Final Report Capstone Project: Tesla Stock Price Forecasting

1. Problem Statement

In the highly competitive finance and trading industry, accurate stock price forecasting is essential for optimizing investment strategies and managing risk. Tesla's stock, characterized by its volatility and responsiveness to market dynamics, poses a challenge for financial institutions seeking to make short-term, data-driven investment decisions. Trading firms require robust predictive models to anticipate stock price movements and capitalize on profit opportunities. This project addresses this need by analyzing two years of historical stock data, along with correlated stocks and economic indicators, to forecast Tesla's stock price over a six-month period. The model's accuracy will be rigorously tested, with results providing actionable insights that can support trading strategies.

The approach for this project is designed to forecast Tesla's stock price by leveraging a combination of correlated stock data and key economic indicators. The methodology consists of several stages, including Data Collection, Exploratory Data Analysis, Preprocessing, Modelling and Evaluation, each aiming to refine the predictive model and deliver accurate forecasts.

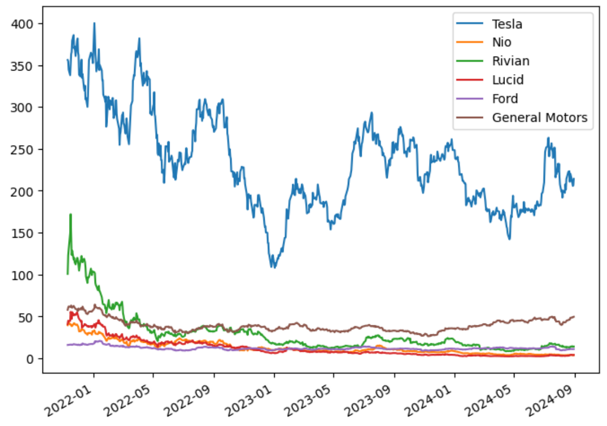
2. Data Collection

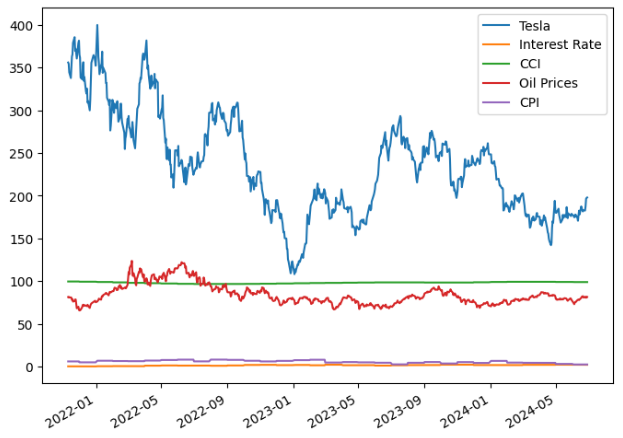
Historical Tesla stock data and correlated stocks (Nio, Rivian, Lucid, Ford, GM) were obtained using the Yahoo Finance API via the yfinance library.

Economic indicators were sourced from various platforms:

- Interest Rate and Inflation (CPI) from FRED.
- Consumer Confidence Index from Investing.com.
- Oil Prices from Macrotrends.

The initial data cleaning involved aligning the indexes of all datasets. Economic indicators, originally recorded monthly, were interpolated to daily values. Additionally, the data was sliced to ensure consistent start and end dates across all time series. After aligning all the time series, I plotted them to obtain an initial overview of the project data:

