Project 4- Stock Price Prediction: Do I Buy or Sell?

Rachael Innes

Kajal Jain

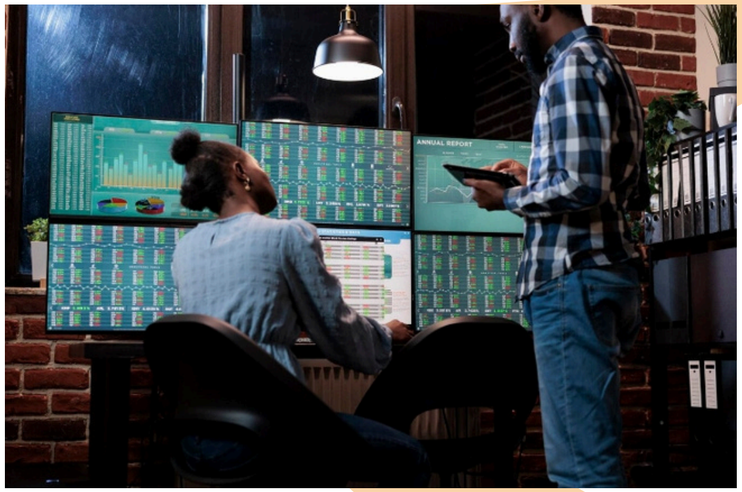
Jordan Chia

Tsz Hin(Raymond) Tang

Disclaimer

- This project is for educational purposes only and does not constitute financial or investment advice.
- The predictions made by the models are based on historical data and may not reflect future market conditions.
- Please consult a financial advisor before making any investment decisions.

Project Objective



The main objectives of this project were to:

- Develop a machine learning model to predict short-term stock movements
- Provide buy or sell signals for stocks based on historical data
- Develop an interactive tool that could be used to predict outcomes of investment (Buy or Sell).

Who are the users?

- **Retail investors seeking guidance in trading**
- **Financial advisors**
- **Data enthusiasts interested in financial modeling**

Why is machine learning helpful ?

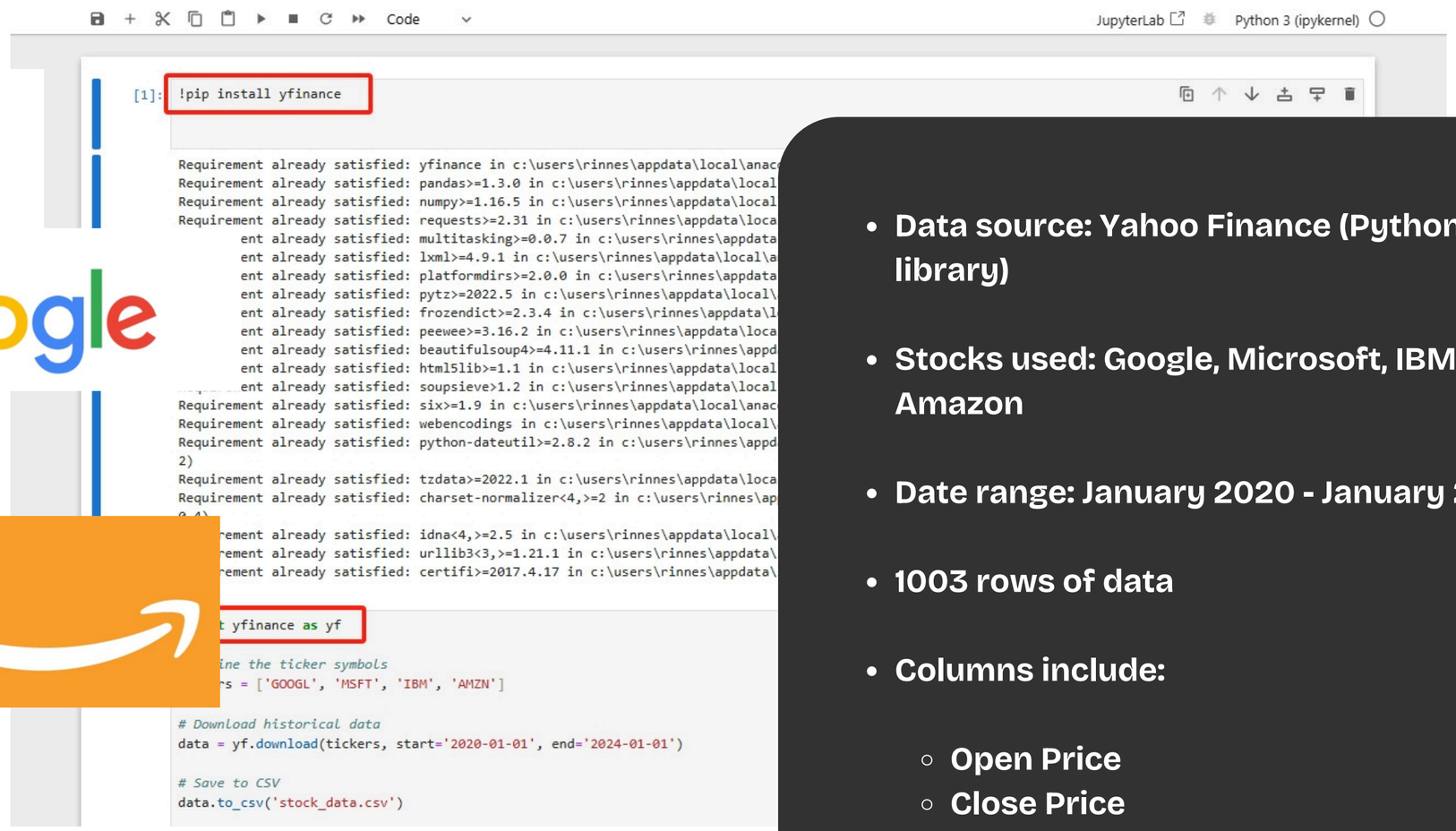
Using machine learning algorithms to analyse stock performance helps users get:

- **Recommendations for investments (Buy or Sell)**
- **Instant insights to make decisions**
- **Removes the need to use company financial documents like below**

	Quarter Ended March 31, 2023	2024
Google Search & other	$ 40,359	$ 46,156
YouTube ads	6,693	8,090
Google Network	7,496	7,413
Google advertising	54,548	61,659
Google subscriptions, platforms, and devices	7,413	8,739
Google Services total	61,961	70,398
Google Cloud	7,454	9,574
Other Bets	288	495
Hedging gains (losses)	84	72
Total revenues	$ 69,787	$ 80,539
Total TAC	$ 11,721	$ 12,946
Number of employees	190,711	180,895

