TESLA STOCK ANALYSIS

PRESENTATION BY:

YAMINI MANRAL

Introduction

- Analyzing Tesla's stock using machine learning can involve the development of predictive models that aim to forecast price movements, identify optimal entry and exit points, and assess risk factors. Additionally, sentiment analysis of news articles, social media, and market reports can provide valuable information on public perception and potential market sentiment shifts.
- In this era of big data, machine learning equips investors with advanced tools to navigate the dynamic landscape of financial markets. The integration of predictive modeling, natural language processing, and data-driven analytics can contribute to a more comprehensive understanding of Tesla's stock behavior, offering investors a competitive edge in decision-making.

Objectives

- Utilize historical stock price and relevant financial data to build machine learning models that can predict future stock prices.
- Develop models to analyze and predict the volatility of Tesla stock. Understanding stock volatility can be valuable for risk management and investment strategy.
- Build anomaly detection models to identify unusual patterns or outliers in Tesla stock data. This can be useful for detecting potential irregularities or market disruptions.
- Compare Tesla's stock performance with that of its competitors in the automotive and technology sectors. Machine learning models can help identify patterns and trends that differentiate Tesla from its peers, providing a comprehensive view of its competitive position in the market.
- Develop models to predict the long-term performance of Tesla stock, considering factors such as industry trends, technological advancements, and global economic conditions.

About

- Tesla stock has been known for its high volatility, with significant price fluctuations over time. The stock's performance has been influenced by factors such as production numbers, delivery figures, financial results, and market sentiment.
- Tesla has experienced substantial growth in market capitalization, becoming one of the most valuable automakers globally. The company's ambitious plans for electric vehicles (EVs) and renewable energy have contributed to its perceived long-term potential.
- Tesla's financial performance has varied, with periods of losses and profits. Investors closely monitor the company's earnings reports, production numbers, and gross margins to assess its financial health and sustainability.
- Tesla is often associated with innovation in electric vehicle technology and autonomous driving. Announcements about new products, advancements in battery technology, and updates on the development of self-driving capabilities have impacted the stock's performance.

Methodologies

Mean, Variance, Standard Deviation, Skewness and Kurtosis

```
Mean: 214.785895
Variance: 1819.000293
Std Dev: 42.649740
Skewness: -0.379325
Kurtosis: 2.396004
```

Mean: Represents the central tendency of a dataset.

Variance: Measures the spread of data points from the mean.

Standard Deviation: Indicates the average deviation of data points from the mean.

Skewness: Measures the asymmetry of a distribution.

Kurtosis: Measures the tailedness of a distribution.