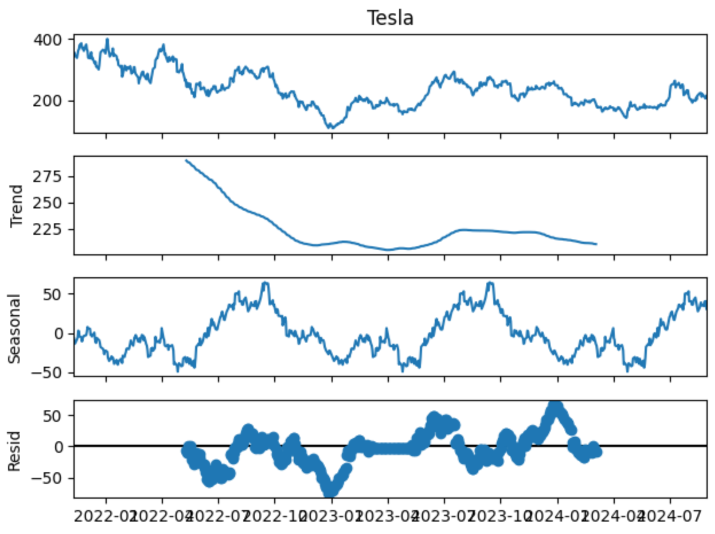




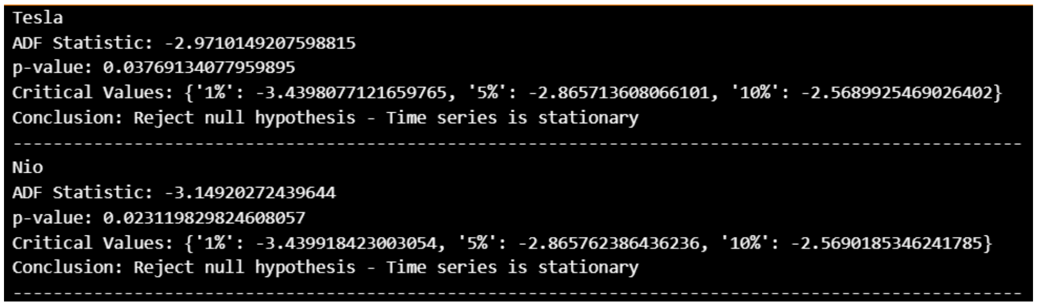
3. Exploratory Data Analysis (EDA)

In the EDA phase, the following analyses were performed:

- Plotted the Components (Trend, Seasonal, and Residual) for Each Time Series: This step involved decomposing each time series into its constituent components: trend (long-term movement), seasonal (regular, repeating patterns), and residual (random noise). Plotting these components helps in understanding the underlying patterns and periodic behaviors in the data.

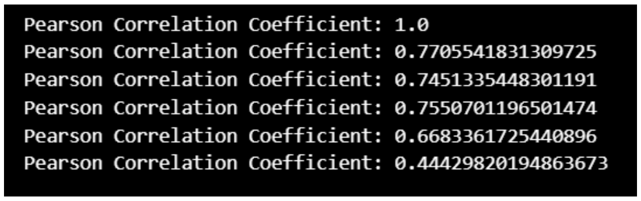


- Tested for Stationarity Using the Augmented Dickey-Fuller Test: The ADF Test was applied to assess whether each time series is stationary. A stationary time series has constant mean and variance over time, which is a crucial assumption for many time series models. The test helps determine if differencing or transformation is needed to stabilize the mean.



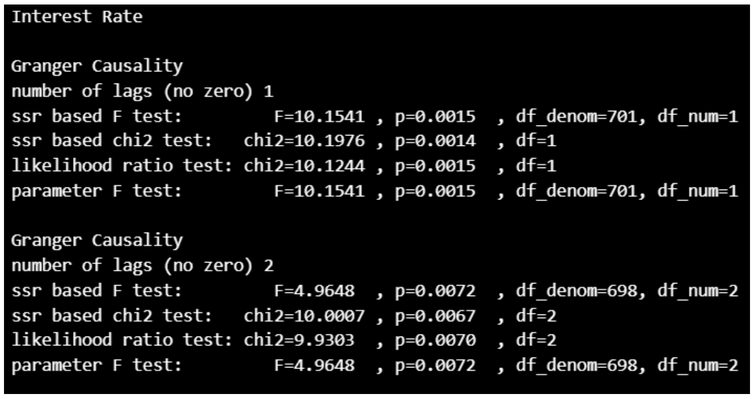
```
Tesla
ADF Statistic: -2.9710149207598815
p-value: 0.03769134077959895
Critical Values: {'1%': -3.4398077121659765, '5%': -2.865713608066101, '10%': -2.5689925469026402}
Conclusion: Reject null hypothesis - Time series is stationary
-----------------------------------------------------------------------------------------------
Nio
ADF Statistic: -3.14920272439644
p-value: 0.023119829824608057
Critical Values: {'1%': -3.439918423003054, '5%': -2.865762386436236, '10%': -2.5690185346241785}
Conclusion: Reject null hypothesis - Time series is stationary
-----------------------------------------------------------------------------------------------
```

- Checked for Correlation Between Tesla and All Other Time Series: Correlation analysis was performed to evaluate the strength and direction of the relationship between Tesla's stock price and other time series (correlated stocks and economic indicators). This helps in identifying which variables have a significant linear relationship with Tesla's stock price.



