Stock Price Analysis and Prediction

Using Python for Data Visualization, Modeling, and Insights

Team Members: Shivangi Modi (20093), Vaishnavi Patil (20133), Kashmira Chaudhari (20158)

Table of Contents

Introduction

- This project focuses on analyzing and predicting stock prices using machine learning and data visualization techniques.

Key Objectives:

- Explore the trends in stock prices over time.
- Engineer features such as moving averages.
- Predict stock prices using regression models.
- Provide insights into sector-wise performance.

Related Work

Incorporation of Research Insights:

- Adapted feature engineering techniques from research papers to create moving averages and volume-based metrics.
- Used Random and Decision Tree models and Support Vector Machine based on comparative studies of their performance on stock data.
- Evaluated mutual information and entropy metrics inspired by theoretical studies on feature importance.

Research Papers:

- 'A Comparative Study of Machine Learning Algorithms for Stock Market Prediction'.
- Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques'.
- Stock Market Prediction Using Machine Learning'.

A Comparative Study of Supervised Machine Learning Algorithms for Stock Market Trend Prediction

Dataset Details: Ten years of historical data from Indian stock indices and companies, including Reliance and Infosys.

Features:

Open, high, low, close prices, Volume of stocks traded.

Preprocessing:

- Standardization and normalization were applied to numerical features to ensure uniformity.
- Categorical variables were encoded using one-hot encoding where applicable.
- Derived technical indicators like moving averages, Relative Strength Index (RSI), and Bollinger Bands.

Methods Used:

Decision Trees, Random Forest, Support Vector Machines (SVM), Logistic Regression, k-Nearest Neighbors (k-NN), and Naïve Bayes.

Performance was evaluated using metrics like accuracy, precision, recall, and F1-score.

Visualization:

- Performance metrics (accuracy and F-measure) plotted for each algorithm.
- Timeseries charts illustrating trends identified by models.

What Performed Better and Why:

- Random Forest and SVM performed best in terms of accuracy and robustness.

Useful Insights:

- Random Forest's ensemble approach reduced overfitting and captured complex patterns.

Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques

Dataset Details: 10 years of Stock index data from the S&P 500 and other global indices.

Features:

Open, high, low, close prices of indices, Volume traded.

Preprocessing:

- Handled missing data using interpolation and imputation techniques.
- Removed noisy data using smoothing methods like Exponential Moving Average (EMA).

Methods Used:

- **Comparison of machine learning models:** ANN, SVM, Random Forest, and Naive Bayes.
- Trend Deterministic Data Preparation for converting indicators to categorical trends.

Visualization:

- Correlation heatmaps showing relationships between technical indicators.
- Time-series plots for trends and predicted vs. actual index movements.
- Feature importance plots for tree-based models like XGBoost.

What Performed Better and Why:

- Random Forest showed the highest accuracy when using trend deterministic data, due to its ability to capture nonlinear relationships.

Useful Insights:

- Trend deterministic data significantly improved model accuracy (e.g., ~90% for Random Forest and SVM).

Stock Market Prediction Using Machine Learning

Dataset Details: Yahoo Finance and Google Finance.

Features:

Open, high, low, close prices, Volume and volatility metrics.

Preprocessing: Data normalization, handling of missing values, and alignment of timeseries data for consistency in trends were key preprocessing steps.

Methods Used:

- Sector-wise trend analysis using statistical models.
- Linear Regression, Decision Trees, Random Forest, and Artificial Neural Networks (ANNs).

Visualization:

- Scatter plots showing residuals for regression models.
- Line graphs comparing predicted and actual stock prices.

What Performed Better and Why:

- Random Forest achieved better accuracy compared to other traditional models.

Useful Insights:

- Technology and consumer discretionary sectors showed the highest volatility.

Dataset Overview

Dataset: Historical stock prices of multiple companies over five years.

Columns include: Open, High, Low, Close, Volume, Name, and Date.

Data preprocessing steps include handling missing values, sorting by date, and feature engineering.