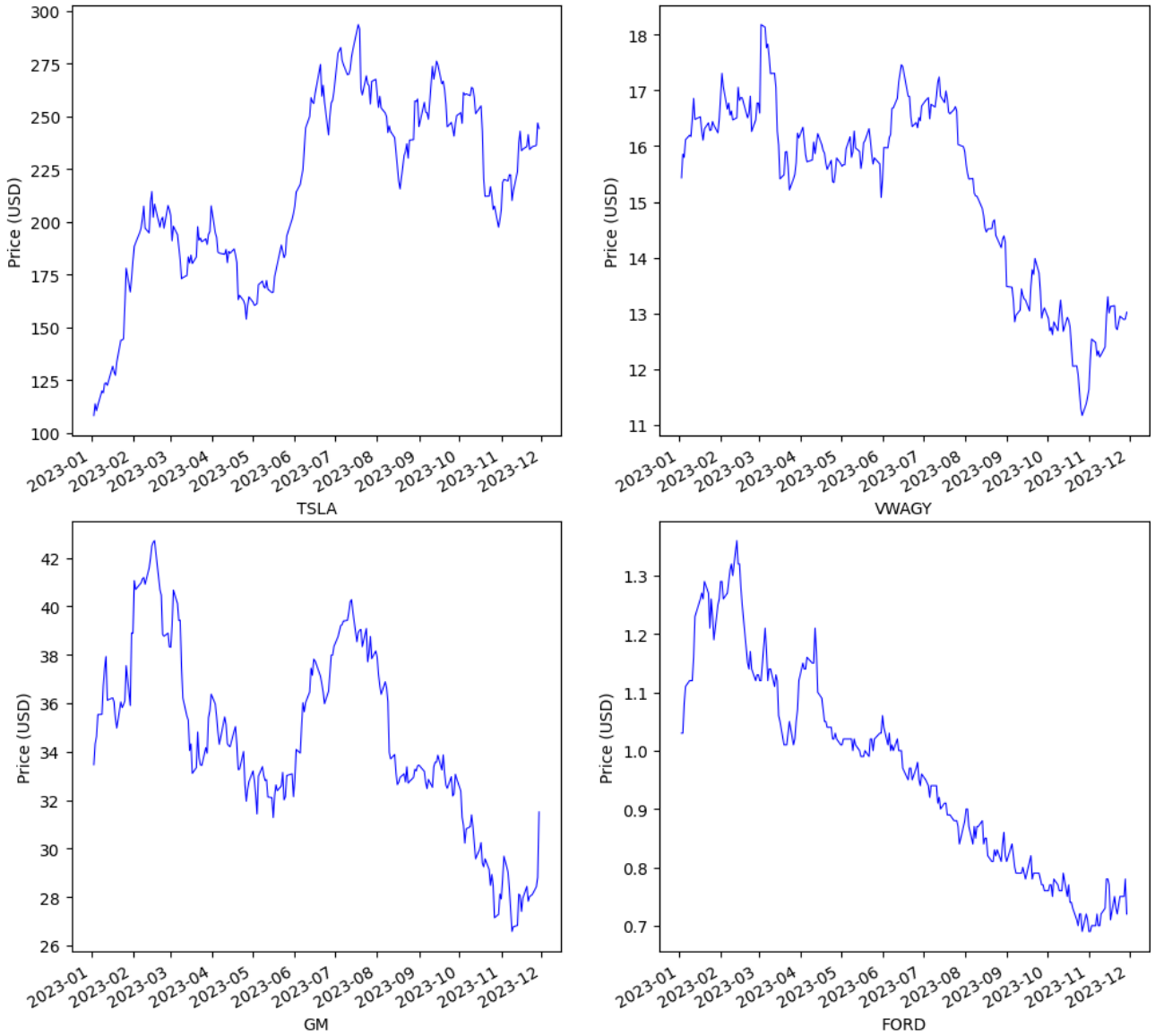
Stock Comparison

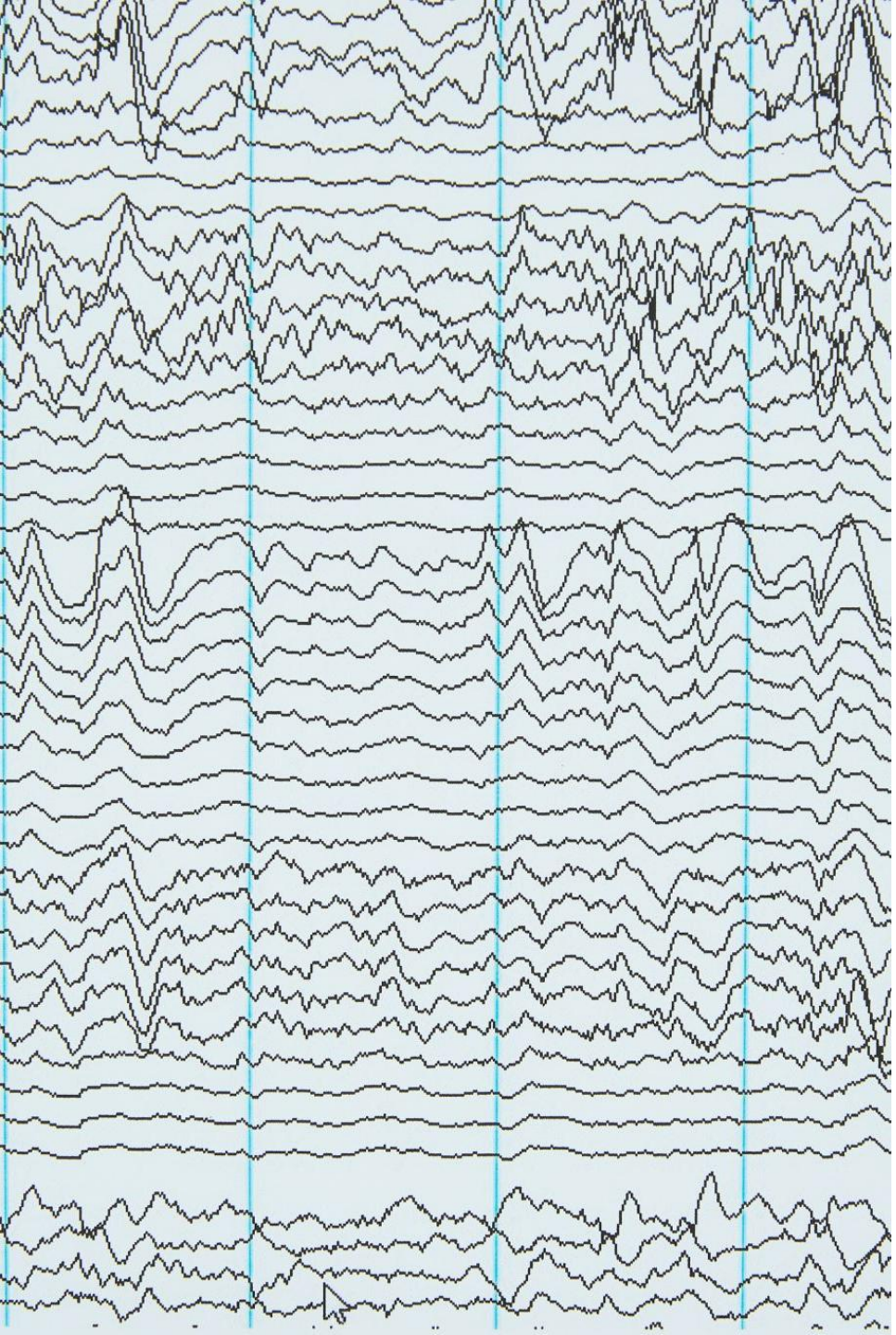
Tesla

Volkswagen

General Motors

Ford

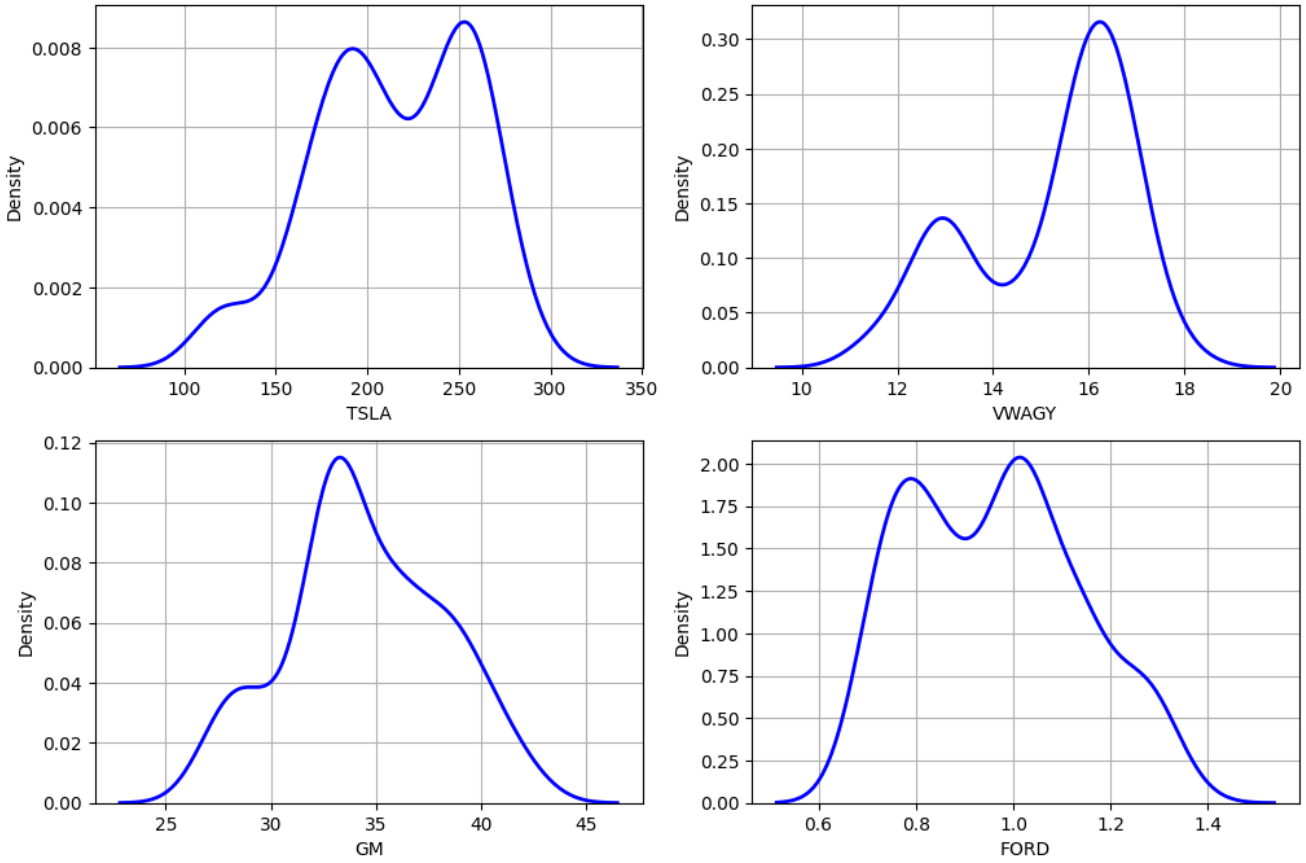




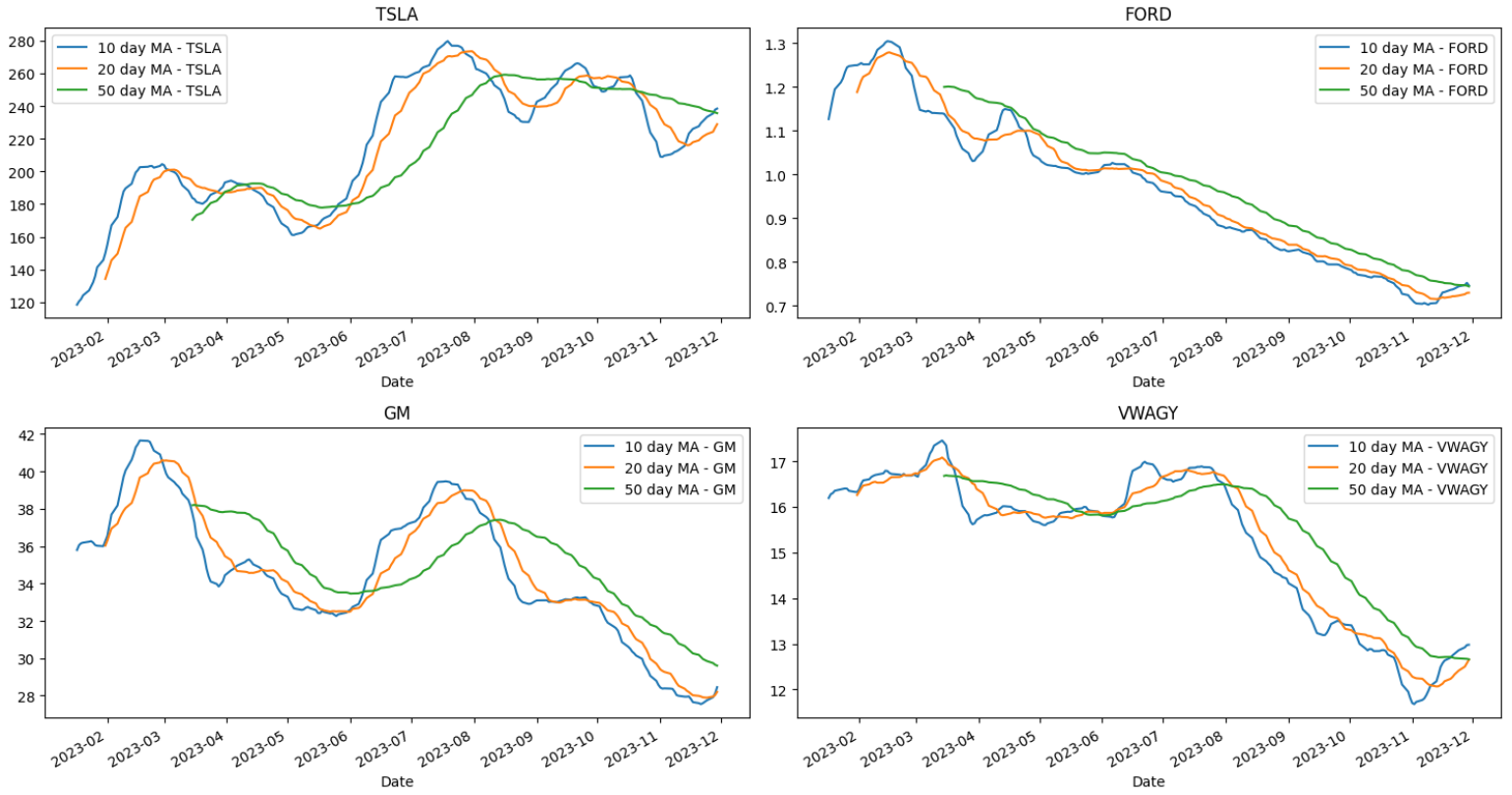
Kernel Density

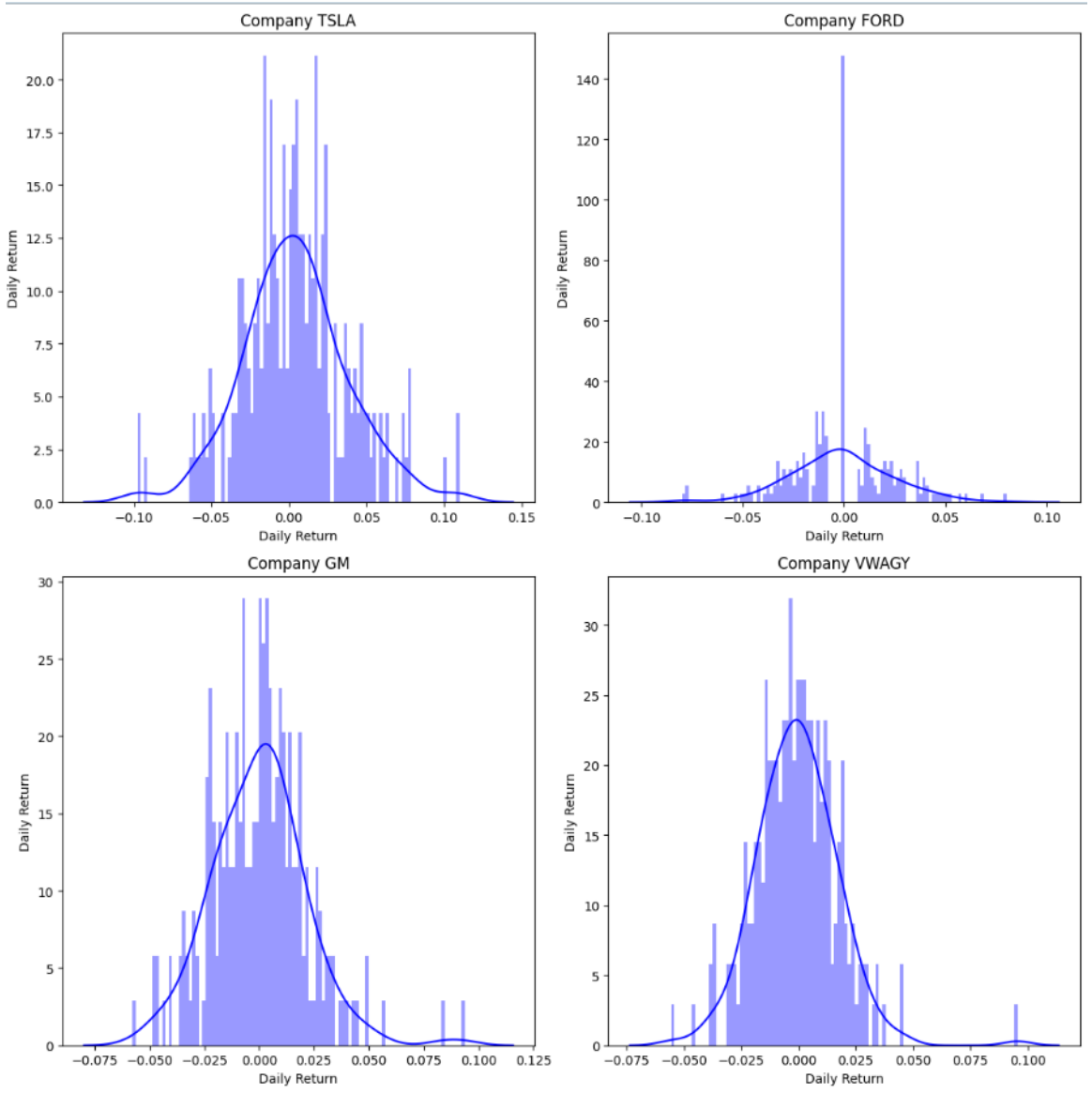
- The bandwidth parameter in KDE controls the amount of smoothing applied to the data. A smaller bandwidth results in a more sensitive estimate, closely following the data points, while a larger bandwidth produces a smoother estimate.
- KDE allows for the comparison of multiple distributions. By overlaying KDE plots for different datasets or subsets, analysts can visually compare the shapes and characteristics of the distributions.
- The area under the KDE curve over a specific range corresponds to the probability of observing data within that range. This property makes KDE a valuable tool for calculating probabilities and understanding the likelihood of certain events within the dataset.