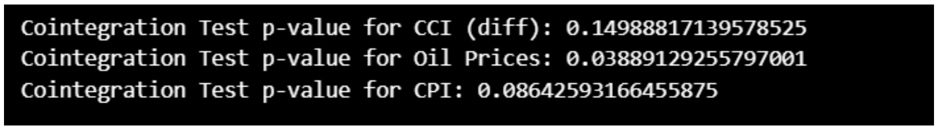
```
Pearson Correlation Coefficient: 1.0
Pearson Correlation Coefficient: 0.7705541831309725
Pearson Correlation Coefficient: 0.7451335448301191
Pearson Correlation Coefficient: 0.7550701196501474
Pearson Correlation Coefficient: 0.6683361725440896
Pearson Correlation Coefficient: 0.44429820194863673
```

- Checked for Granger Causality Between Tesla and All Other Time Series: The Granger Causality Test was used to determine if past values of one time series can predict future values of Tesla's stock price. This test helps in identifying whether changes in one time series provide useful information for forecasting Tesla's stock price.



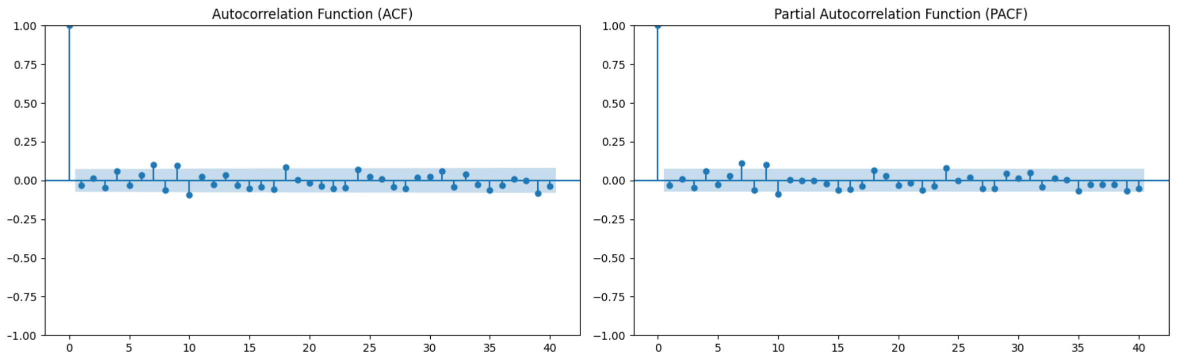
```
Interest Rate

Granger Causality
number of lags (no zero) 1
ssr based F test:         F=10.1541 , p=0.0015  , df_denom=701, df_num=1
ssr based chi2 test:   chi2=10.1976 , p=0.0014  , df=1
likelihood ratio test: chi2=10.1244 , p=0.0015  , df=1
parameter F test:         F=10.1541 , p=0.0015  , df_denom=701, df_num=1

Granger Causality
number of lags (no zero) 2
ssr based F test:         F=4.9648  , p=0.0072  , df_denom=698, df_num=2
ssr based chi2 test:   chi2=10.0007 , p=0.0067  , df=2
likelihood ratio test: chi2=9.9303  , p=0.0070  , df=2
parameter F test:         F=4.9648  , p=0.0072  , df_denom=698, df_num=2
```

- Checked for Cointegration Between Tesla and All Other Time Series: Cointegration analysis was conducted to assess if there is a long-term equilibrium relationship between Tesla's stock price and other time series. Cointegrated series move together over time, which can be useful for building models that account for long-term relationships.



```
Cointegration Test p-value for CCI (diff): 0.14988817139578525
Cointegration Test p-value for Oil Prices: 0.03889129255797001
Cointegration Test p-value for CPI: 0.08642593166455875
```

- Plotted Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) for All Time Series: The ACF and PACF plots were used to visualize the correlation of a time series with its own past values. The ACF shows the correlation of the series with lags of itself, while the PACF shows the correlation after removing the effect of intermediate lags. These plots help in identifying appropriate lag lengths for time series models.