- Number of Companies: 505
- Time Range: 2013-2018

Dataset Information

```
df = pd.read_csv('all_stocks_5yr.csv')
print(df.head(100))
```

```
          date   open   high    low  close    volume Name
0   2013-02-08  15.07  15.12  14.63  14.75   8407500  AAL
1   2013-02-11  14.89  15.01  14.26  14.46   8882000  AAL
2   2013-02-12  14.45  14.51  14.10  14.27   8126000  AAL
3   2013-02-13  14.30  14.94  14.25  14.66  10259500  AAL
4   2013-02-14  14.94  14.96  13.16  13.99  31879900  AAL
..         ...    ...    ...    ...    ...       ...  ...
95  2013-06-26  16.50  16.64  16.17  16.17   3604500  AAL
96  2013-06-27  16.29  16.34  16.00  16.31   3566000  AAL
97  2013-06-28  16.24  16.55  16.16  16.42   7063900  AAL
98  2013-07-01  16.50  17.04  16.48  16.80   4666900  AAL
99  2013-07-02  16.78  16.79  16.36  16.43   4009300  AAL

[100 rows x 7 columns]
```

```
# list all Name
df['Name'].unique()
```

```
array(['AAL', 'SLG', 'SLB', 'BLK', 'SJM', 'BLL', 'SIG', 'BMY', 'SHW',
       'SEE', 'BRK.B', 'SCHW', 'BSX', 'SCG', 'BWA', 'SBUX', 'BXP', 'SBAC',
       'RTN', 'CAG', 'RSG', 'CAH', 'RRC', 'CAT', 'ROST', 'ROP', 'BK',
       'SNA', 'BIIB', 'SNI', 'AXP', 'SYMC', 'AYI', 'SYK', 'AZO', 'SWK',
       'A', 'SWKS', 'BAC', 'STZ', 'STX', 'BAX', 'CA', 'STT', 'STI', 'BBT',
       'SRE', 'SRCL', 'BBY', 'SPG', 'BDX', 'SPGI', 'BEN', 'SO', 'BF.B',
       'SNPS', 'BA', 'ROK', 'CBG', 'RMD', 'PRGO', 'CINF', 'PPL', 'CI',
       'PPG', 'CLX', 'PNW', 'PNR', 'CL', 'PNC', 'CMA', 'PM', 'CHTR',
       'CMCSA', 'CME', 'PKI', 'PKG', 'CMG', 'PH', 'CMI', 'PHM', 'CMS',
       'PG', 'PGR', 'CNC', 'PFG', 'PLD', 'SYY', 'PRU', 'CHRW', 'CBOE',
       'RL', 'CBS', 'RJF', 'RHT', 'CB', 'RHI', 'CCI', 'RF', 'CCL', 'RE',
       'CDNS', 'PSA', 'REG', 'CELG', 'RCL', 'CERN', 'QCOM', 'PX', 'CF',
       'PXD', 'PWR', 'CHD', 'PVH', 'CHK', 'PSX', 'REGN', 'CNP', 'TAP',
       'TDG', 'WMB', 'AES', 'WHR', 'AET', 'WFC', 'AFL', 'WEC', 'WDC',
       'AGN', 'WBA', 'AIG', 'WAT', 'AIV', 'V', 'AIZ', 'VZ', 'VTR', 'AJG',
       'VRTX', 'AKAM', 'VRSN', 'ALB', 'VRSK', 'ALGN', 'VNO', 'WMT', 'AEP',
       'WM', 'AEE', 'ZION', 'AAP', 'ZBH', 'ABBV', 'YUM', 'XYL', 'ABC',
       'XRX', 'ABT', 'XRAY', 'ACN', 'XOM', 'VMC', 'ADBE', 'XLNX', 'ADI',
       'XEL', 'ADM', 'XEC', 'ADP', 'WY', 'WYN', 'ADSK', 'WYNN', 'ADS',
       'WU', 'XL', 'ALK', 'VLO', 'VIAB', 'AON', 'TWX', 'AOS', 'TSS',
       'APA', 'TSN', 'TSCO', 'APC', 'TRV', 'APD', 'TROW', 'APH', 'TXN',
       'TRIP', 'ARE', 'TMO', 'ARNC', 'TMK', 'ATVI', 'TJX', 'TIF', 'AVB',
       'TGT', 'AVGO', 'TEL', 'AVY', 'TPR', 'AWK', 'ANTM', 'T', 'ALL',
       'VFC', 'VAR', 'ALXN', 'UTX', 'AMAT', 'USB', 'AMD', 'URI', 'AME',
       'UPS', 'UNP', 'TXT', 'AMGN', 'AMG', 'UNH', 'AMP', 'ULTA', 'UHS',
...
       'DTE', 'MOS', 'DUK', 'MON', 'MNST', 'DVA', 'MMM', 'MRO', 'ZTS',
       'HCN', 'IQV', 'COTY', 'NWS', 'NWSA', 'FOX', 'FOXA', 'ALLE', 'GOOG',
       'NAVI', 'INFO', 'SYF', 'CFG', 'QRVO', 'WRK', 'PYPL', 'KHC', 'HPQ',
       'HPE', 'CSRA', 'WLTW', 'UA', 'FTV', 'EVHC', 'HLT', 'DXC', 'BHGE',
       'BHF', 'DWDP', 'APTV'], dtype=object)
```

Design and Methodology

Why this approach?

- **Identified challenges:** Missing values, complex stock trends, and sector-wide comparisons.
- **Investigated supervised learning models:** Linear Regression, Decision Trees, Random Forest, and Support Vector Machine (SVM).
- Selected models based on performance metrics such as R^2 score.