Kernel density plot

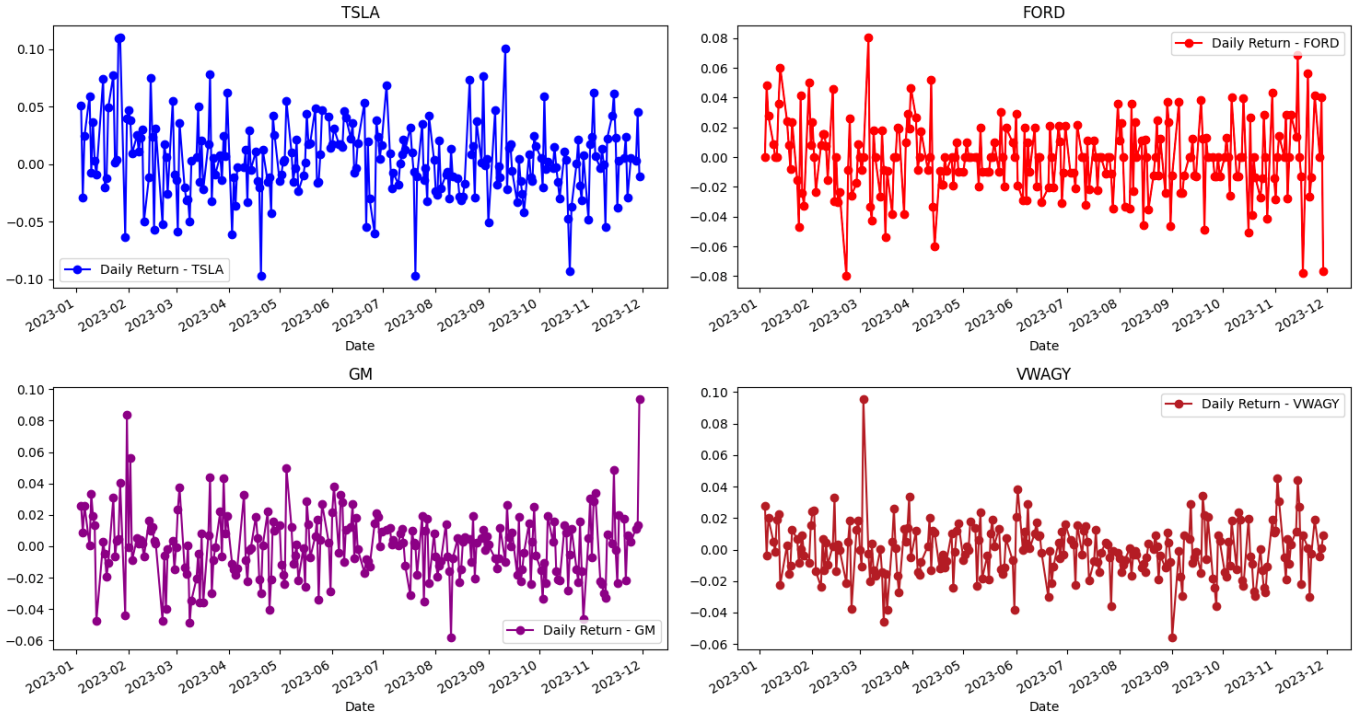


MOVING AVERAGE FOR ALL COMPETITORS



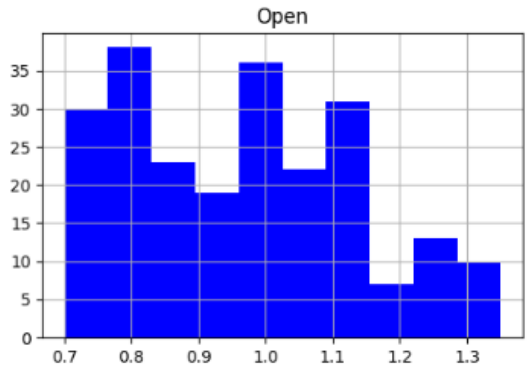


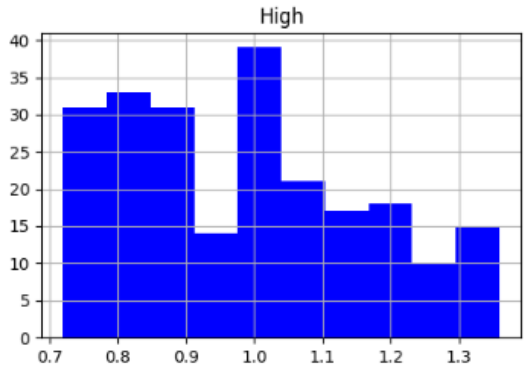
Calculating daily returns

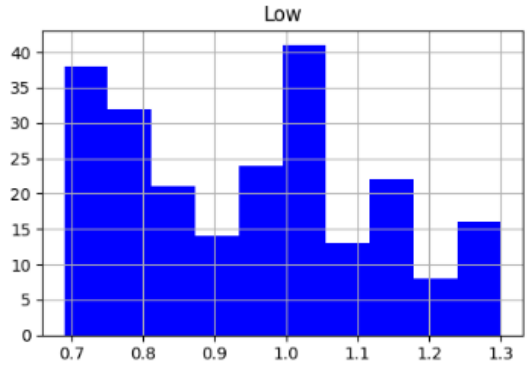


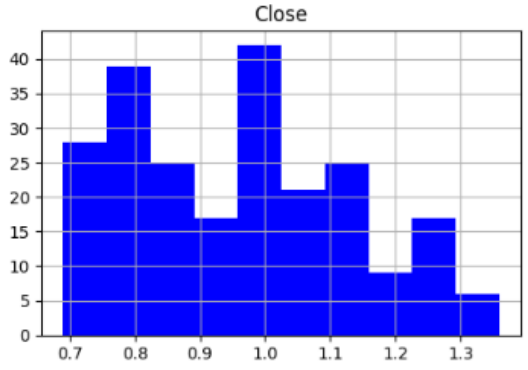
Percentage change – Daily Returns

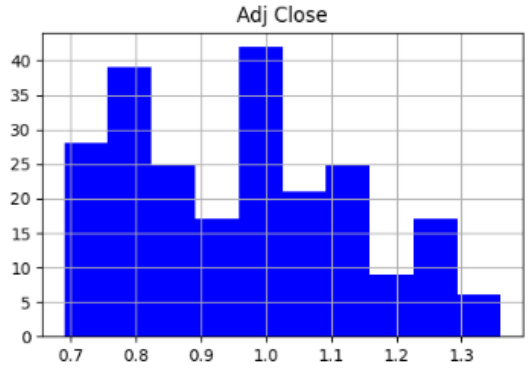
Histogram - FORD

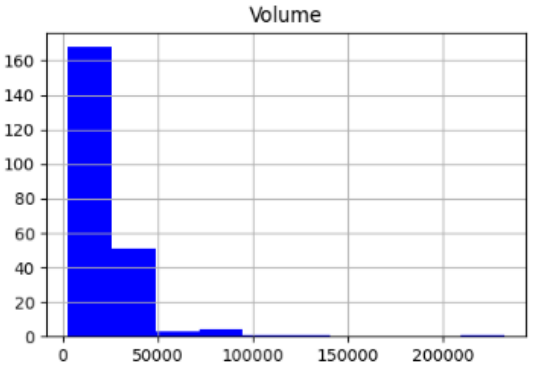




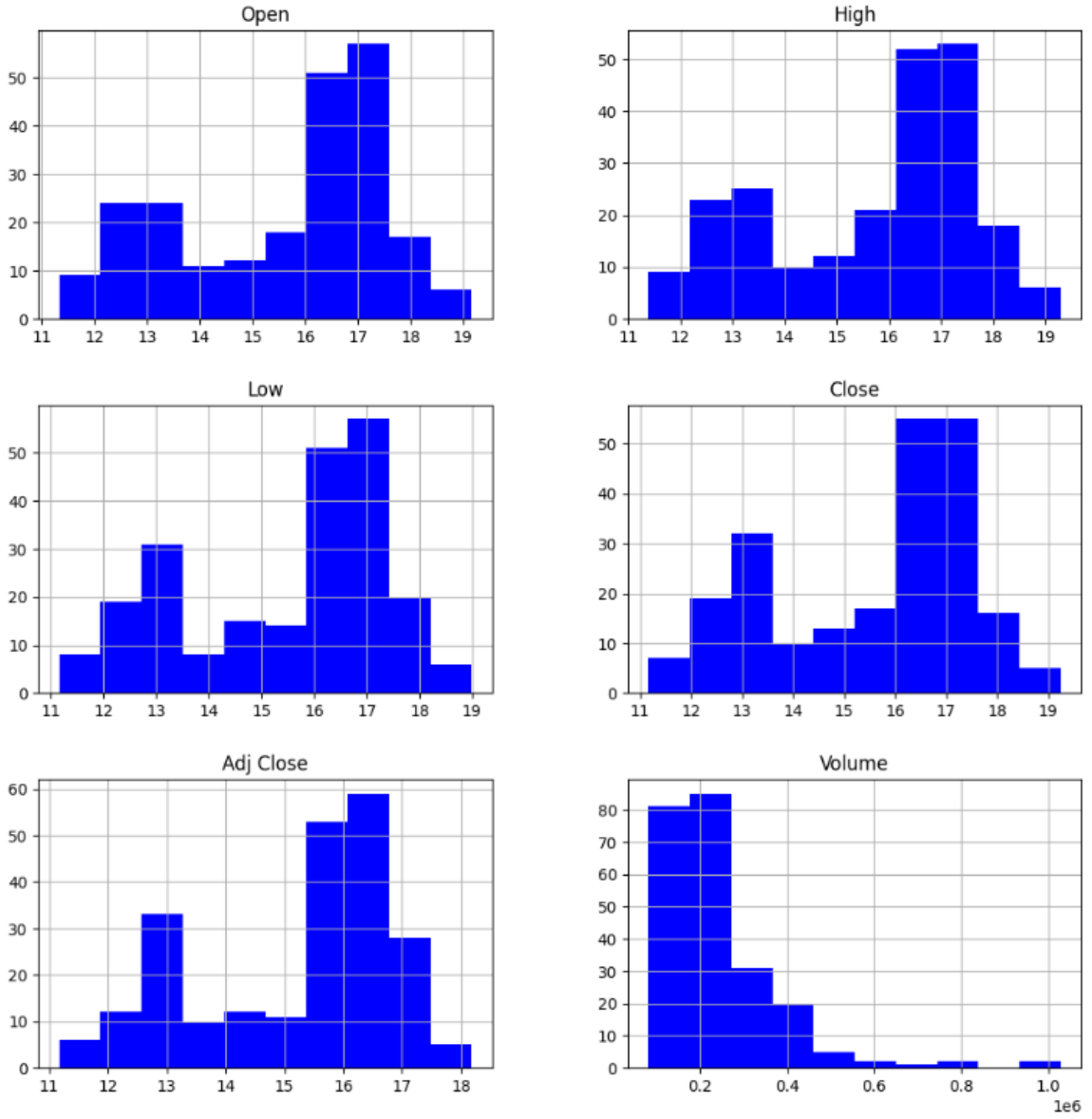




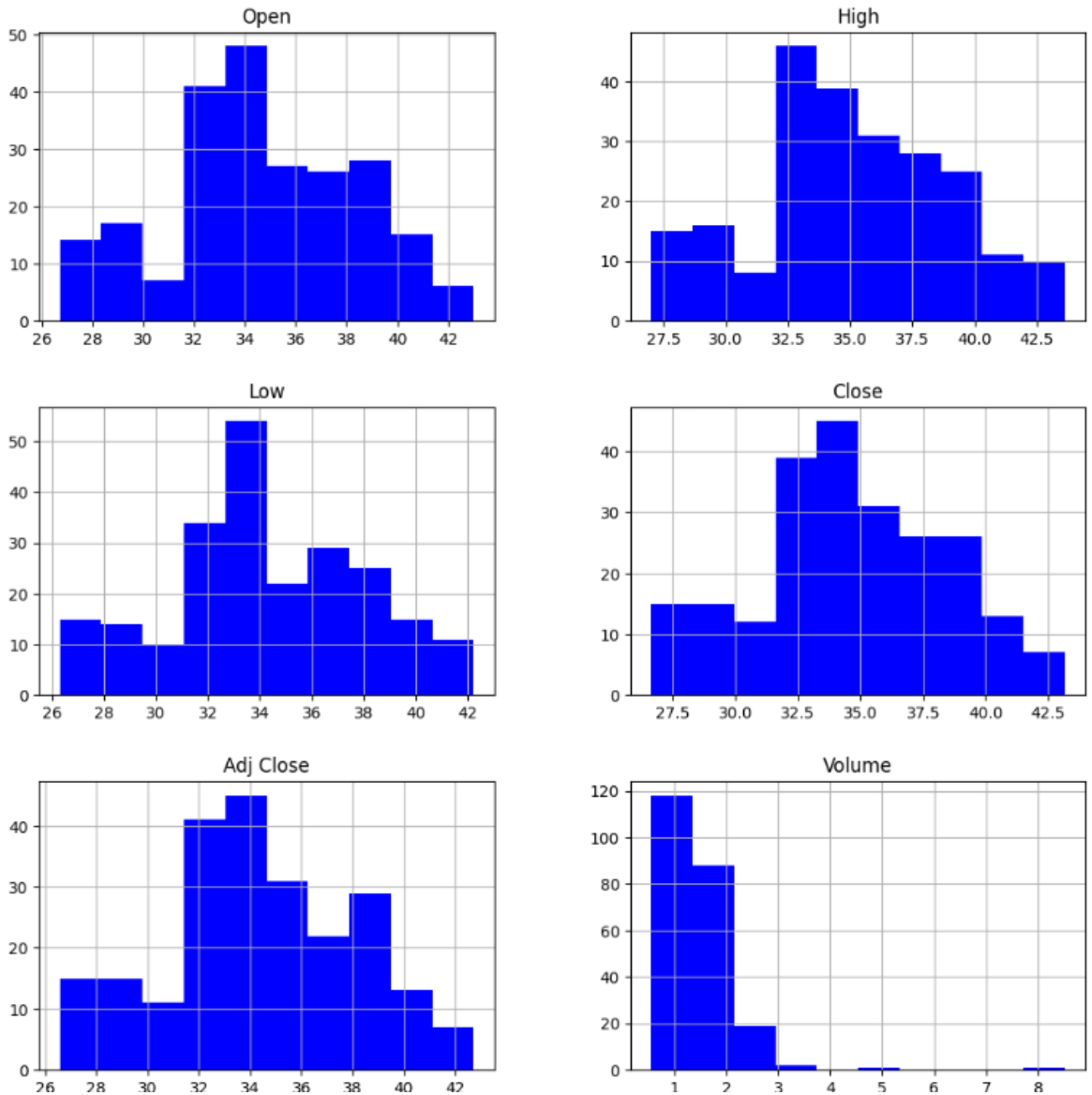




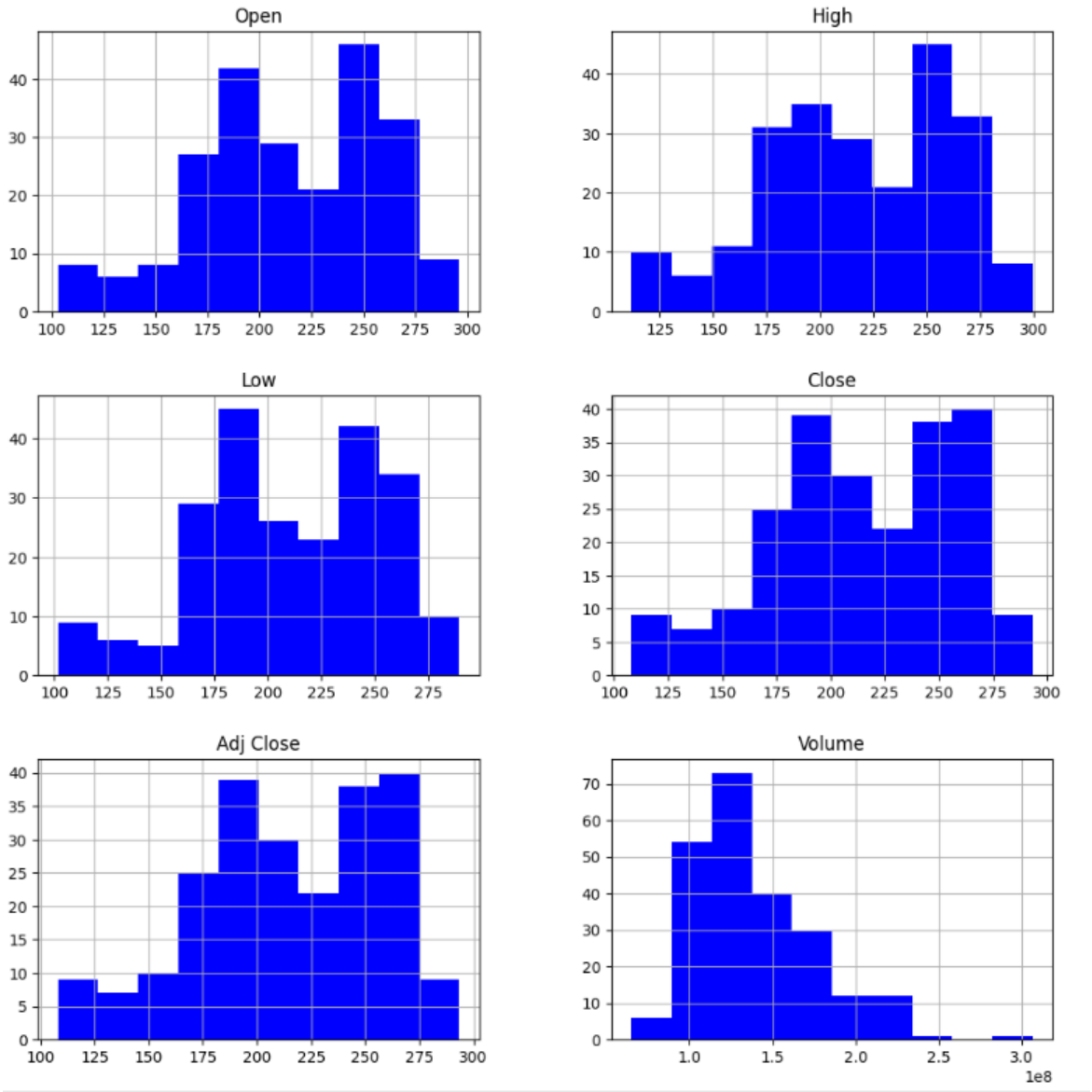
Histogram - VWAGY

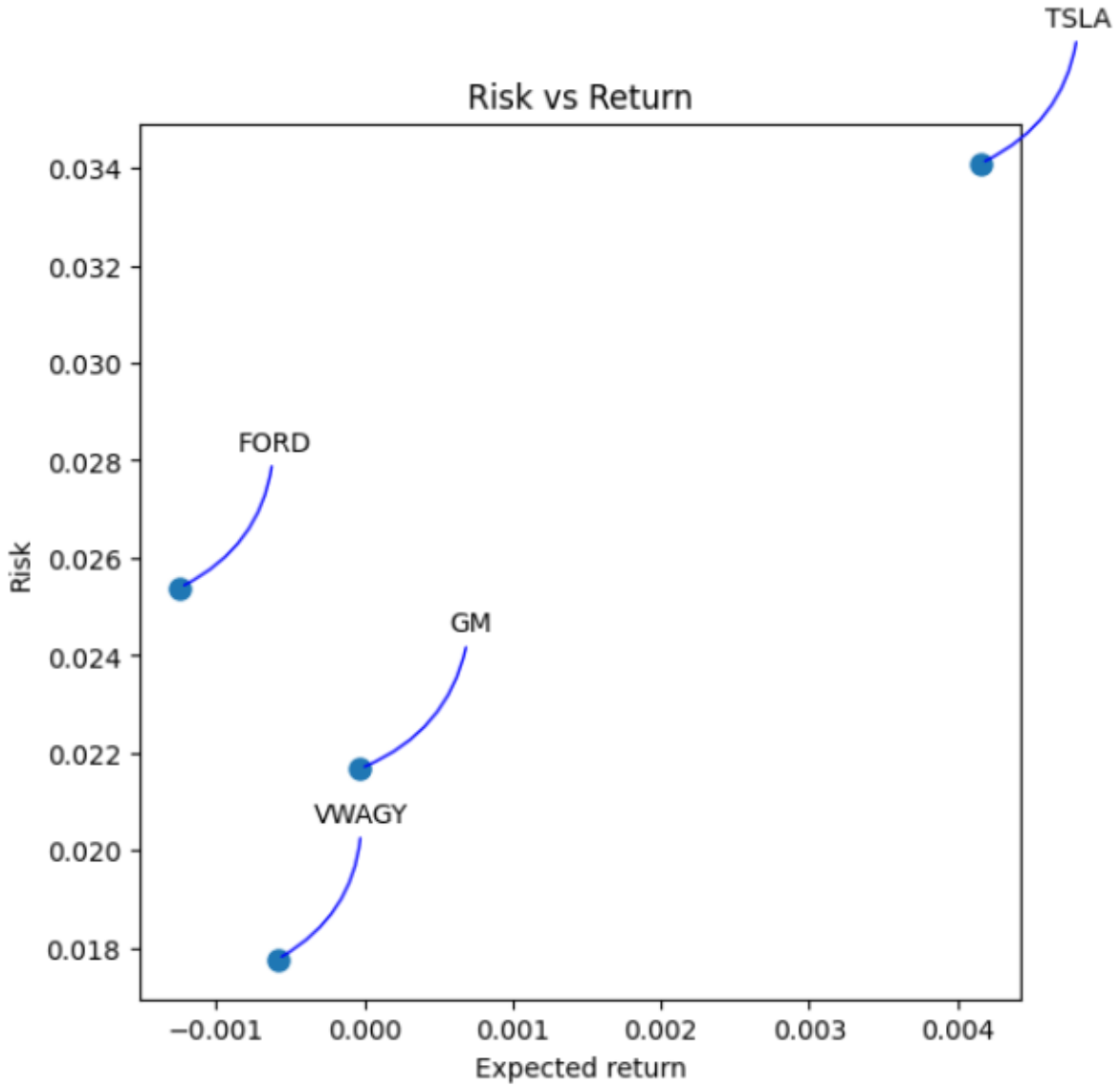


Histogram - GM



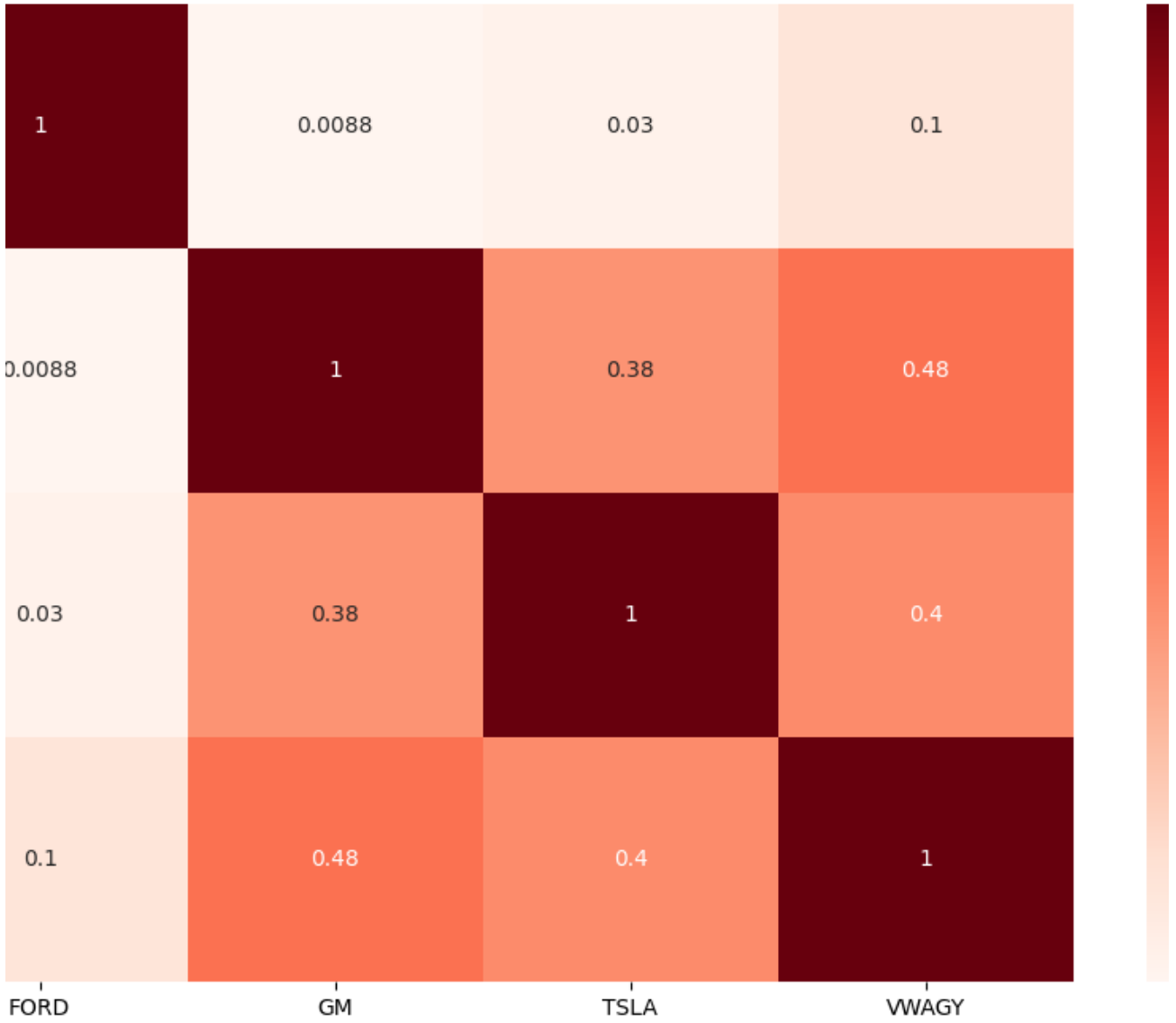
Histogram - TSLA



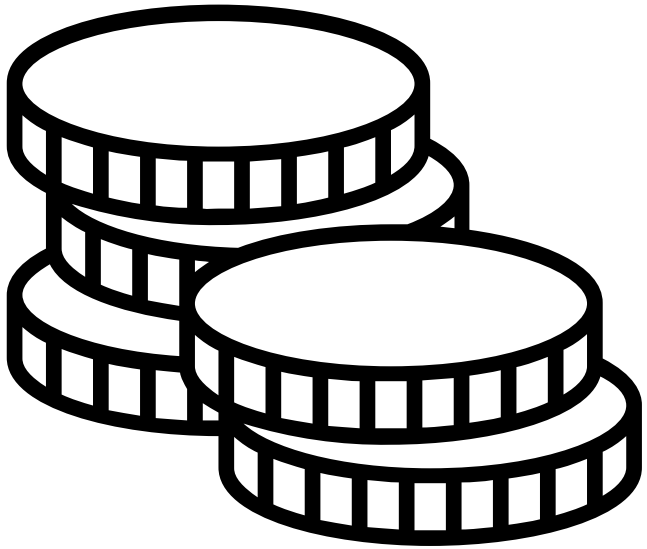


Risk vs Returns

Correlation between returns of all competitors



Historical Price Data	Fama-French Factors	Momentum Factors	Competitors Yest Close	Macro Economic Factor
• Open • Close • High • Low • Volume	• Mkt-RF (Market Risk Premium) • SMB (Small Minus Big) • HML (High Minus Low) • RF (Risk Free Rate)	• SMA (Simple Moving Average) • EMA (Exponential Moving Average) • RSI	• Volkswagen • General Motors • Ford	• ADS (Aruoba-Diebold-Scotti)



FEATURE ENGINEERING

```
stock_data.tail(10)
```

	Date	Open	High	Low	Close	Adj Close	Volume
3380	2023-12-01	233.139999	240.190002	231.899994	238.830002	238.830002	121173500
3381	2023-12-04	235.750000	239.369995	233.289993	235.580002	235.580002	104099800
3382	2023-12-05	233.869995	246.660004	233.699997	238.720001	238.720001	137971100
3383	2023-12-06	242.919998	246.570007	239.169998	239.369995	239.369995	126436200
3384	2023-12-07	241.550003	244.080002	236.979996	242.639999	242.639999	107142300
3385	2023-12-08	240.270004	245.270004	239.270004	243.839996	243.839996	102980100
3386	2023-12-11	242.740005	243.440002	237.449997	239.740005	239.740005	97913900