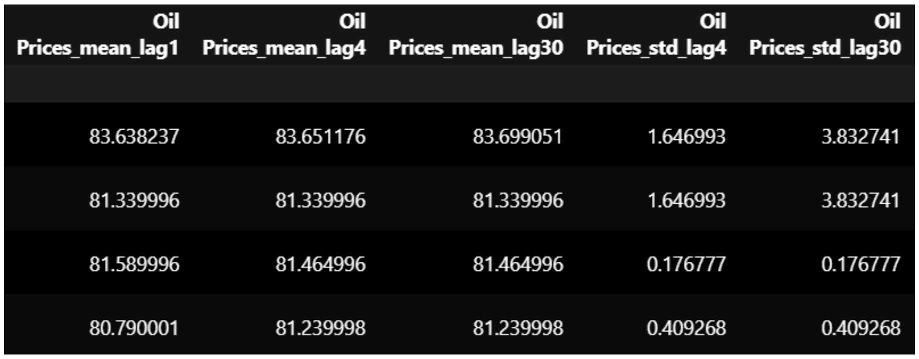
The data collection and EDA code are available in the Jupyter Notebook 01_Wrangling_and_EDA located in the notebooks folder. Following these steps, the processed data was saved for use in subsequent analysis in the next notebook.

4. Preprocessing

The following steps were performed during the preprocessing phase:

- Missing Values: Checked for missing values in the dataset, and none were found.

- Feature Engineering: Extracted date-related features such as month, day, weekday, and year from the date index. Created lag features by calculating the rolling mean for lagged windows of sizes 1, 4, and 30 days, as well as the rolling standard deviation for windows of 4 and 30 days for each time series. Window sizes were chosen based on meaningful time intervals: a size of 1 represents the previous day's data, a size of 4 captures the current week's data, and a size of 30 reflects monthly trends.



Oil Prices_mean_lag1	Oil Prices_mean_lag4	Oil Prices_mean_lag30	Oil Prices_std_lag4	Oil Prices_std_lag30
83.638237	83.651176	83.699051	1.646993	3.832741
81.339996	81.339996	81.339996	1.646993	3.832741
81.589996	81.464996	81.464996	0.176777	0.176777
80.790001	81.239998	81.239998	0.409268	0.409268

- Set Time Series Frequency: The time series dataframe frequency was set to daily to ensure compatibility with ARIMA modeling.
- Train-Test Split: The data was split into training and test sets, with the first 80% of the data used for training and the last 20% reserved for testing

5. Modelling

The first model implemented was ARIMA, applied to Tesla's time series data. I utilized the auto_arima function from the pmdarima library to automatically select the best ARIMA models based on AIC and BIC. The selected models were then refined using statsmodels to calculate evaluation metrics.

The calculated metrics for each model included AIC, BIC, RMSE, and MAE. These metrics were computed using time series cross-validation with three folds:

- Fold 1: Training set of 205 data points, validation set of 205 data points.
- Fold 2: Training set of 410 data points, validation set of 205 data points.
- Fold 3: Training set of 615 data points, validation set of 205 data points.

For each fold, the four metrics were computed, and the average was taken to assess model performance. The top four ARIMA models identified by auto_arima were:

- ARIMA (0,1,0)
- ARIMA (1,1,0)
- ARIMA (0,1,1)
- ARIMA (1,1,1)

The second phase of modeling involved extending the four ARIMA models into ARIMAX models by incorporating exogenous variables. The exogenous variables included all the features created during the feature engineering phase, such as the date-related features and lagged statistics (rolling means and standard deviations).

Similar to the ARIMA models, I instantiated the same four models:

- ARIMAX (0,1,0)
- ARIMAX (1,1,0)
- ARIMAX (0,1,1)
- ARIMAX (1,1,1)

For each ARIMAX model, I calculated the same four evaluation metrics (AIC, BIC, RMSE, and MAE) using time series cross-validation, as done in the ARIMA phase. The metrics were averaged across the folds to assess the performance of each model.