Feature Engineering:

- Created moving averages (MA10, MA50, MA200).
- Added features like daily return and average volume.

Implementation

Steps Taken:

- **Data Cleaning:** Removed null values and sorted data.
- **Exploratory Data Analysis (EDA):** Visualized stock trends, correlations, and sector performance.
- **Feature Engineering:** Added moving averages and volume-based metrics.
- **Machine Learning Models:** Applied Linear Regression, Random Forest, Decision Trees, and SVM.
- **Visualization:** Used Seaborn and Matplotlib for impactful visualizations.

Entropy and Mutual Information

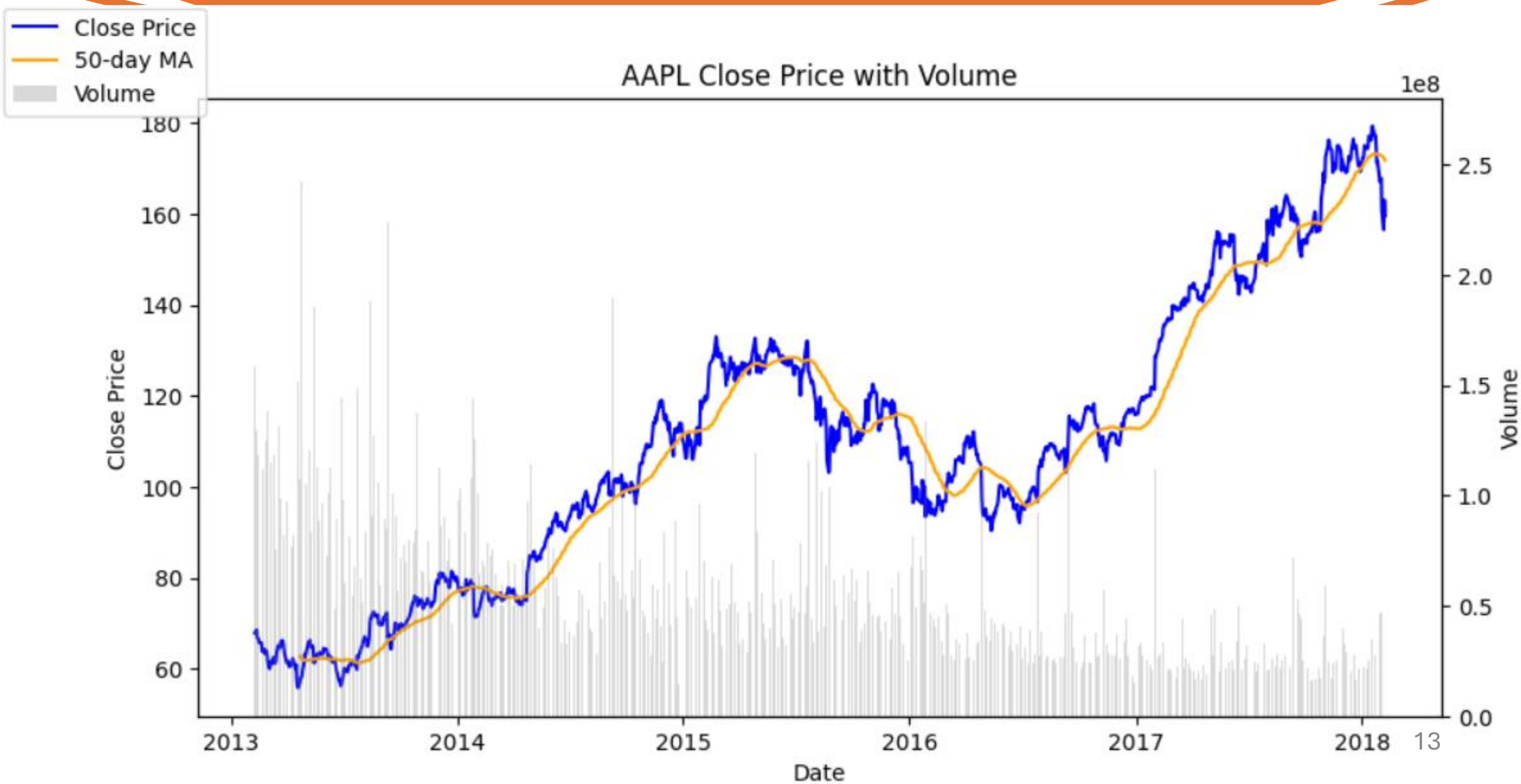
```python
# Compute entropy for each feature
from scipy.stats import entropy
print("Entropy for each feature:")
for col in features.columns:
    ent = entropy(pd.value_counts(features[col].values, normalize=True), base=2)
    print(f"{col}: {ent:.4f}")
```

```
Entropy for each feature:
open: 10.1825
high: 10.1940
low: 10.1904
volume: 10.2981
<ipython-input-15-9c0dba3116cb>:5: FutureWarning: pandas.value_counts is deprecated and will be removed in a future version. Use pd.Series(obj).value_counts() instead.
  ent = entropy(pd.value_counts(features[col].values, normalize=True), base=2)
```