3387	2023-12-12	238.550003	238.990005	233.869995	237.009995	237.009995	95328300
3388	2023-12-13	234.190002	240.300003	228.199997	239.289993	239.289993	146286300
3389	2023-12-14	241.220001	253.880005	240.789993	251.050003	251.050003	160829200

```
stock_data.head(10)
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2010-06-29	1.266667	1.666667	1.169333	1.592667	1.592667	281494500
1	2010-06-30	1.719333	2.028000	1.553333	1.588667	1.588667	257806500
2	2010-07-01	1.666667	1.728000	1.351333	1.464000	1.464000	123282000
3	2010-07-02	1.533333	1.540000	1.247333	1.280000	1.280000	77097000
4	2010-07-06	1.333333	1.333333	1.055333	1.074000	1.074000	103003500
5	2010-07-07	1.093333	1.108667	0.998667	1.053333	1.053333	103825500
6	2010-07-08	1.076000	1.168000	1.038000	1.164000	1.164000	115671000
7	2010-07-09	1.172000	1.193333	1.103333	1.160000	1.160000	60759000
8	2010-07-12	1.196667	1.204667	1.133333	1.136667	1.136667	33037500
9	2010-07-13	1.159333	1.242667	1.126667	1.209333	1.209333	40201500

stock_data

Generating Momentum Factors with stock_data

```
stock_data["Returns"] = stock_data["Close"].pct_change()
stock_data["MovingAverage_50"] = stock_data["Close"].rolling(window=50).mean()
stock_data["MovingAverage_200"] = stock_data["Close"].rolling(window=200).mean()
stock_data["VolumeChange"] = stock_data["Volume"].pct_change()
stock_data["VolumeMovingAverage_50"] = stock_data["Volume"].rolling(window=50).mean()
stock_data["VolumeMovingAverage_200"] = stock_data["Volume"].rolling(window=200).mean()
risk_free_rate = 0.02
stock_data["MarketReturns"] = stock_data["Returns"].mean()
stock_data["CAPM"] = risk_free_rate + stock_data["MarketReturns"] * (stock_data["Returns"] - risk_free_rate)
stock_data["HighLowRange"] = stock_data["High"] - stock_data["Low"]
stock_data["HighCloseRange"] = stock_data["High"] - stock_data["Close"].shift()