	Analysts' Estimates for Q3 2024	Q2 2024	Q3 2023
Revenue	$86.41 billion	$84.74 billion	$76.69 billion
Earnings Per Share	$1.85	$1.89	$1.55
Net Income	$23.03 billion	$23.62 billion	$19.69 billion

Data Collection



- Data source: Yahoo Finance (Python library)
- Stocks used: Google, Microsoft, IBM, Amazon
- Date range: January 2020 - January 2024
- 1003 rows of data
- Columns include:
  - Open Price
  - Close Price
  - Low Price
  - High Price
  - Adjusted Close Price
  - Volume (trading volume of shares)

Data Cleaning

- Set "Date" as an index
- Drop rows with missing values
- Standard scaling for normalization
- Created a “clean” CSV file

[20]:
```python
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Load the dataset
df = pd.read_csv('stock_data.csv', header=[0, 1], index_col=0)

# Fix the multi-level columns
df.columns = df.columns.map('_'.join)

# Reset the index to get 'Date' as a column
df.reset_index(inplace=True)

# Fix the date format
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Set the date as the index
df.set_index('Date', inplace=True)

# Drop any rows with missing values
df.dropna(inplace=True)

# Select the columns to scale (excluding 'Ticker')
columns_to_scale = df.columns.difference(['Ticker_AMZN'])

# Initialize the scaler
scaler = StandardScaler()

# Scale the selected columns
df[columns_to_scale] = scaler.fit_transform(df[columns_to_scale])

# Save the cleaned and scaled data to a new CSV file
df.to_csv('cleaned_stock_data.csv')

print("Data cleaning and scaling complete. The cleaned data is saved to 'cleaned_stock_data.csv'.")
```

```
Data cleaning and scaling complete. The cleaned data is saved to 'cleaned_stock_data.csv'.
```

Database Creation

- PostgreSQL database: "Stock Market Project"
- Created a schema for the tables
- SVS files uploaded via Python for further processing

[8]:
```
import pandas as pd
from sqlalchemy import create_engine

# Create a connection to the PostgreSQL database
engine = create_engine('postgresql+psycopg2://postgres:Arabella08@localhost:5432/Stock_Market_Project')

# Query the data
query = "SELECT * FROM cleaned_stock_data"
df = pd.read_sql(query, engine)
```

Machine Learning Model Overview

- Environment setup: Python libraries (pandas, sklearn, etc.)
- RandomForestClassifier for initial modelling.
- Tensor Flow was also used as an alternative modelling
- Threshold set for buy/sell signals (2% price change)

Random Model Classifier Model Training & Evaluation

- 1 (Buy): Indicates that the price is expected to increase above a certain threshold, (0.02%), suggesting that the stock should be bought.
- 0 (Sell): Indicates that the price is expected to decrease below a certain threshold (0.02%), suggesting that the stock should be sold.
- PRECISION - is important in this context, as high precision means that when the model predicts a "buy" signal, it is likely to be correct.

Results for AMZN:

	precision	recall	f1-score	support
0.0	0.70	0.74	0.72	87
1.0	0.70	0.66	0.68	80
accuracy			0.70	167
macro avg	0.70	0.70	0.70	167
weighted avg	0.70	0.70	0.70	167

Results for GOOGL:

	precision	recall	f1-score	support
0.0	0.53	0.61	0.57	67
1.0	0.67	0.60	0.63	89
accuracy			0.60	156
macro avg	0.60	0.60	0.60	156
weighted avg	0.61	0.60	0.60	156

Results for IBM:

	precision	recall	f1-score	support
0.0	0.75	0.65	0.70	95
1.0	0.66	0.75	0.70	84
accuracy			0.70	179
macro avg	0.70	0.70	0.70	179
weighted avg	0.70	0.70	0.70	179

Results for MSFT:

	precision	recall	f1-score	support
0.0	0.64	0.66	0.65	82
1.0	0.65	0.62	0.63	82
accuracy			0.64	164
macro avg	0.64	0.64	0.64	164
weighted avg	0.64	0.64	0.64	164

Tensor Flow using deep neural network Model Training & Evaluation

- Used regularization techniques (BatchNorm + Dropout+ early stopping)
- Parameters fine tuned for optimizer during model compilation using sequential method model

Training Set Classification Report:

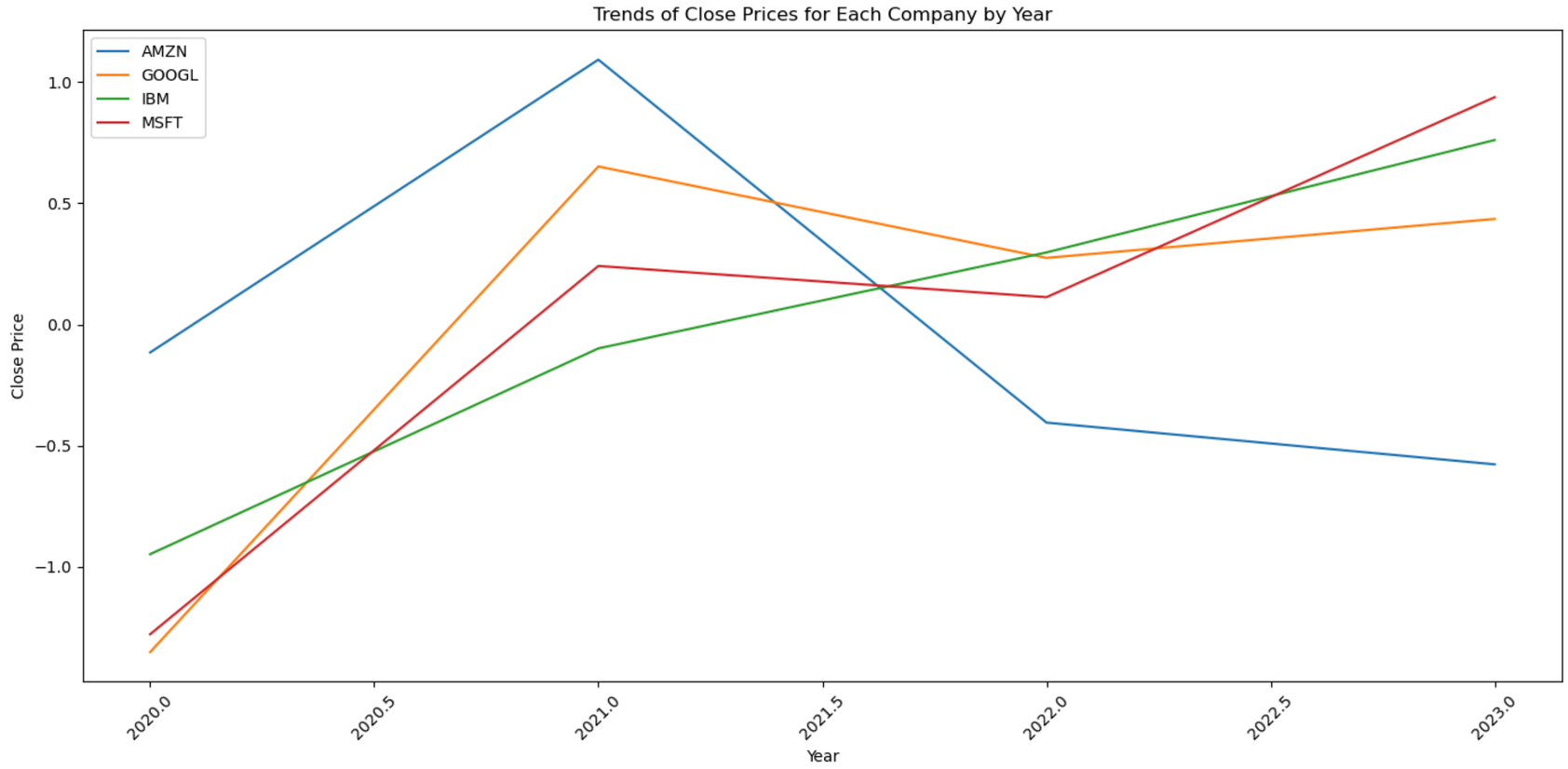
	precision	recall	f1-score	support
0	0.63	0.27	0.38	441
1	0.46	0.80	0.58	347
accuracy			0.50	788
macro avg	0.54	0.53	0.48	788
weighted avg	0.55	0.50	0.47	788