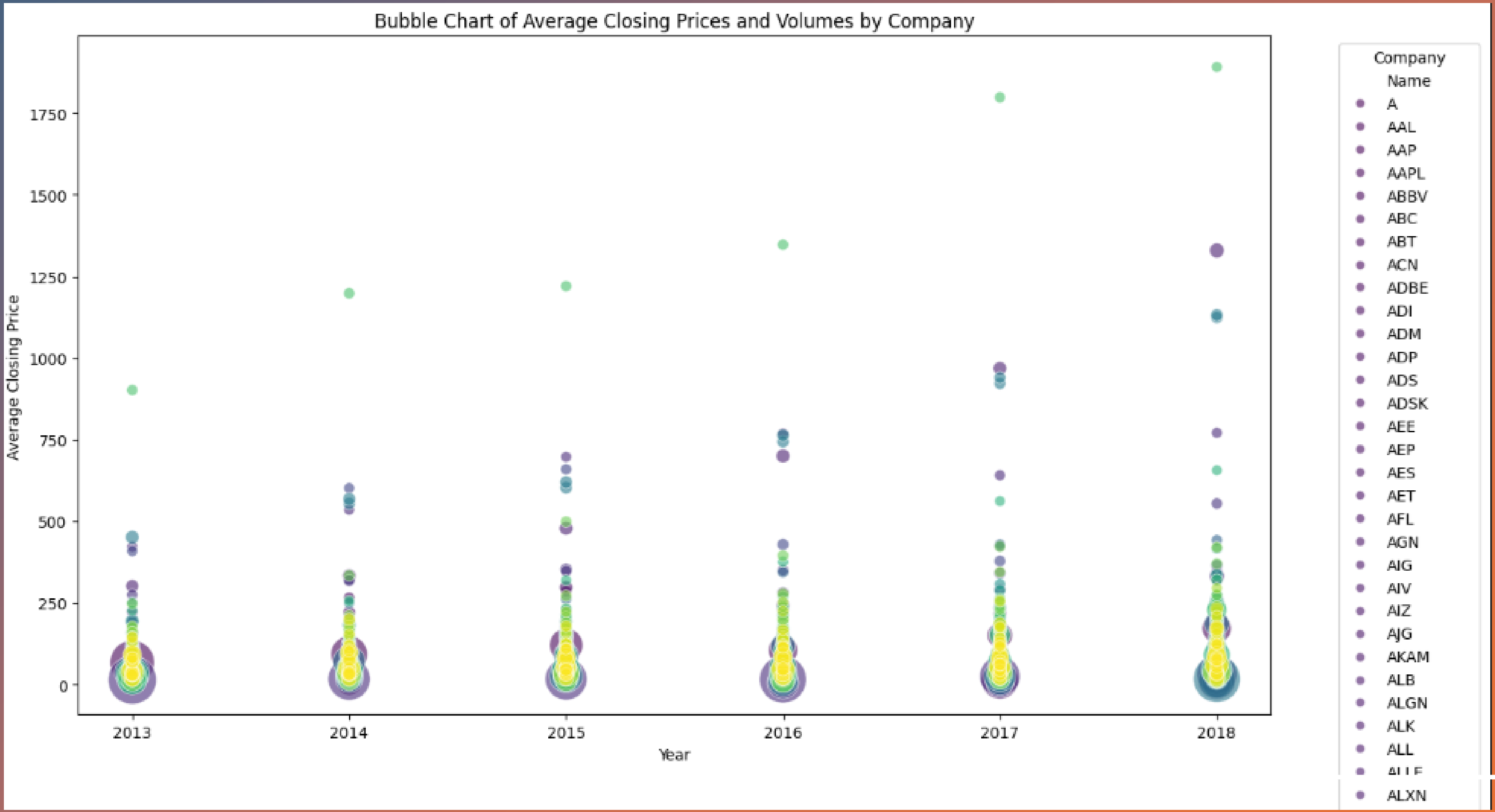
```python
# Compute mutual information scores
mutual_info = mutual_info_regression(X_scaled, target)
mi_scores = pd.Series(mutual_info, index=features.columns)
print("\nMutual Information Scores:")
print(mi_scores.sort_values(ascending=False))
```

```

Mutual Information Scores:
low       3.661837
high      3.624320
open      3.047436
volume    0.456255
dtype: float64
```