stock_data["LowCloseRange"] = stock_data["Close"].shift() - stock_data["Low"]
stock_data["Volatility"] = stock_data[["HighLowRange", "HighCloseRange", "LowCloseRange"]].mean(axis=1)
window = 14
delta = stock_data["Close"].diff()
gain = delta.where(delta > 0, 0)
loss = -delta.where(delta < 0, 0)
average_gain = gain.rolling(window=window).mean()
average_loss = loss.rolling(window=window).mean()
relative_strength = average_gain / average_loss
stock_data["RSI"] = 100 - (100 / (1 + relative_strength))
```

```
# Momentum factors
window = 10
stock_data["Momentum"] = stock_data["Close"] - stock_data["Close"].shift(window)

# Volume-related features
stock_data["VolumeChange"] = stock_data["Volume"].pct_change()
stock_data["VolumeMovingAverage_10"] = stock_data["Volume"].rolling(window=10).mean()

# Price/return lags
lags = [1, 3, 5]  # Define the desired lag periods
for lag in lags:
    stock_data[f"PriceLag_{lag}"] = stock_data["Close"].shift(lag)
    stock_data[f"ReturnLag_{lag}"] = stock_data["Returns"].shift(lag)
```

Fama/French 5 Factors (2x3) [Daily]

```
# fama_french_data

fama_french_data = pd.read_csv('/content/F-F_Research_Data_5_Factors_2x3_daily.CSV')
fama_french_data.head()
fama_french_data = fama_french_data.rename(columns={'date': 'Date'})
fama_french_data.head()
fama_french_data["Date"] = pd.to_datetime(fama_french_data["Date"], format="%Y%m%d")
# fama_french_data.drop("Date", axis=1, inplace=True)
fama_french_data.head()
```

	Date	Mkt-RF	SMB	HML	RMW	CMA	RF
0	1963-07-01	-0.67	0.02	-0.35	0.03	0.13	0.012
1	1963-07-02	0.79	-0.28	0.28	-0.08	-0.21	0.012
2	1963-07-03	0.63	-0.18	-0.10	0.13	-0.25	0.012
3	1963-07-05	0.40	0.09	-0.28	0.07	-0.30	0.012
4	1963-07-08	-0.63	0.07	-0.20	-0.27	0.06	0.012

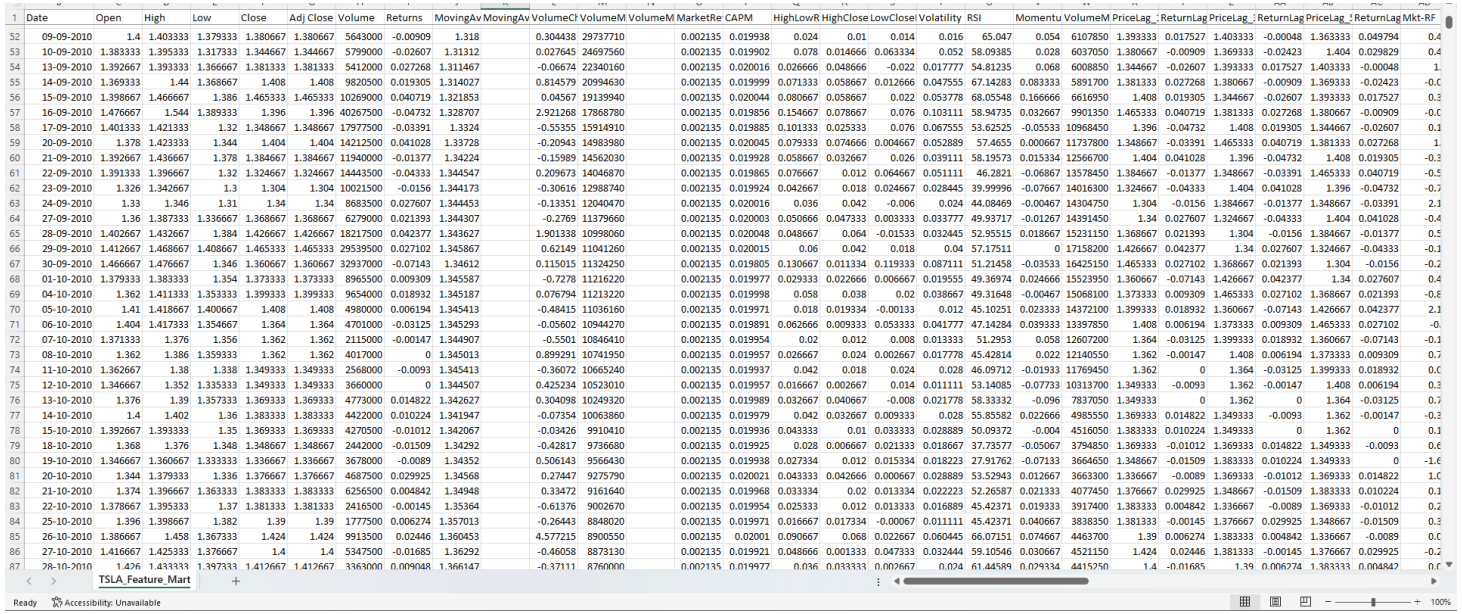
ADS Index Most Current Vintage

