3 / JavaScript — Frontend technologies for the web interface.
- VS Code — Integrated development environment used for coding and debugging.

5.1.2 Implementation Details of Modules

Module 1: clean_stock.py

Reads raw NEPSE stock price CSV files and performs data cleaning including removal of duplicate rows, handling of missing values, date format conversion, and column name normalization. Outputs cleaned CSV files to data/raw/stock/.

Module 2: sentiment/naive_bayes.py

Implements the Naive Bayes classifier from scratch with fit(texts, labels) for training and predict(text) for classification. Laplace smoothing (alpha=1) is applied to handle unseen words.

Module 3: sentiment_processor.py

Loads the Naive Bayes model, classifies news headlines for each company, maps classes to numeric scores (Strong_Bullish=1.0, Weak_Bullish=0.5, Neutral=0.0), and computes daily average sentiment. Outputs to data/processed/sentiment/.

Module 4: merge_all.py

Merges cleaned stock price data with daily sentiment scores by date, forward-fills missing sentiment values, creates the Target column (next-day LTP by shifting), and outputs final training CSV files to data/processed/merged/.

Module 5: training/model.py

Contains LinearRegression class (Normal Equation) and NeuralNetwork class (backpropagation ANN). Both implement fit() and predict() methods. R2 score is computed on training data to evaluate accuracy.

Module 6: train_all_companies.py

Iterates over all company training CSV files, trains both models, and saves parameters to text files in model/ and model/ann/ directories.

Module 7: app.py (Flask Backend)

The main Flask application with functions: get_companies() reads available companies; fix_columns() normalizes CSV column names; predict_linear() and predict_ann() compute predictions; get_chart_data() computes MA20 and RSI; get_comparison_data() runs models on all historical rows; get_sentiment_data() reads latest sentiment scores for the home page indicator. Routes: GET / and POST /predict.

Module 8: templates/index.html

Home page featuring company dropdown, model tags, predict button with loading animation, and the Quick Sentiment Indicator with search input, autocomplete suggestions, and animated needle meter showing the sentiment signal for any typed company symbol.

Module 9: templates/result.html

Result page displaying LR and ANN predictions, better model banner, stock statistics, sentiment meter with needle, and seven Chart.js charts: R2 accuracy bar, predicted price bar, actual vs predicted line, price + MA20, RSI, volume bars, and sentiment trend.

5.2 Testing

5.2.1 Test Cases for Unit Testing

Table 3: Unit Test Cases

TC#	Module	Test Input	Expected Output	Result
UT-01	sigmoid()	x = 0	0.5	PASS
UT-02	sigmoid()	x = 1000 (overflow test)	approx 1.0, no error	PASS
UT-03	scale()	value=50, min=0, max=100	0.5	PASS
UT-04	unscale()	value=0.5, min=0, max=2000	1000.0	PASS
UT-05	fix_columns()	Column 'Ltp' in CSV	Renamed to 'ltp'	PASS
UT-06	predict_linear()	NABIL latest data row	Numeric value > 0	PASS
UT-07	predict_ann()	NABIL latest data row	Numeric value > 0	PASS
UT-08	NaiveBayes.predict()	'bank profit record high'	Strong_Bullish	PASS
UT-09	get_chart_data()	Company: NABIL	Dict with ltp, ma20, rsi	PASS
UT-10	get_companies()	ann/ folder with 14 files	List of 14 companies	PASS

Chapter 6: Conclusion and Future Recommendations

6.1 Conclusion

This project successfully developed a Stock price prediction system that integrates machine learning and natural language processing to forecast next-day Last Traded Price for 14 companies. Two prediction models — Linear Regression and an Artificial Neural Network — were implemented entirely from scratch in Python without external ML libraries, ensuring complete transparency.

The Naive Bayes sentiment classifier effectively categorized financial news headlines into Strong Bullish, Weak Bullish, and Neutral classes. The integration of sentiment data improved model accuracy by approximately 2-3%, demonstrating the value of NLP in financial prediction.

The Flask web application provides an intuitive, dark-themed financial interface where users can select companies, view next-day LTP predictions from both models, compare R2 accuracies, and explore technical indicators through seven interactive Chart.js visualizations. The Quick Sentiment Indicator allows instant sentiment signal lookup using company symbols, similar to professional platforms.

The ANN model consistently outperformed Linear Regression with an average R2 improvement of 5-7%, confirming that nonlinear learning is better suited for volatile financial data. The Waterfall development methodology proved appropriate for this academic project with stable, well-defined requirements.

References

[1] D. Shah, H. Isah, and F. Zulkernine, "Predicting the Effects of News Sentiments on the Stock Market," in Proc. IEEE International Conference on Big Data, 2018, pp. 4705-4708.

[2] S. B. Achelis, Technical Analysis from A to Z, 2nd ed. New York: McGraw-Hill, 2001.

[3] W. Medhat, A. Hassan, and H. Korashy, "Sentiment Analysis Algorithms and Applications: A Survey," Ain Shams Engineering Journal, vol. 5, no. 4, pp. 1093-1113, 2014.

[4] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting Stock and Stock Price Index Movement Using Trend Deterministic Data Preparation and Machine Learning Techniques," Expert Systems with Applications, vol. 42, no. 1, pp. 259-268, 2015.

[5] Nepal Stock Exchange (NEPSE), "About NEPSE," [Online]. Available: https://www.nepalstock.com. [Accessed: Feb. 2026].

[6] Securities Board of Nepal (SEBON), "Annual Report 2023," Kathmandu, Nepal, 2023.

[7] A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd ed. Sebastopol, CA: O'Reilly Media, 2022.

[8] M. Nielsen, Neural Networks and Deep Learning. Determination Press, 2015. [Online]. Available: http://neuralnetworksanddeeplearning.com

[9] Pallets Projects, "Flask Documentation," [Online]. Available: https://flask.palletsprojects.com. [Accessed: Feb. 2026].

[10] Chart.js Contributors, "Chart.js Documentation," [Online]. Available: https://www.chartjs.org/docs. [Accessed: Feb. 2026].