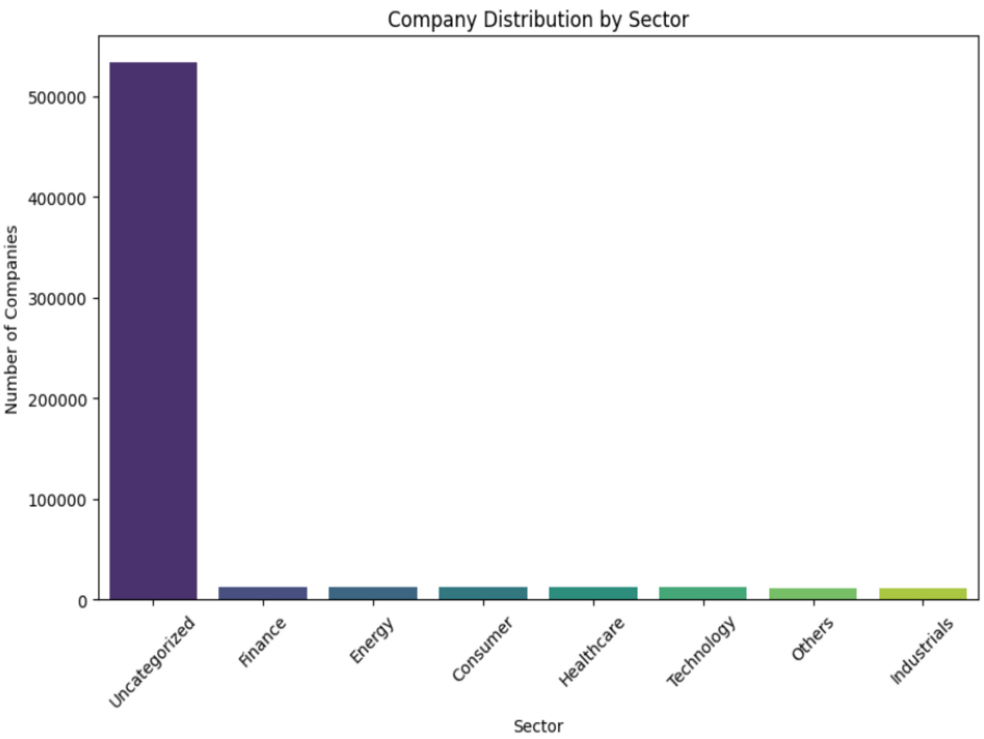
Apple Inc stock price over the years based on volume



Average Closing Prices and Volume by 505 Companies



Sector Wise Mapping




```python
# Map company names into categories
sector_mapping = {
    'Technology': ['AAPL', 'MSFT', 'GOOG', 'FB', 'INTC', 'NVDA', 'CSCO', 'ADBE', 'ORCL', 'IBM'],
    'Healthcare': ['JNJ', 'PFE', 'MRK', 'ABT', 'ABBV', 'BMY', 'LLY', 'AMGN', 'MDT', 'CVS'],
    'Finance': ['JPM', 'BAC', 'GS', 'WFC', 'MS', 'C', 'AXP', 'BLK', 'BK', 'STT'],
    'Consumer': ['KO', 'PEP', 'PG', 'WMT', 'MCD', 'DIS', 'NKE', 'SBUX', 'TGT', 'YUM'],
    'Energy': ['XOM', 'CVX', 'COP', 'SLB', 'HAL', 'KMI', 'PSX', 'EOG', 'MPC', 'PXD'],
    'Industrials': ['BA', 'HON', 'CAT', 'GE', 'UPS', 'MMM', 'RTX', 'LMT', 'DE', 'ITW'],
    'Others': ['ZTS', 'TSN', 'DHR', 'V', 'MA', 'PYPL', 'T', 'VZ', 'CMCSA', 'AMZN']
}
```