Test Set Classification Report:

	precision	recall	f1-score	support
0	0.81	0.13	0.23	128
1	0.37	0.94	0.53	70
accuracy			0.42	198
macro avg	0.59	0.54	0.38	198
weighted avg	0.66	0.42	0.34	198

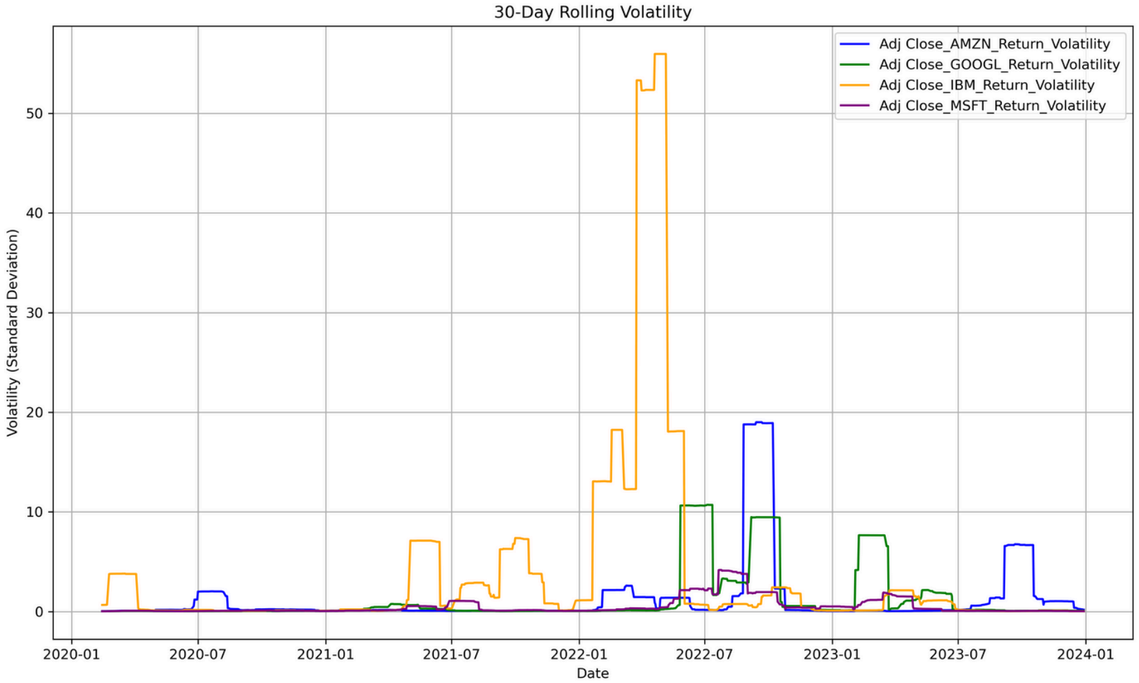
Data Visualization

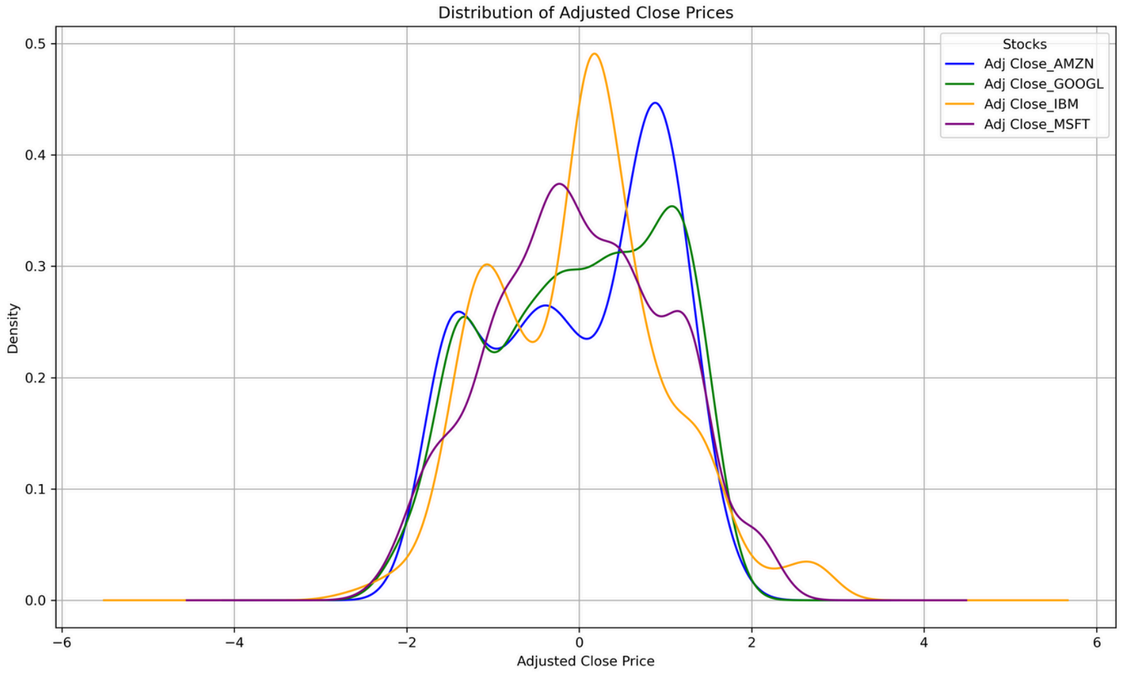
- Line graph of stock prices (2020-2023)
- Overview of trends for each company

