Project Name: Stock Price Prediction using Machine Learning

Group No. G5

Group Member: 27,28,29,31,32

Introduction

This project is about predicting the future movement of Tesla's stock price using Machine Learning (ML). The idea is to find out whether the stock price will go up or down the next day based on past data. In simple words, our ML model tries to give a signal whether buying the stock today would be profitable or not. We used daily stock data from Tesla between 2010 and 2017. Each record includes opening price, highest and lowest prices, closing price, and the trading volume. This project walks through every step – data analysis, feature creation, model training, and evaluating which ML model performs the best.

1. Objective

The main goal of this project is to predict whether the next day's closing price of Tesla will be higher or lower than today's closing price. To achieve this, we will: 1. Collect and clean historical Tesla stock data. 2. Analyze how stock prices move over time. 3. Create new useful features for better prediction. 4. Train different ML models. 5. Compare their performance to find the best one.

2. Dataset Description

The dataset used in this project contains Tesla's stock prices from 1st January 2010 to 31st December 2017. Columns in the dataset: - Date: The trading date. - Open: The price at which the stock opened. - High: The highest price on that day. - Low: The lowest price on that day. - Close: The price at market closing. - Adj Close: Adjusted closing price (same as Close, so removed later). - Volume: The total number of stocks traded that day. There are 1692 rows and 7 columns. There are no missing values, but some dates are skipped since the stock market is closed on weekends and holidays.

3. Tools and Libraries

We used Python libraries to make the work easier: - NumPy: For fast mathematical calculations. Pandas: For data handling and analysis. - Matplotlib and Seaborn: For data visualization and graphs. - Scikit-learn: For building and testing ML models. - XGBoost: For powerful gradient boosting model to improve accuracy.

4. Exploratory Data Analysis (EDA)

EDA means understanding the data better before using it in ML models. We plotted Tesla's closing price and saw an overall upward trend, meaning the stock value increased over the years. We checked how the prices and volume were distributed and found: - Prices were not evenly spread;

they had two main value ranges. - Volume data had some very high outlier values. We also compared stock prices by year and found that prices almost doubled between 2013 and 2014. Quarter-end months (March, June, September, December) usually had higher prices since companies release their financial results then.

5. Feature Engineering

Feature Engineering means creating new columns from existing data to help the model learn better. We created: - day, month, year: Extracted from the Date column. - is_quarter_end: Shows if the month is the end of a financial quarter. - open-close: Difference between opening and closing prices. - low-high: Difference between lowest and highest prices. - target: 1 if next day's closing price is higher than today, else 0. This "target" column becomes our label for prediction. The data was balanced (almost equal 1s and 0s), which is ideal for training.

6. Correlation Analysis

We checked how different columns are related to each other using a heatmap. - OHLC (Open, High, Low, Close) values were strongly related to each other. - The new features we created were not highly correlated, so they added new useful information.

7. Data Preprocessing

Before training, we scaled all numeric values to keep them within the same range. This makes the model learn faster and more accurately. Then, we divided the data: - 90% for training - 10% for testing (validation) This helps us test how well our model performs on unseen data.

8. Model Development

We trained three machine learning models: 1. Logistic Regression – Simple and interpretable. 2. Support Vector Classifier (SVC) – Works well with complex data. 3. XGBoost Classifier – A powerful boosting algorithm that combines many weak models to form a strong one. Results (ROC-AUC score): - Logistic Regression: 0.54 (Validation) - SVC: 0.45 (Validation) - XGBoost: 0.57 (Validation) Observation: XGBoost performed best but slightly overfitted (very high training accuracy). Logistic Regression was more stable.

9. Model Evaluation

We used ROC-AUC score and confusion matrix to check how well the models predicted. A confusion matrix helps visualize how many predictions were correct or wrong. Logistic Regression gave balanced results with fewer errors on unseen data.

10. Conclusion

- The new engineered features helped improve prediction quality. - XGBoost gave the highest accuracy but needed fine-tuning to avoid overfitting. - Logistic Regression worked consistently and

was easy to interpret. In the future, the accuracy can be improved by: 1. Using deep learning models like LSTM or GRU which handle time-based data better. 2. Adding technical indicators such as moving averages, RSI, and MACD. 3. Including external market factors or news data to make predictions more realistic.