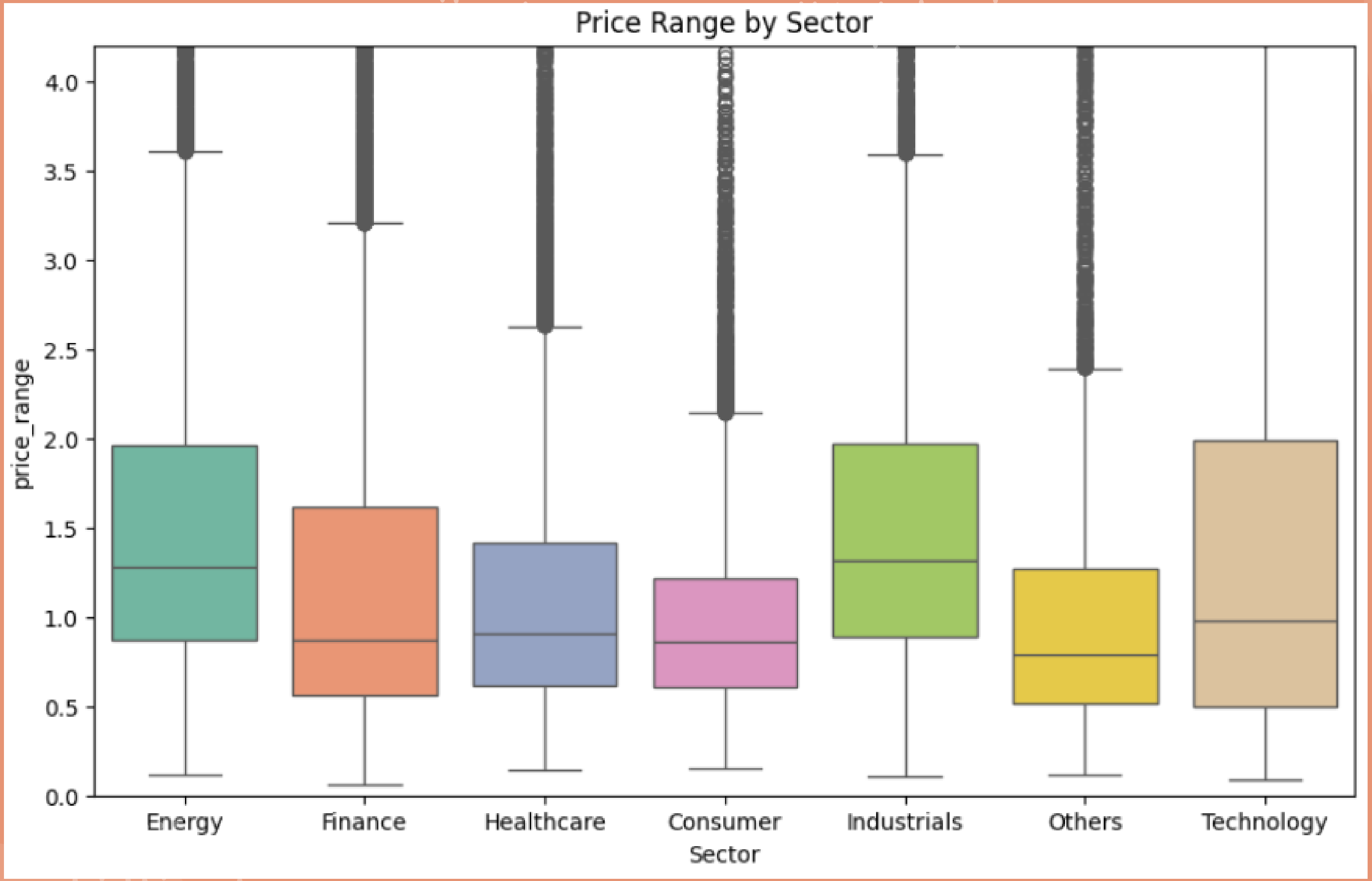
```python
# Function to map company names to sectors
def map_sector(name):
    for sector, companies in sector_mapping.items():
        if name in companies:
            return sector
    return 'Uncategorized'
```

```python
# Apply mapping to dataset
df['Sector'] = df['Name'].apply(map_sector)
```