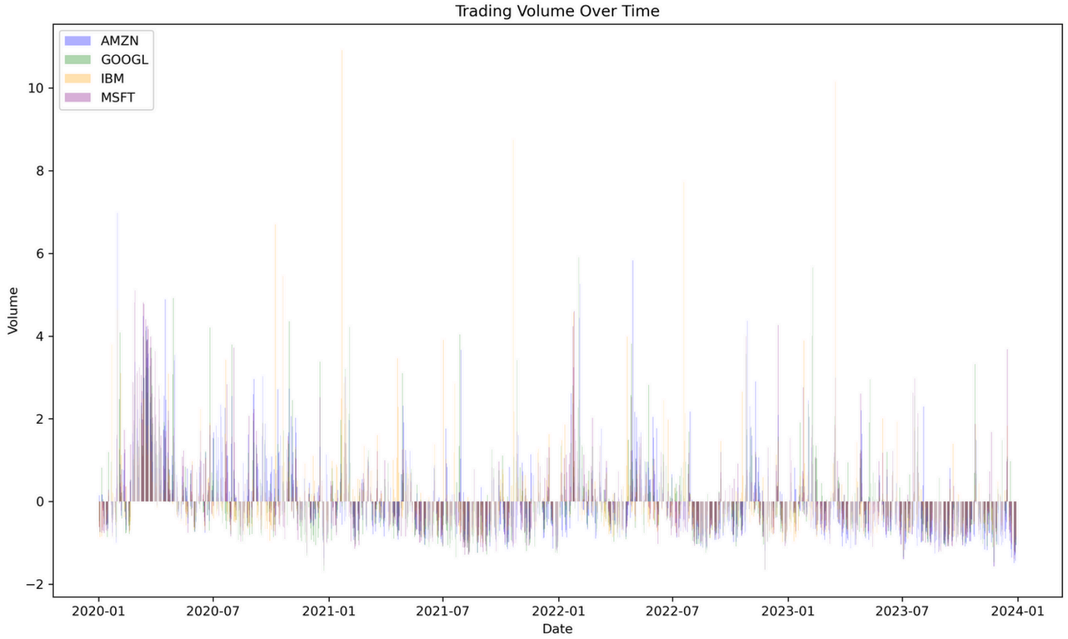


Data Visualization

- Distribution of adjusted close price
- Trading valume over time
- 30 Day Rolling Volatility