```
# ADS_Index_Most_Current_Vintage
ADS_Index_Most_Current_Vintage = pd.read_excel('/content/ADS_Index_Most_Current_Vintage.xlsx')

ADS_Index_Most_Current_Vintage = ADS_Index_Most_Current_Vintage.rename(columns={'date': 'Date'})
ADS_Index_Most_Current_Vintage.head()

ADS_Index_Most_Current_Vintage["Date"] = pd.to_datetime(ADS_Index_Most_Current_Vintage["Date"], format="%Y:%m:%d")
ADS_Index_Most_Current_Vintage.head()
```

	Date	ADS_Index
0	1960-03-01	-0.572849
1	1960-03-02	-0.621380
2	1960-03-03	-0.666849
3	1960-03-04	-0.709275
4	1960-03-05	-0.748672

FRED - Federal Reserve Economic Data

```
# fill with your own FRED API
var_list = ['T10Y3M', 'OBMMIJUMBO30YF',  # term premium 10yr-3mon, 30 yr mortgage jumbo loan
            'DEXUSEU', 'DEXJPUS', 'DEXUSUK', # spot exchange rates to EUR, JPY, GBP
            'CBBTCUSD', 'CBETHUSD',  # cryptocurrencies
                'T10YIE', 'DCOILBRENTEU', # breakeven inflation + brent oil price
                'VIXCLS', # implied volatilities
                'DAAA', 'DBAA', # corporate bond yield
                'NIKKEI225', 'AMERIBOR', 'T10YIE', 'T5YIE', 'BAMLH0A0HYM2','BAMLH0A0HYM2EY',
                                'DGS10', 'DGS1', 'RIFSPPFAAD90NB', 'DCPN3M', 'DCPF1M', 'DCOILWTICO',
                                'DHHNGSP', 'USRECD', 'USRECDM', 'USRECDP', # JPN stock mkt index
             'UNRATE', # unemployment rate
             'SP500']

# api 307baa1e181fcc4713a148763423c76e
fred = Fred(api_key='307baa1e181fcc4713a148763423c76e')

data_dict = {}
for var in var_list:
    series = fred.get_series(var)
    series.name = var
    data_dict[var] = series