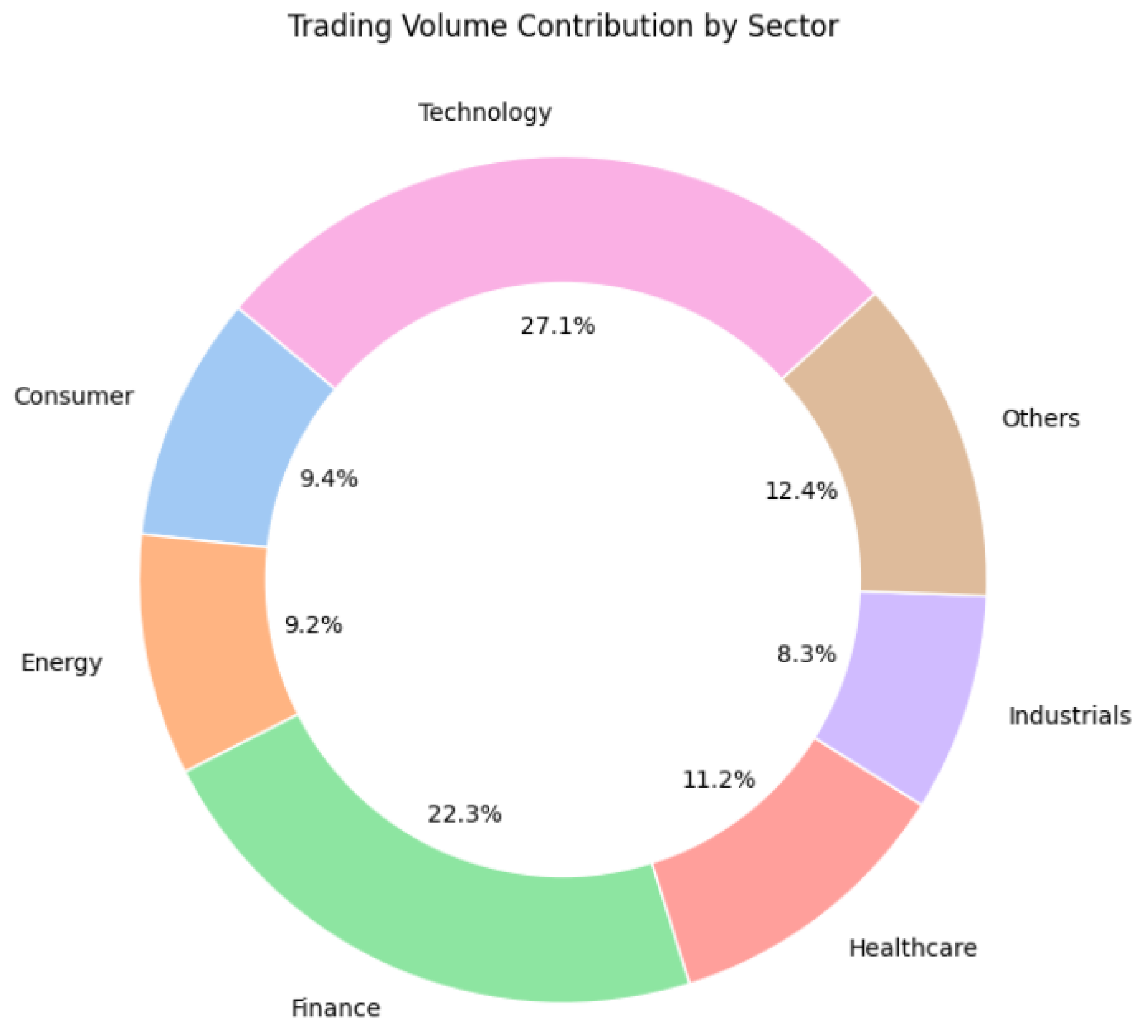
```python
# Validate the mapping
print(df['Sector'].value_counts())
```

```
Sector
Uncategorized    533077
Finance           12590
Energy            12590
Consumer          12590
Healthcare        12588
Technology        12304
Others            11970
Industrials       11331
Name: count, dtype: int64
```

Price Range for Each Sector



Trading Volume Contribution for Each Sector



Key Results:

Linear Regression: R^2 Score = 1
Random Forest: R^2 Score = 1
Decision Tree: R^2 Score = 1
SVM: R^2 Score = 0.99

Visualizations:

Stock price trends for AAPL, GOOG and many more.
Sector-wise performance over time.
Correlation heatmap between stocks, BoxPlot, KDEPlot, BarPlots and Bubble Charts

R^2 Scores for various models

```
# Linear Regression
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)
y_pred_lr = lr_model.predict(X_test)
r2_lr = r2_score(y_test, y_pred_lr)
print(f"Linear Regression R2 Score: {r2_lr:.4f}")
```

```
Linear Regression R2 Score: 1.0000
```

```
# Random Forest
rf_model = RandomForestRegressor(random_state=42, n_estimators=100)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)
r2_rf = r2_score(y_test, y_pred_rf)
print(f"Random Forest R2 Score: {r2_rf:.4f}")
```

```
Random Forest R2 Score: 1.0000
```