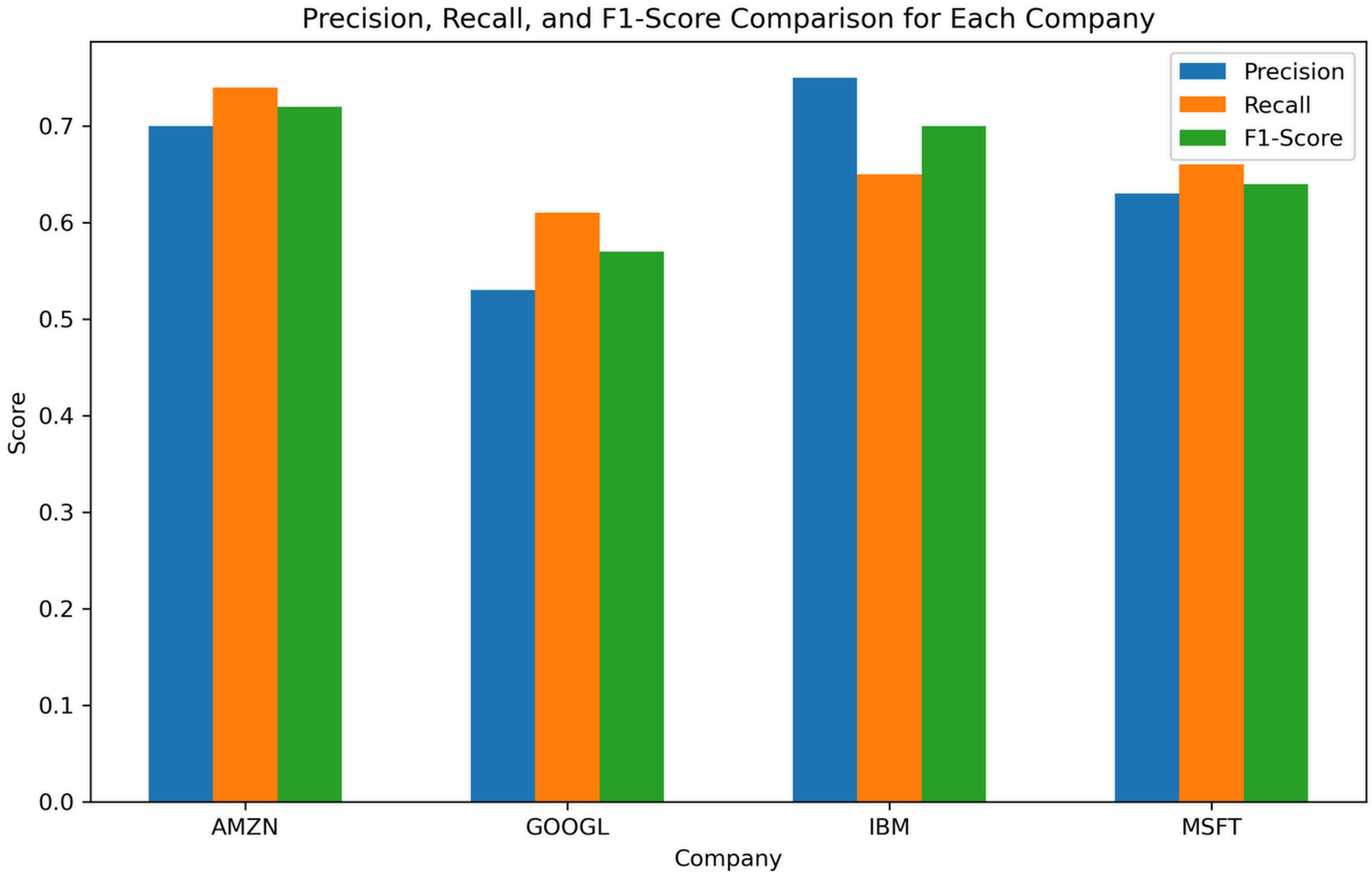




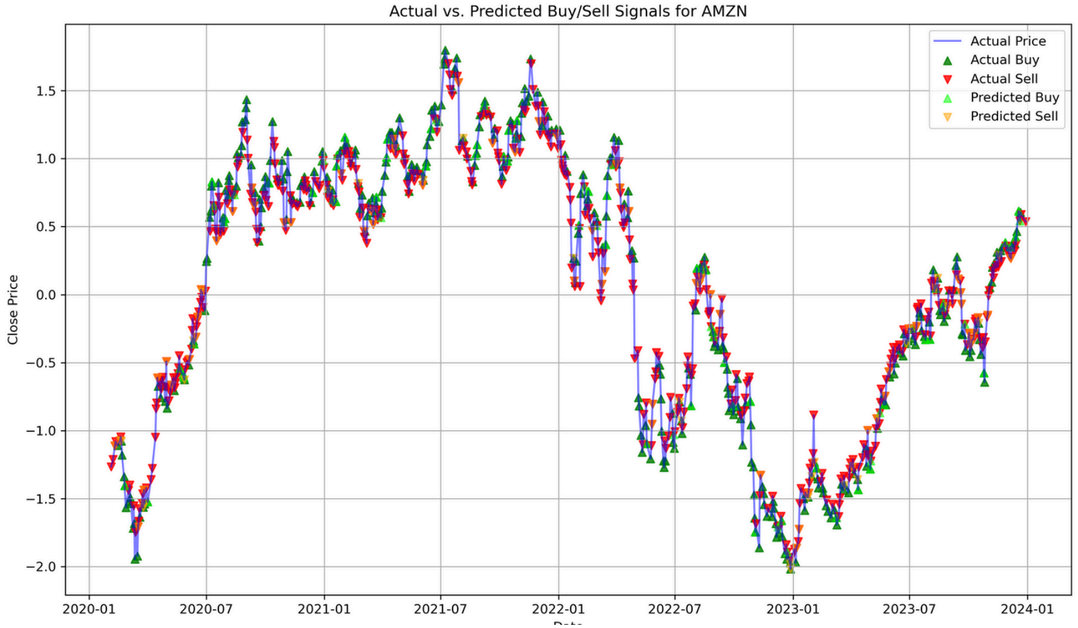


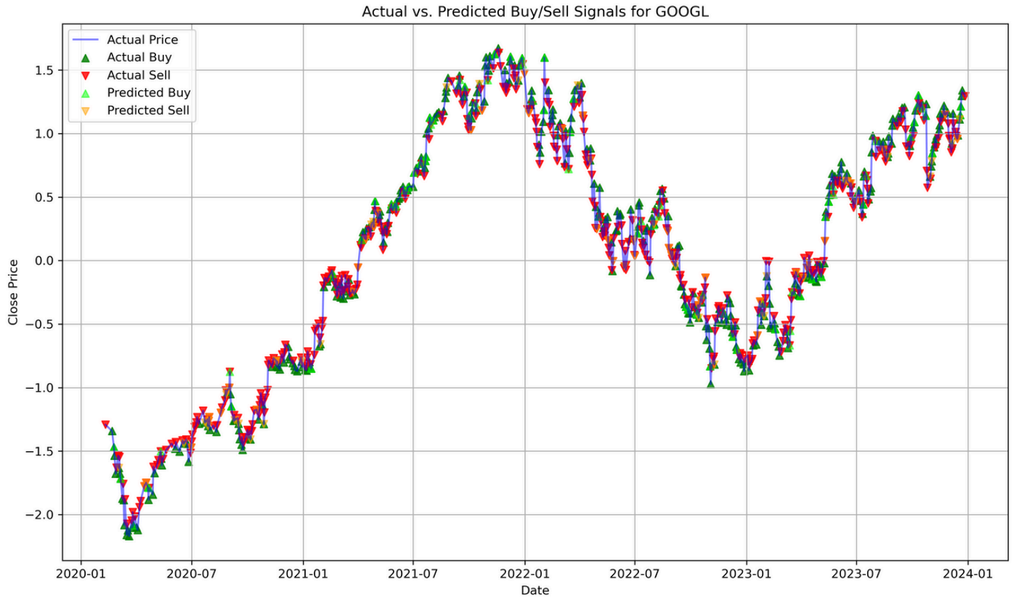
Data Visualization - Random Classifier Model

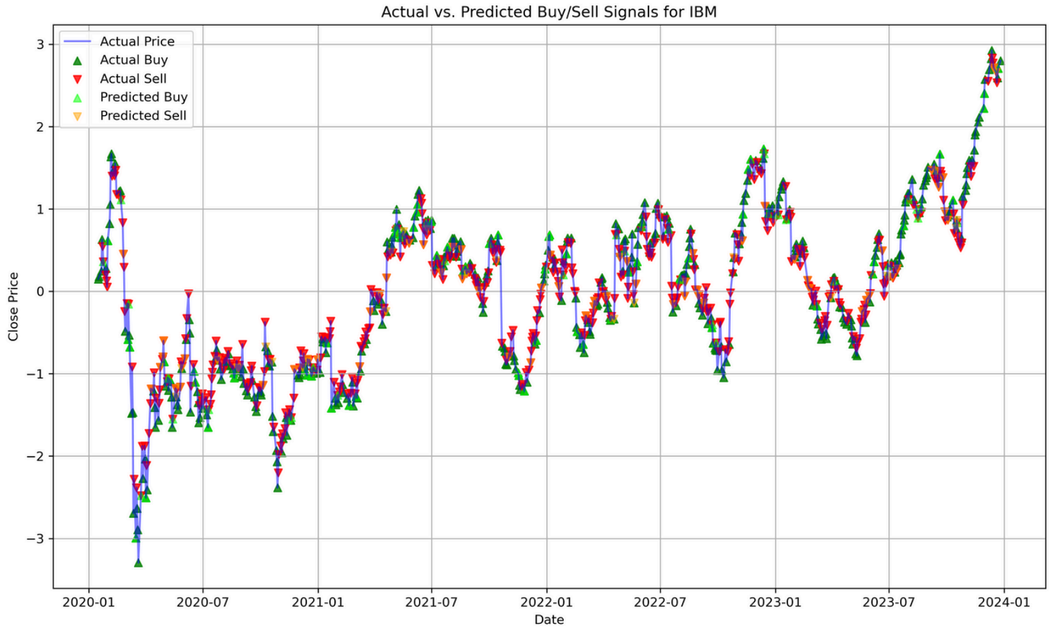
- **Precision, Recall and F1-Score Comparison**