# Create a DataFrame from the fetched data
df_fred = pd.DataFrame(data_dict)

df_fred.tail()
```



Date	Open	High	Low	Close	Adj Close	Volume	Returns	MovingAv	MovingAv	VolumeCl	VolumeM	VolumeM	MarketRe	CAPM	HighLowR	HighClose	LowClose	Volatility	RSI	Momentu	VolumeM	PriceLag_	ReturnLag	PriceLag_	ReturnLag	PriceLag_	ReturnLag	Mkt-RF
09-09-2010	1.4	1.403333	1.379333	1.380667	1.380667	5643000	-0.00909	1.318		0.304438	29737710		0.002135	0.019938	0.024	0.01	0.014	0.016	65.047	0.054	6107850	1.393333	0.017527	1.403333	-0.00048	1.363333	0.049794	0.4
10-09-2010	1.383333	1.395333	1.317333	1.344667	1.344667	5799000	-0.02607	1.31312		0.027645	24697560		0.002135	0.019902	0.078	0.014666	0.063334	0.052	58.09385	0.028	6037050	1.380667	-0.00909	1.369333	-0.02423	1.404	0.029829	0.4
13-09-2010	1.392667	1.393333	1.366667	1.381333	1.381333	5412000	0.027268	1.311467		-0.06674	22340160		0.002135	0.020016	0.026666	0.048666	-0.022	0.017777	54.81235	0.068	6008850	1.344667	-0.02607	1.393333	0.017527	1.403333	-0.00048	1.
14-09-2010	1.369333	1.44	1.368667	1.408	1.408	9820500	0.019305	1.314027		0.814579	20994630		0.002135	0.019999	0.071333	0.058667	0.012666	0.047555	67.14283	0.083333	5891700	1.381333	0.027268	1.380667	-0.00909	1.369333	-0.02423	-0.0
15-09-2010	1.398667	1.466667	1.386	1.465333	1.465333	10269000	0.040719	1.321853		0.04567	19139940		0.002135	0.020044	0.080667	0.058667	0.022	0.053778	68.05548	0.166666	6616950	1.408	0.019305	1.344667	-0.02607	1.393333	0.017527	0.3
16-09-2010	1.476667	1.544	1.389333	1.396	1.396	40267500	-0.04732	1.328707		2.921268	17868780		0.002135	0.019856	0.154667	0.078667	0.076	0.103111	58.94735	0.032667	9901350	1.465333	0.040719	1.381333	0.027268	1.380667	-0.00909	-0.0
17-09-2010	1.401333	1.421333	1.32	1.348667	1.348667	17977500	-0.03391	1.3324		-0.55355	15914910		0.002135	0.019885	0.101333	0.025333	0.076	0.067555	53.62525	-0.05533	10968450	1.396	-0.04732	1.408	0.019305	1.344667	-0.02607	0.1
20-09-2010	1.378	1.423333	1.344	1.404	1.404	14212500	0.041028	1.33728		-0.20943	14983980		0.002135	0.020045	0.079333	0.074666	0.004667	0.052889	57.4655	0.000667	11737800	1.348667	-0.03391	1.465333	0.040719	1.381333	0.027268	1.
21-09-2010	1.392667	1.436667	1.378	1.384667	1.384667	11940000	-0.01377	1.34224		-0.15989	14562030		0.002135	0.019928	0.058667	0.032667	0.026	0.039111	58.19573	0.015334	12566700	1.404	0.041028	1.396	-0.04732	1.408	0.019305	-0.3
22-09-2010	1.391333	1.396667	1.32	1.324667	1.324667	14443500	-0.04333	1.344547		0.209673	14046870		0.002135	0.019865	0.076667	0.012	0.064667	0.051111	46.2821	-0.06867	13578450	1.384667	-0.01377	1.348667	-0.03391	1.465333	0.040719	-0.5
23-09-2010	1.326	1.342667	1.3	1.304	1.304	10021500	-0.0156	1.344173		-0.30616	12988740		0.002135	0.019924	0.042667	0.018	0.024667	0.028445	39.99996	-0.07667	14016300	1.324667	-0.04333	1.404	0.041028	1.396	-0.04732	-0.7
24-09-2010	1.33	1.346	1.31	1.34	1.34	8683500	0.027607	1.344453		-0.13351	12040470		0.002135	0.020016	0.036	0.042	-0.006	0.024	44.08469	-0.00467	14304750	1.304	-0.0156	1.384667	-0.01377	1.348667	-0.03391	2.1
27-09-2010	1.36	1.387333	1.336667	1.368667	1.368667	6279000	0.021393	1.344307		-0.2769	11379660		0.002135	0.020003	0.050666	0.047333	0.003333	0.033777	49.93717	-0.01267	14391450	1.34	0.027607	1.324667	-0.04333	1.404	0.041028	-0.4
28-09-2010	1.402667	1.432667	1.384	1.426667	1.426667	18217500	0.042377	1.343627		1.901338	10998060		0.002135	0.020048	0.048667	0.064	-0.01533	0.032445	52.95515	0.018667	15231150	1.368667	0.021393	1.304	-0.0156	1.384667	-0.01377	0.5
29-09-2010	1.412667	1.468667	1.408667	1.465333	1.465333	29539500	0.027102	1.345867		0.62149	11041260		0.002135	0.020015	0.06	0.042	0.018	0.04	57.17511	0	17158200	1.426667	0.042377	1.34	0.027607	1.324667	-0.04333	-0.1
30-09-2010	1.466667	1.476667	1.346	1.360667	1.360667	32937000	-0.07143	1.34612		0.115015	11324250		0.002135	0.019805	0.130667	0.011334	0.119333	0.087111	51.21458	-0.03533	16425150	1.465333	0.027102	1.368667	0.021393	1.304	-0.0156	-0.2
01-10-2010	1.379333	1.383333	1.354	1.373333	1.373333	8965500	0.009309	1.345587		-0.7278	11216220		0.002135	0.019977	0.029333	0.022666	0.006667	0.019555	49.36974	0.024666	15523950	1.360667	-0.07143	1.426667	0.042377	1.34	0.027607	0.4
04-10-2010	1.362	1.411333	1.353333	1.399333	1.399333	9654000	0.018932	1.345187		0.076794	11213220		0.002135	0.019998	0.058	0.038	0.02	0.038667	49.31648	-0.00467	15068100	1.373333	0.009309	1.465333	0.027102	1.368667	0.021393	-0.8
05-10-2010	1.41	1.418667	1.400667	1.408	1.408	4980000	0.006194	1.345413		-0.48415	11036160		0.002135	0.019971	0.018	0.019334	-0.00133	0.012	45.10251	0.023333	14372100	1.399333	0.018932	1.360667	-0.07143	1.426667	0.042377	2.1
06-10-2010	1.404	1.417333	1.354667	1.364	1.364	4701000	-0.03125	1.345293		-0.05602	10944270		0.002135	0.019891	0.062666	0.009333	0.053333	0.041777	47.14284	0.039333	13397850	1.408	0.006194	1.373333	0.009309	1.465333	0.027102	-0.
07-10-2010	1.371333	1.376	1.356	1.362	1.362	2115000	-0.00147	1.344907		-0.5501	10846410		0.002135	0.019954	0.02	0.012	0.008	0.013333	51.2953	0.058	12607200	1.364	-0.03125	1.399333	0.018932	1.360667	-0.07143	-0.1
08-10-2010	1.362	1.386	1.359333	1.362	1.362	4017000	0	1.345013		0.899291	10741950		0.002135	0.019957	0.026667	0.024	0.002667	0.017778	45.42814	0.022	12140550	1.362	-0.00147	1.408	0.006194	1.373333	0.009309	0.7
11-10-2010	1.362667	1.38	1.338	1.349333	1.349333	2568000	-0.0093	1.345413		-0.36072	10665240		0.002135	0.019937	0.042	0.018	0.024	0.028	46.09712	-0.01933	11769450	1.362	0	1.364	-0.03125	1.399333	0.018932	0.0
12-10-2010	1.346667	1.352	1.335333	1.349333	1.349333	3660000	0	1.344507		0.425234	10523010		0.002135	0.019957	0.016667	0.002667	0.014	0.011111	53.14085	-0.07733	10313700	1.349333	-0.0093	1.362	-0.00147	1.408	0.006194	0.3
13-10-2010	1.376	1.39	1.357333	1.369333	1.369333	4773000	0.014822	1.342627		0.304098	10249320		0.002135	0.019989	0.032667	0.040667	-0.008	0.021778	58.33332	-0.096	7837050	1.349333	0	1.362	0	1.364	-0.03125	0.7
14-10-2010	1.4	1.402	1.36	1.383333	1.383333	4422000	0.010224	1.341947		-0.07354	10063860		0.002135	0.019979	0.042	0.032667	0.009333	0.028	55.85582	0.022666	4985550	1.369333	0.014822	1.349333	-0.0093	1.362	-0.00147	-0.3
15-10-2010	1.392667	1.393333	1.35	1.369333	1.369333	4270500	-0.01012	1.342067		-0.03426	9910410		0.002135	0.019936	0.043333	0.01	0.033333	0.028889	50.09372	-0.004	4516050	1.383333	0.010224	1.349333	0	1.362	0	0.1
18-10-2010	1.368	1.376	1.348	1.348667	1.348667	2442000	-0.01509	1.34292		-0.42817	9736680		0.002135	0.019925	0.028	0.006667	0.021333	0.018667	37.73577	-0.05067	3794850	1.369333	-0.01012	1.369333	0.014822	1.349333	-0.0093	0.6
19-10-2010	1.346667	1.360667	1.333333	1.336667	1.336667	3678000	-0.0089	1.34352		0.506143	9566430		0.002135	0.019938	0.027334	0.012	0.015334	0.018223	27.91762	-0.07133	3664650	1.348667	-0.01509	1.383333	0.010224	1.349333	0	-1.6
20-10-2010	1.344	1.379333	1.336	1.376667	1.376667	4687500	0.029925	1.34568		0.27447	9275790		0.002135	0.020021	0.043333	0.042666	0.000667	0.028889	53.52943	0.012667	3663300	1.336667	-0.0089	1.369333	-0.01012	1.369333	0.014822	1.0
21-10-2010	1.374	1.396667	1.363333	1.383333	1.383333	6256500	0.004842	1.34948		0.33472	9161640		0.002135	0.019968	0.033334	0.02	0.013334	0.022223	52.26587	0.021333	4077450	1.376667	0.029925	1.348667	-0.01509	1.383333	0.010224	0.1
22-10-2010	1.378667	1.395333	1.37	1.381333	1.381333	2416500	-0.00145	1.35364		-0.61376	9002670		0.002135	0.019954	0.025333	0.012	0.013333	0.016889	45.42371	0.019333	3917400	1.383333	0.004842	1.336667	-0.0089	1.369333	-0.01012	0.2
25-10-2010	1.396	1.398667	1.382	1.39	1.39	1777500	0.006274	1.357013		-0.26443	8848020		0.002135	0.019971	0.016667	0.017334	-0.00067	0.011111	45.42371	0.040667	3838350	1.381333	-0.00145	1.376667	0.029925	1.348667	-0.01509	0.3
26-10-2010	1.386667	1.458	1.367333	1.424	1.424	9913500	0.02446	1.360453		4.577215	8900550		0.002135	0.02001	0.090667	0.068	0.022667	0.060445	66.07151	0.074667	4463700	1.39	0.006274	1.383333	0.004842	1.336667	-0.0089	0.0