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor

# Decision Tree
dt_model = DecisionTreeRegressor(random_state=42)
dt_model.fit(X_train, y_train)
y_pred_dt = dt_model.predict(X_test)
r2_dt = r2_score(y_test, y_pred_dt)
print(f"Decision Tree R2 Score: {r2_dt:.4f}")
```

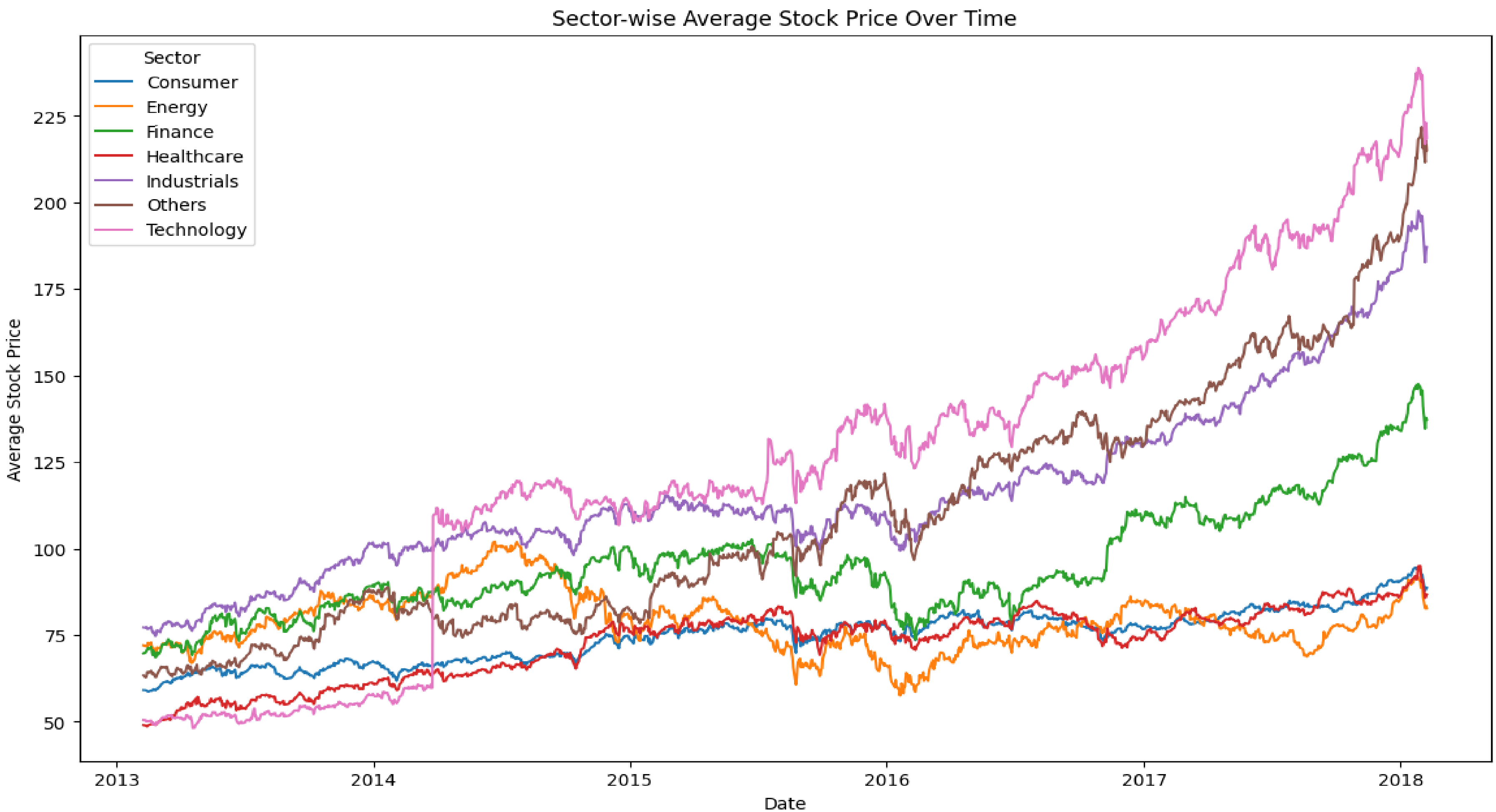
```
Decision Tree R2 Score: 1.0000
```

```
# Parameter grid for SVM
param_grid = {
    'C': [0.1, 1, 10],
    'gamma': [0.01, 0.1, 1],
    'kernel': ['linear', 'rbf']
}
```

```
# GridSearchCV to find best parameters
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVR
X_train, X_test, y_train, y_test = train_test_split(X_scaled, target, test_size=0.2, random_state=42)
grid_search = GridSearchCV(SVR(), param_grid, cv=5, scoring='r2')
grid_search.fit(X_train, y_train)

print("Best SVM Parameters:", grid_search.best_params_)

# Predict on the test data
y_pred = grid_search.predict(X_test)

# Calculate R² score
r2 = r2_score(y_test, y_pred)
print(f"R² Score: {r2:.4f}")
```

```
Best SVM Parameters: {'C': 10, 'gamma': 0.01, 'kernel': 'linear'}
R² Score: 0.9997
```

Random Forest took 11 minutes to train while Decision Tree took just 10 seconds



Performance of Sector-wise Average Stock Price Over Time

Enhancement Ideas

INCORPORATE DEEP LEARNING MODELS SUCH AS LSTMS FOR TIME-SERIES FORECASTING.

EXTEND ANALYSIS TO GLOBAL STOCK MARKETS.

AUTOMATE DATA UPDATES WITH APIS FOR REAL-TIME PREDICTIONS.

EXPLORE SENTIMENT ANALYSIS FROM FINANCIAL NEWS TO ENHANCE PREDICTIONS.

Conclusion

This project demonstrates the power of machine learning in financial data analysis:

- Successfully applied regression models to predict stock prices.
- Provided actionable insights on sector performance and market trends.
- Highlights the importance of feature engineering and data visualization.

References

- A comparative study of supervised machine learning algorithms for stock market trend prediction
- Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques
- Stock Market Prediction Using Machine Learning

THANK YOU