The next step in the modeling process was to implement machine learning models. I instantiated three linear models from the scikit-learn library:

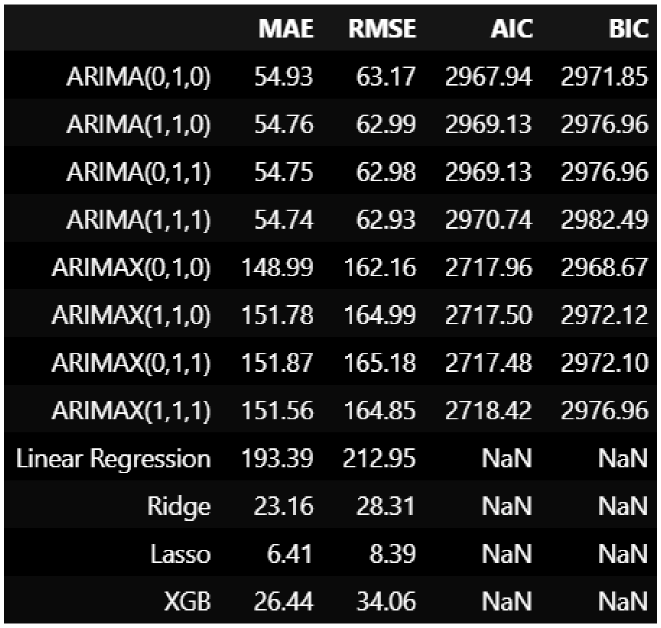
- Linear Regression
- Lasso Regression
- Ridge Regression

Additionally, I included an ensemble method, the **XGB Regressor**, from the xgboost library to evaluate the potential improvement offered by a tree-based approach.

The same time series cross-validation strategy was applied as with the ARIMA and ARIMAX models, using the same fold structure. For these models, only RMSE and MAE were calculated, as AIC and BIC are specific to ARIMA models.

The results from the cross-validation provided a comparison of the performance between linear models and the ensemble method against the statistical approaches.

Below is a comparison of the performance metrics for all models, including statistical models (ARIMA and ARIMAX), machine learning models (Linear, Lasso, Ridge), and the ensemble method (XGB Regressor). ARIMA models are effective for capturing linear trends, but machine learning models such as Lasso may capture more complex relationships in the data.



	MAE	RMSE	AIC	BIC
ARIMA(0,1,0)	54.93	63.17	2967.94	2971.85
ARIMA(1,1,0)	54.76	62.99	2969.13	2976.96
ARIMA(0,1,1)	54.75	62.98	2969.13	2976.96
ARIMA(1,1,1)	54.74	62.93	2970.74	2982.49
ARIMAX(0,1,0)	148.99	162.16	2717.96	2968.67
ARIMAX(1,1,0)	151.78	164.99	2717.50	2972.12
ARIMAX(0,1,1)	151.87	165.18	2717.48	2972.10
ARIMAX(1,1,1)	151.56	164.85	2718.42	2976.96
Linear Regression	193.39	212.95	NaN	NaN
Ridge	23.16	28.31	NaN	NaN
Lasso	6.41	8.39	NaN	NaN
XGB	26.44	34.06	NaN	NaN