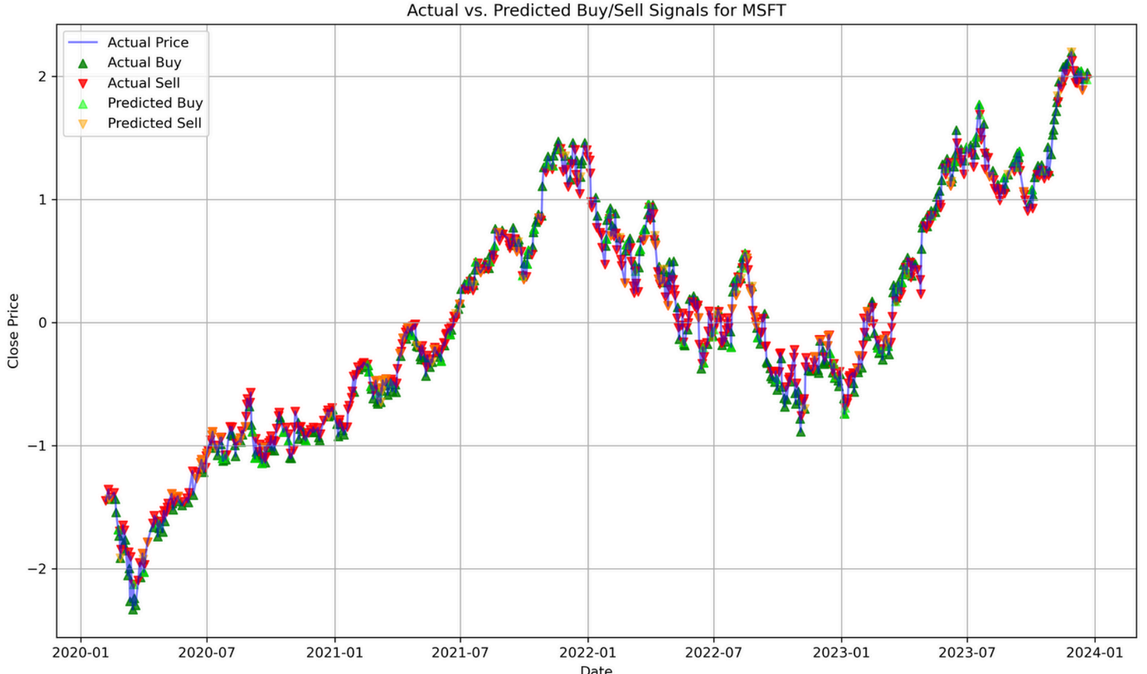


Data Visualization - Random Classifier Model

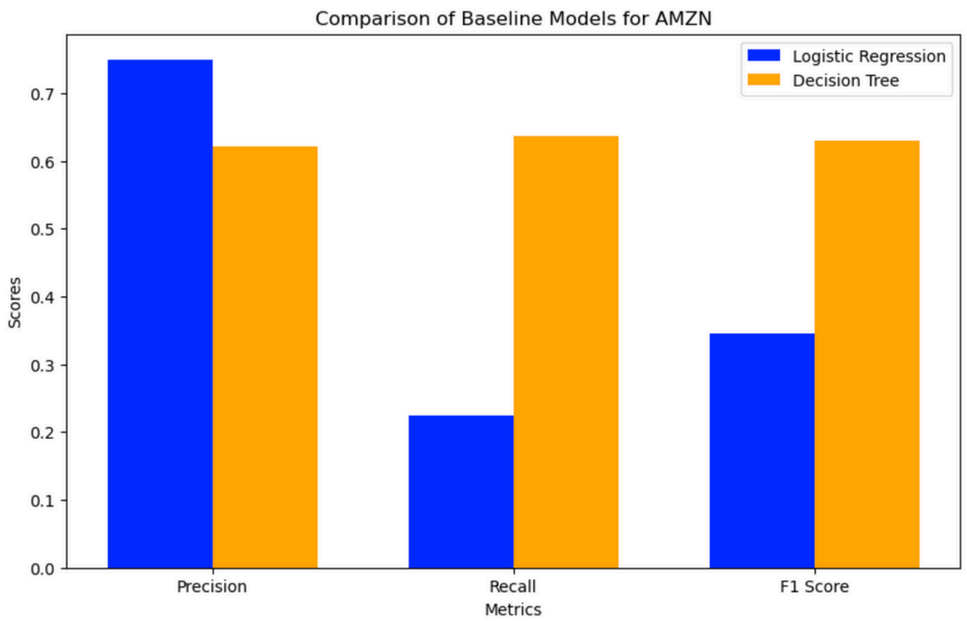


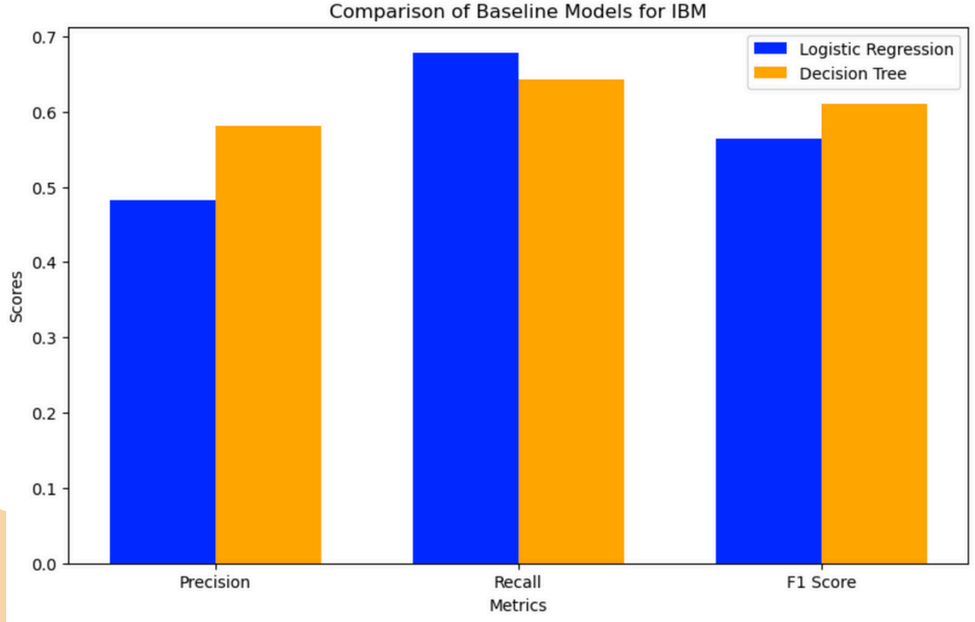


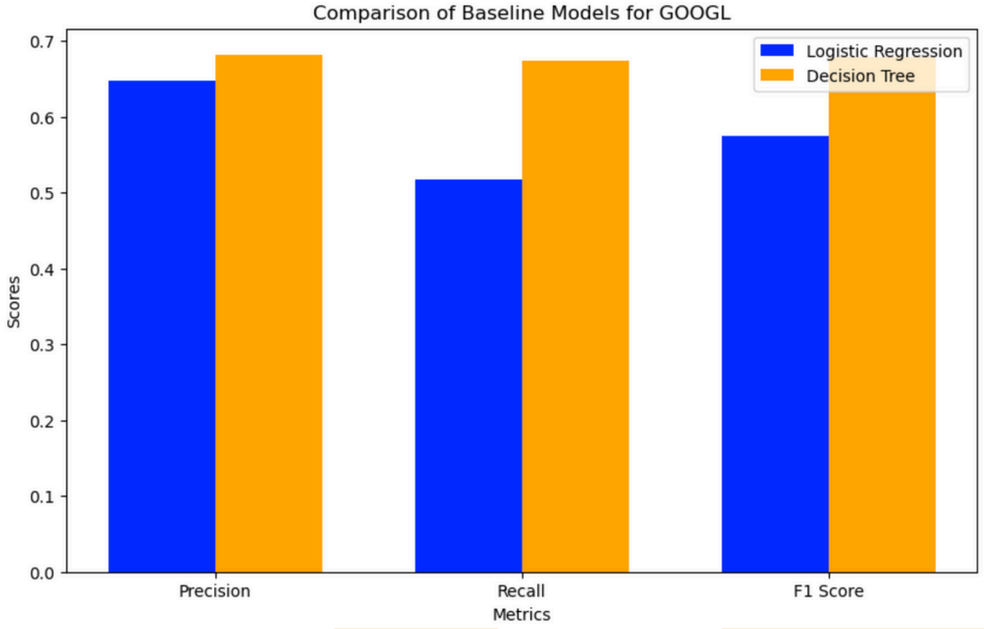


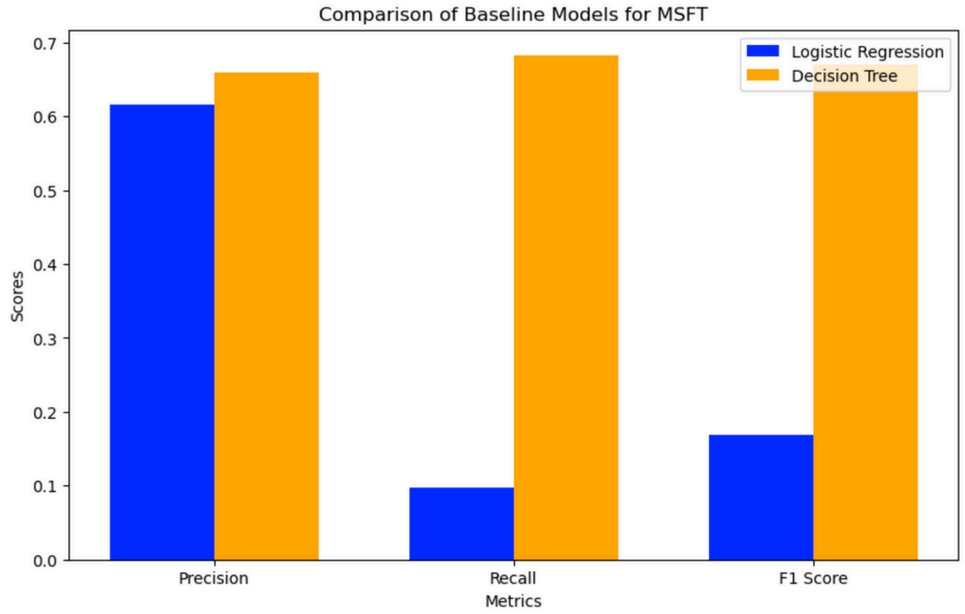


Data Visualization - TensorFlow

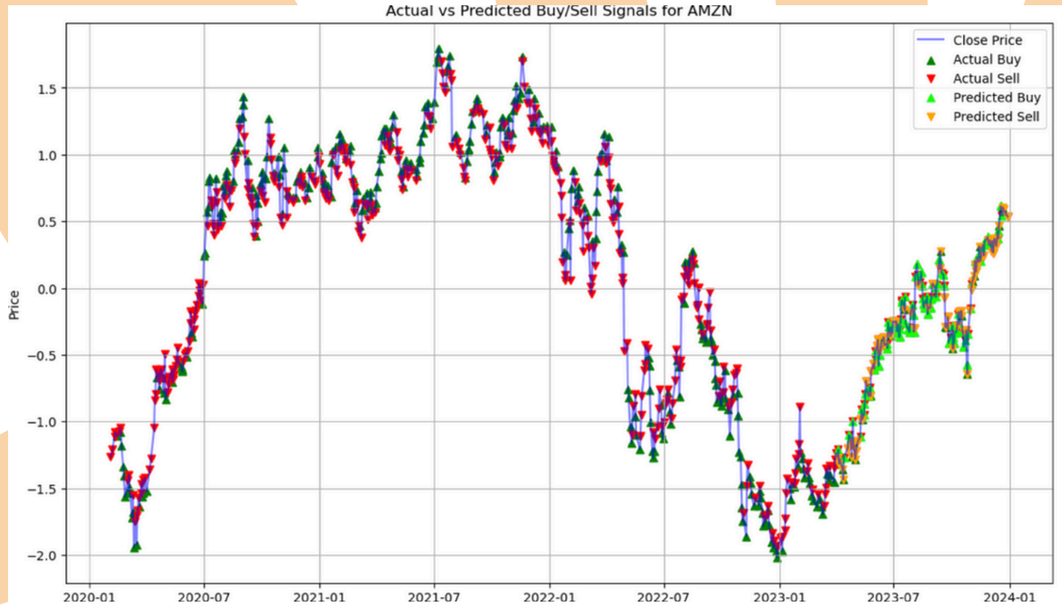


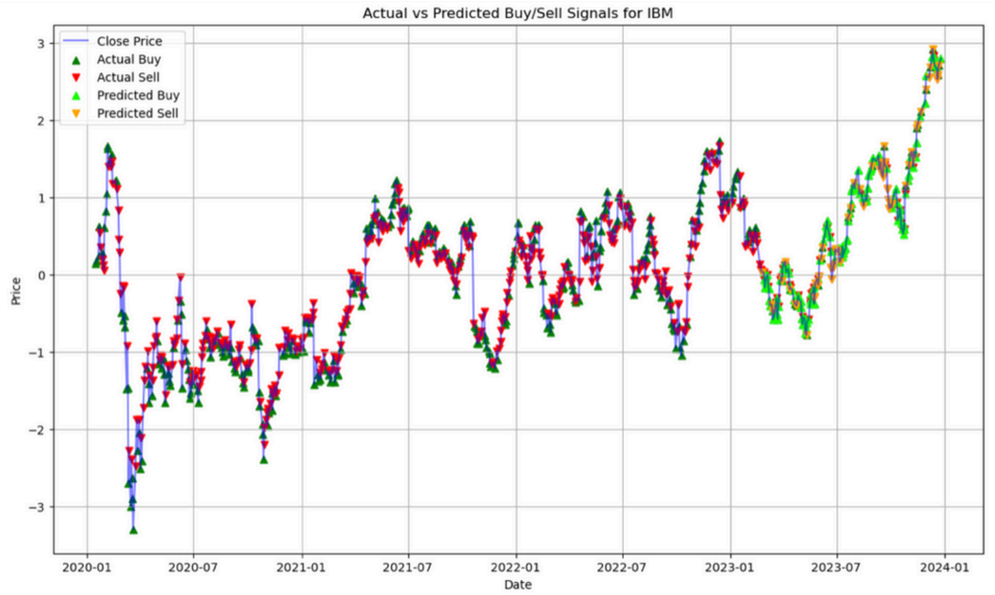


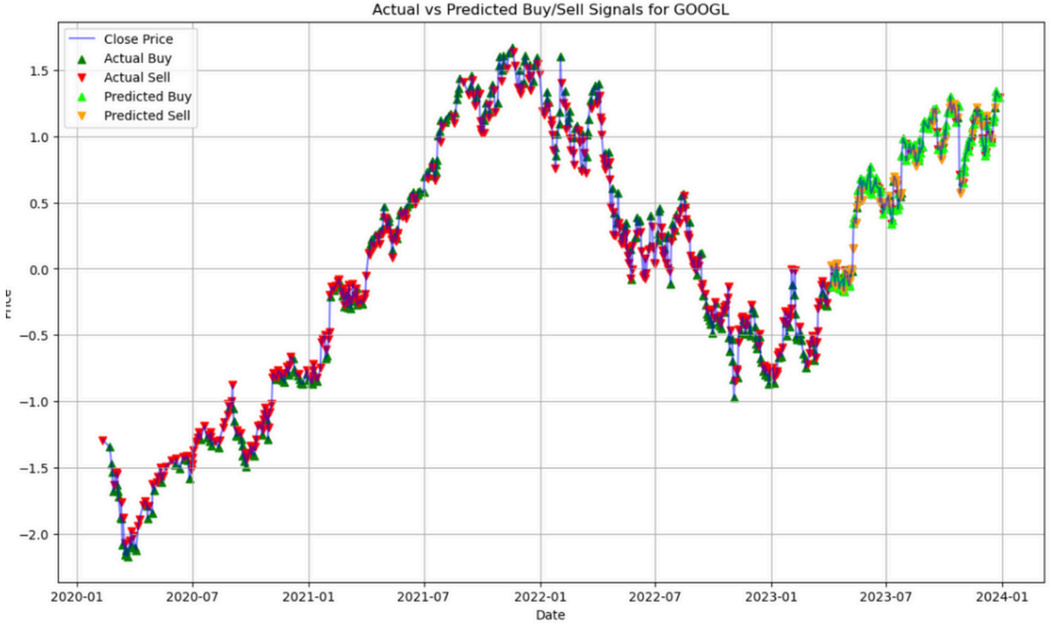


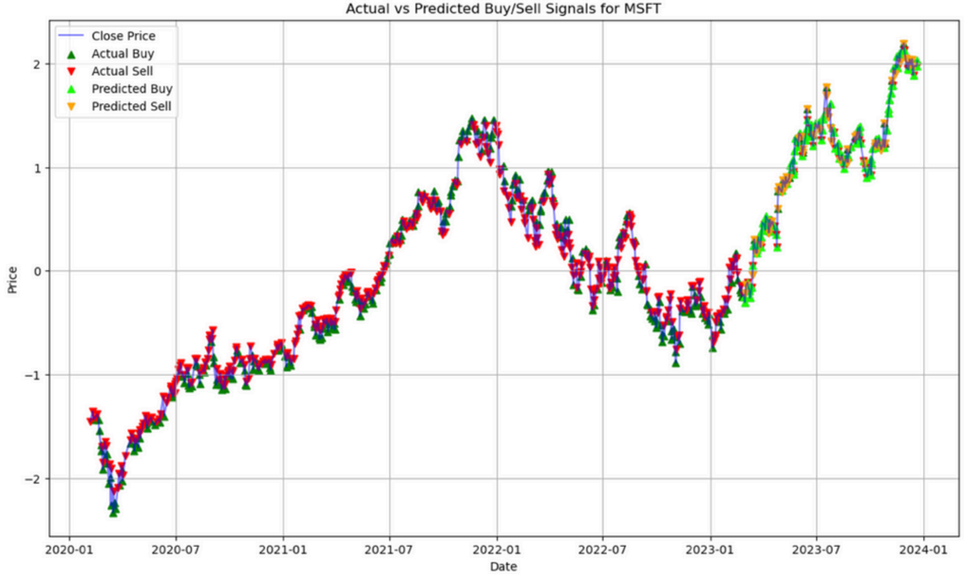


Data Visualization - TensorFlow









Model Improvements

- Hyperparameter tuning to optimize F1-score
- Best parameters identified for RandomForestClassifier
- Updated the thresholds (5%, 8%)

Interactive Application

- Streamlit Application for user interaction
- Buy/Sell decision-making support for users
- show app

Benefits for Users

- Informed decision-making based on statistical analysis
- Time-saving automation for trading decisions
- Adaptability with market trends

Future Enhancements

- Incorporate additional features: news sentiment analysis, economic indicators
- Develop a user-friendly dashboard for visualization
- Implement continuous learning from new data

Conclusion

- **Predictive Power: Machine learning finds hidden patterns**
- **User Empowerment: Users can make data-driven decisions**
- **Shift from instinctive to analytical decision-making**

References

- Data sources (Yahoo Finance)
- Tools used (Python, PostgreSQL, Streamlit, Tensor Flow, Random Forest Classifier, yFinance (Python Library)

Thank You