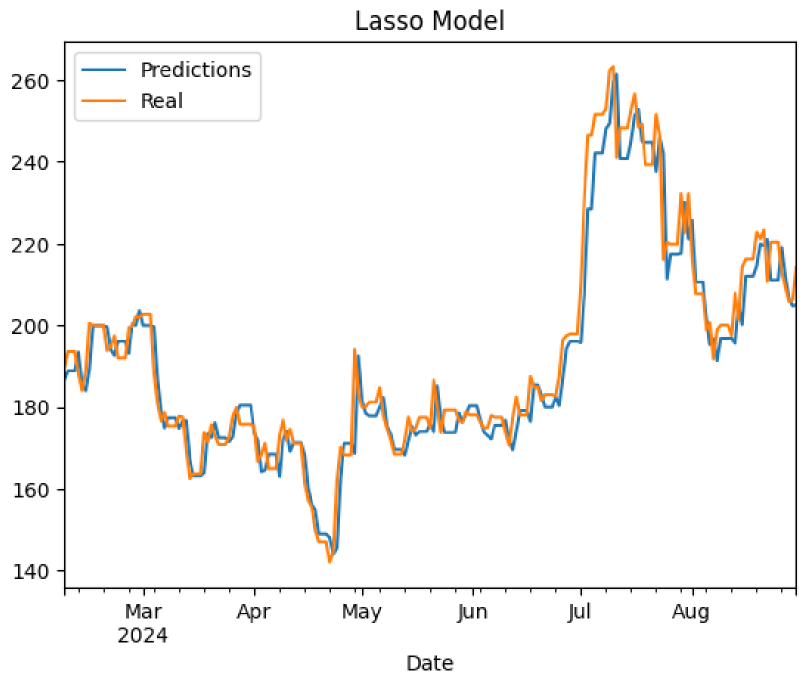
6. Evaluation

For the evaluation phase, I selected the best models based on their performance metrics. The ARIMA (1,1,1) model was chosen as the best statistical model, as it provided a good balance across the metrics. For machine learning models, Lasso Regression performed the best.

To further improve the forecast, I attempted to create a hybrid model that combined the predictions from both the ARIMA (1,1,1) and Lasso models. I developed a function to optimize the weights assigned to each model's forecast, aiming to minimize the error against the test set identified during the preprocessing stage.

Interestingly, the result showed that the optimal weight was 1 (100%) for the Lasso model and 0 (0%) for ARIMA (1,1,1), meaning the Lasso model alone provided the best forecast, with no contribution from ARIMA.

As a result, Lasso Regression was selected as the final model for predicting Tesla's stock price. Lasso performed the best due to its ability to eliminate variables with low predictive power, which helps reduce overfitting and improve model generalization.



The Preprocessing, Modelling, and Evaluation code are available in the Jupyter Notebook 02_Preprocessing_and_Model located in the notebooks folder.

7. Conclusion

In this project, we explored various methods for forecasting Tesla's stock price using a combination of statistical models (ARIMA and ARIMAX), machine learning models (Linear, Lasso, Ridge), and an ensemble method (XGB Regressor). After thorough evaluation using time series cross-validation, Lasso Regression emerged as the most effective model, outperforming both the ARIMA-based models and the hybrid approach.

Despite experimenting with hybrid models, the Lasso model alone provided the most accurate predictions, as confirmed by its optimized performance and alignment with the test set.

This result highlights the strength of machine learning approaches, particularly Lasso Regression, for forecasting complex time series data in financial markets. Future work could explore additional features, alternative machine learning techniques, or even incorporating external sentiment analysis to further refine predictions.

8. Recommendations

To further enhance the accuracy and robustness of the stock price forecasting model, I suggest the following actions:

- Incorporate Alternative Data Sources: While the current model leverages economic indicators and correlated stock data, incorporating external datasets such as social media sentiment (e.g., Twitter or news sentiment) or even macroeconomic forecasts could provide additional predictive power.

- Explore Advanced Machine Learning Models: While Lasso Regression performed well, experimenting with more advanced models such as Long Short-Term Memory (LSTM) networks or other deep learning architectures designed for time series could potentially capture more complex patterns in stock price movements.

- Feature Engineering and Selection: Further refinement of the feature set, including testing additional lagged features or other time-based signals, might improve the model's performance. Additionally, using feature selection techniques could help in identifying the most influential variables.

- Expand Training Data: A longer historical time frame could provide the model with more training data and thus capture broader trends. Including data from multiple economic cycles might also improve the model’s ability to handle future uncertainties.

- Regular Model Retraining: Given the volatility of stock markets, it’s crucial to retrain the model regularly with the most up-to-date data. Markets evolve, and the model needs to be updated frequently to maintain its predictive accuracy.

- Risk Management: For trading and finance companies, this model can be integrated into broader risk management strategies. However, it is essential to complement this model with other qualitative factors, such as geopolitical events or sudden market shocks, which may not be fully captured by the model.

- Hybrid Models with Other Asset Classes: Expanding the scope to forecast other asset classes like commodities or bonds in relation to Tesla stock could provide more holistic trading strategies. This could also diversify risk and provide better insights into interdependencies between various financial assets.