27-10-2010	1.416667	1.425333	1.376667	1.4	1.4	5347500	-0.01685	1.36292		-0.46058	8873130		0.002135	0.019921	0.048666	0.001333	0.047333	0.032444	59.10546	0.030667	4521150	1.424	0.02446	1.381333	-0.00145	1.376667	0.029925	-0.2
28-10-2010	1.426	1.433333	1.397333	1.412667	1.412667	3363000	0.009048	1.366147		-0.37111	8760000		0.002135	0.019977	0.036	0.033333	0.002667	0.024	61.44589	0.029334	4415250	1.4	-0.01685	1.39	0.006274	1.383333	0.004842	0.0

TSLA_Feature_Mart

Ready Accessibility: Unavailable 100%

Resulting Feature Mart

Date	ADS_Inde	T10Y3M	OBMMIJU	DEXUSEU	DEXJPUS	DEXUSUK	CBBTCUSD	CBETHUSD	T10YIE	DCOILBRE	VIXCLS	DAAA	DBAA	NIKKEI225	AMERIBOI	T5YIE	BAMLH0A	BAMLH0A	DGS10	DGS1	RIFSPPFA	DCPN3M	DCPF1M	DCOILWTI	DHHNGSP	USRECD	USRECDM	USRECD
29-06-2010	-0.03355	2.82		1.2187	88.39	1.5081			1.83	74.21	34.13	4.71	6.09	9570.67		1.54	7.06	8.96	2.97	0.31	0.43	0.33	0.18	75.93	4.68	0	0	
30-06-2010	-0.02233	2.79		1.2291	88.49	1.4947			1.82	74.94	34.54	4.66	6.05	9382.64		1.54	7.13	9.02	2.97	0.32	0.42		0.19	75.59	4.53	0	0	
01-07-2010	-0.00971	2.79		1.2464	87.42	1.5126			1.75	71.73	32.86	4.62	5.98	9191.6		1.48	7.17	9.05	2.96	0.32	0.4		0.24	72.95	4.54	0	0	
02-07-2010	0.001724	2.83		1.2577	87.69	1.5179			1.73	71.75	30.12	4.69	6.04	9203.71		1.48	7.13	9.04	3	0.31	0.37	0.28	0.25	72.06	4.72	0	0	
06-07-2010	0.034921	2.78		1.2646	87.55	1.518			1.69	73.08	29.65	4.64	6	9338.04		1.41	7.14	9	2.95	0.32	0.48	0.3	0.24	71.96	4.85	0	0	
07-07-2010	0.039759	2.84		1.2594	87.16	1.5179			1.7	72.97	26.84	4.71	6.08	9279.65		1.4	7.1	8.99	3	0.31	0.5	0.3	0.23	74.05	4.76	0	0	
08-07-2010	0.043194	2.89		1.2683	88.44	1.5157			1.79	74.56	25.71	4.75	6.11	9535.74		1.43	6.98	8.89	3.04	0.3	0.36	0.26	0.27	75.46	4.61	0	0	
09-07-2010	0.045227	2.91		1.2639	88.48	1.5095			1.81	75.2	24.98	4.79	6.14	9585.32		1.47	6.88	8.83	3.07	0.3	0.38		0.25	76.08	4.36	0	0	
12-07-2010	0.044617	2.92		1.2573	88.53	1.5016			1.82	74.35	24.43	4.79	6.13	9548.11		1.48	6.85	8.79	3.08	0.3	0.37	0.3	0.24	74.93	4.42	0	0	
13-07-2010	0.044442	3		1.2719	88.24	1.5175			1.87	76.45	24.56	4.86	6.18	9537.23		1.51	6.7	8.69	3.15	0.3	0.36	0.28	0.26	77.16	4.46	0	0	
14-07-2010	0.044563	2.92		1.2755	88.59	1.5284			1.84	76.63	24.89	4.76	6.09	9795.24		1.48	6.68	8.6	3.07	0.27	0.35	0.29	0.22	77.02	4.39	0	0	
15-07-2010	0.04498	2.85		1.2893	87.41	1.5376			1.79	75.52	25.14	4.68	6.02	9685.53		1.43	6.73	8.58	3	0.27	0.41	0.29	0.26	76.67	4.43	0	0	
16-07-2010	0.045694	2.81		1.2919	86.4	1.5289			1.71	75.55	26.25	4.67	6	9408.36		1.36	6.77	8.58	2.96	0.28	0.4		0.26	75.96	4.68	0	0	
19-07-2010	0.048745	2.82		1.2963	86.77	1.5222			1.7	76.29	25.97	4.7	6.03			1.34	6.77	8.6	2.99	0.28	0.36	0.29	0.24	76.53	4.56	0	0	
20-07-2010	0.0489	2.82		1.2905	87.22	1.5255			1.7	76.31	23.93	4.68	5.97	9300.46		1.32	6.79	8.59	2.98	0.27	0.41		0.25	77.32	4.59	0	0	
21-07-2010	0.048479	2.74		1.2818	87.3	1.5201			1.71	75.75	25.64	4.61	5.88	9278.83		1.34	6.75	8.52	2.9	0.27	0.3	0.28	0.24	76.27	4.7	0	0	
22-07-2010	0.04748	2.8		1.2903	87	1.5283			1.75	77.59	24.63	4.7	5.93	9220.88		1.37	6.67	8.46	2.96	0.27	0.41	0.24	0.19	79.01	4.67	0	0	
23-07-2010	0.045905	2.86		1.2874	87.31	1.5425			1.78	77.27	23.47	4.74	5.97	9430.96		1.39	6.59	8.44	3.02	0.27	0.41		0.22	78.68	4.69	0	0	
26-07-2010	0.038071	2.87		1.2983	87.2	1.549			1.79	77.9	22.73	4.74	5.95	9503.66		1.4	6.54	8.39	3.03	0.29	0.3	0.24	0.24	78.93	4.65	0	0	
27-07-2010	0.034905	2.93		1.2983	87.88	1.5538			1.82	75.52	23.19	4.81	5.98	9496.85		1.44	6.43	8.33	3.08	0.3	0.28		0.15	77.46	4.72	0	0	
28-07-2010	0.031519	2.88		1.2998	87.62	1.5611			1.81	76.66	24.25	4.76	5.95	9753.27		1.46	6.47	8.31	3.03	0.3	0.28	0.24	0.23	77.06	4.75	0	0	
29-07-2010	0.027913	2.88		1.3064	86.87	1.5591			1.82	78.6	24.13	4.8	5.95	9696.02		1.45	6.48	8.3	3.03	0.3	0.35	0.26	0.18	78.3	4.8	0	0	
30-07-2010	0.024085	2.79		1.3069	86.43	1.5714			1.8	77.5	23.5	4.7	5.85	9537.3		1.43	6.57	8.29	2.94	0.29	0.35	0.25	0.21	78.85	4.81	0	0	
02-08-2010	0.011797	2.83		1.3174	86.42	1.5897			1.86	81.93	22.01	4.78	5.89	9570.31		1.5	6.54	8.31	2.99	0.28	0.24	0.29	0.24	81.25	4.94	0	0	
03-08-2010	0.008153	2.78		1.3239	85.88	1.5941			1.87	83.6	22.63	4.73	5.86	9694.01		1.51	6.55	8.26	2.94	0.27	0.32	0.3	0.21	82.52	4.78	0	0	
04-08-2010	0.004822	2.82		1.3158	86.26	1.5887			1.86	83.76	22.21	4.75	5.88	9489.34		1.5	6.49	8.24	2.98	0.28	0.25	0.22	0.21	82.49	4.77	0	0	
05-08-2010	0.001804	2.79		1.3157	85.83	1.586			1.87	82.9	22.1	4.72	5.87	9653.92		1.5	6.51	8.23	2.94	0.27	0.28		0.21	82	4.84	0	0	
06-08-2010	-0.0009	2.71		1.3282	85.25	1.5979			1.82	81.28	21.74	4.65	5.8	9642.12		1.46	6.57	8.21	2.86	0.25	0.32		0.24	80.67	4.67	0	0	
09-08-2010	-0.00734	2.71		1.3241	85.88	1.5945			1.82	81.54	22.14	4.66	5.81	9572.49		1.48	6.54	8.2	2.86	0.26	0.3	0.22	0.23	81.46	4.52	0	0	
10-08-2010	-0.00915	2.64		1.3095	85.88	1.5776			1.84	79.89	22.37	4.68	5.84	9551.05		1.48	6.6	8.22	2.79	0.25	0.3	0.27	0.23	80.24	4.43	0	0	
11-08-2010	-0.01084	2.57		1.2899	85.36	1.5688			1.77	77.83	25.39	4.57	5.75	9292.85		1.43	6.78	8.33	2.72	0.25	0.26		0.21	78.09	4.38	0	0	
12-08-2010	-0.01239	2.58		1.2866	85.83	1.5573			1.69	76.63	25.73	4.58	5.78	9212.59		1.35	6.83	8.42	2.74	0.25	0.25		0.18	75.68	4.42	0	0	
13-08-2010	-0.01381	2.53		1.2767	86.2	1.558			1.68	75.14	26.24	4.52	5.73	9253.46		1.34	6.87	8.43	2.68	0.26	0.25	0.28	0.22	75.39	4.35	0	0	
16-08-2010	-0.01841	2.42		1.2838	85.33	1.5679			1.63	74.56	26.1	4.36	5.58	9196.67		1.29	6.91	8.38	2.58	0.25	0.27	0.27	0.23	75.17	4.37	0	0	
17-08-2010	-0.02156	2.47		1.289	85.51	1.5586			1.63	76.74	24.33	4.47	5.63	9161.68		1.28	6.82	8.34	2.64	0.26	0.28	0.26	0.23	75.76	4.28	0	0	
18-08-2010	-0.02571	2.48		1.2872	85.31	1.5619			1.62	75.1	24.59	4.43	5.58	9240.54		1.28	6.78	8.31	2.64	0.25	0.27	0.24	0.21	75.39	4.35	0	0	

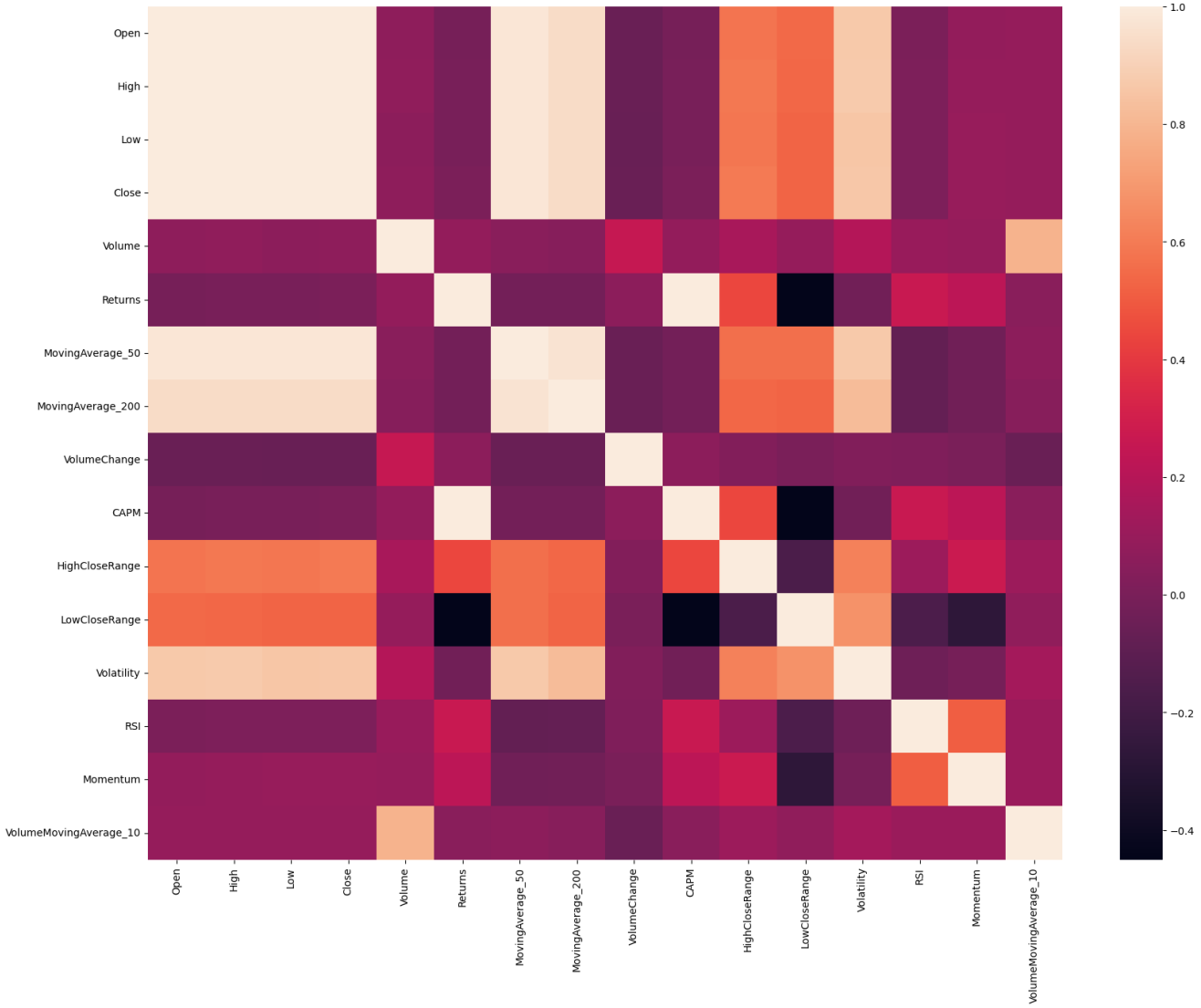
TSLA_Feature_Mart

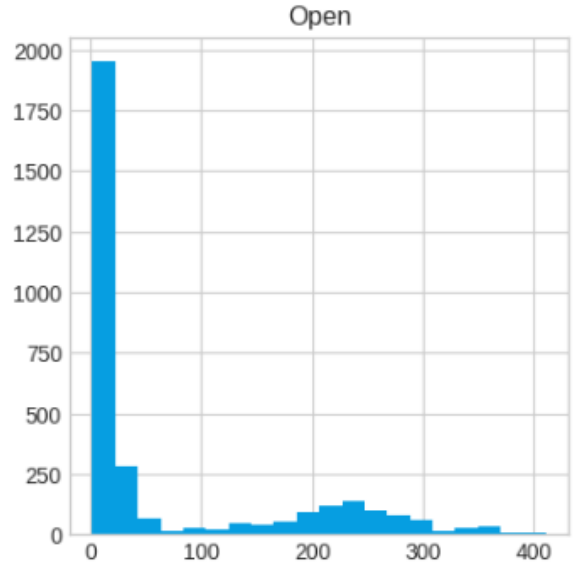
Ready Accessibility: Unavailable 100%

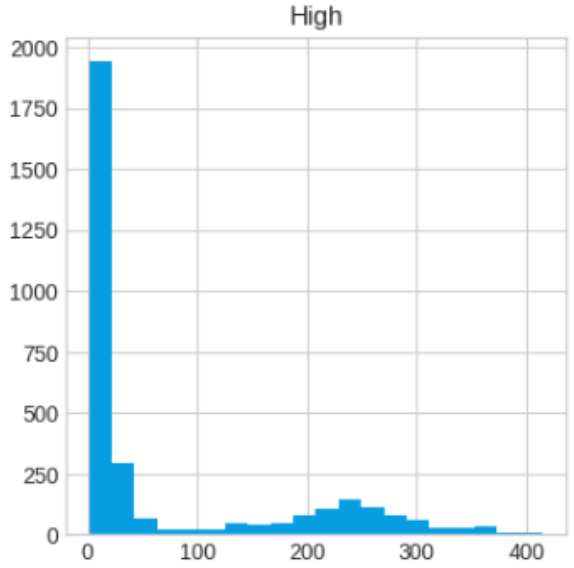
	Low	Close	Volume	Returns	MovingAverage_50	MovingAverage_200	VolumeChange	CAPM	HighCloseRange	LowCloseRange	Volatility	RSI	Momentum	VolumeMovingAverage_10
Low	1.000000	0.999677	0.065395	-0.003758	0.981094	0.942379	-0.059183	-0.003758	0.582226	0.527988	0.856659	0.011313	0.095838	0.089081
Close	0.999677	1.000000	0.069794	0.006237	0.981263	0.942283	-0.056747	0.006237	0.592757	0.525761	0.862790	0.012356	0.098151	0.091477
Volume	0.065395	0.069794	1.000000	0.086469	0.049588	0.037742	0.255410	0.086469	0.154710	0.093337	0.190298	0.102130	0.089080	0.786857
Returns	-0.003758	0.006237	0.086469	1.000000	-0.023327	-0.022065	0.064031	1.000000	0.442781	-0.450123	-0.025592	0.267675	0.221376	0.051940
MovingAverage_50	0.981094	0.981263	0.049588	-0.023327	1.000000	0.967734	-0.057918	-0.023327	0.560946	0.561316	0.867186	-0.076637	-0.035476	0.064102
MovingAverage_200	0.942379	0.942283	0.037742	-0.022065	0.967734	1.000000	-0.058517	-0.022065	0.538863	0.527597	0.823808	-0.073796	-0.027215	0.046895
VolumeChange	-0.059183	-0.056747	0.255410	0.064031	-0.057918	-0.058517	1.000000	0.064031	0.030456	0.000901	0.023571	0.014920	0.002703	-0.055311
CAPM	-0.003758	0.006237	0.086469	1.000000	-0.023327	-0.022065	0.064031	1.000000	0.442781	-0.450123	-0.025592	0.267675	0.221376	0.051940
HighCloseRange	0.582226	0.592757	0.154710	0.442781	0.560946	0.538863	0.030456	0.442781	1.000000	-0.163551	0.620373	0.113541	0.270054	0.112039
LowCloseRange	0.527988	0.525761	0.093337	-0.450123	0.561316	0.527597	0.000901	-0.450123	-0.163551	1.000000	0.672284	-0.157909	-0.271360	0.071572
Volatility	0.856659	0.862790	0.190298	-0.025592	0.867186	0.823808	0.023571	-0.025592	0.620373	0.672284	1.000000	-0.040338	-0.013087	0.140975
RSI	0.011313	0.012356	0.102130	0.267675	-0.076637	-0.073796	0.014920	0.267675	0.113541	-0.157909	-0.040338	1.000000	0.512523	0.106224
Momentum	0.095838	0.098151	0.089080	0.221376	-0.035476	-0.027215	0.002703	0.221376	0.270054	-0.271360	-0.013087	0.512523	1.000000	0.108029
VolumeMovingAverage_10	0.089081	0.091477	0.786857	0.051940	0.064102	0.046895	-0.055311	0.051940	0.112039	0.071572	0.140975	0.106224	0.108029	1.000000

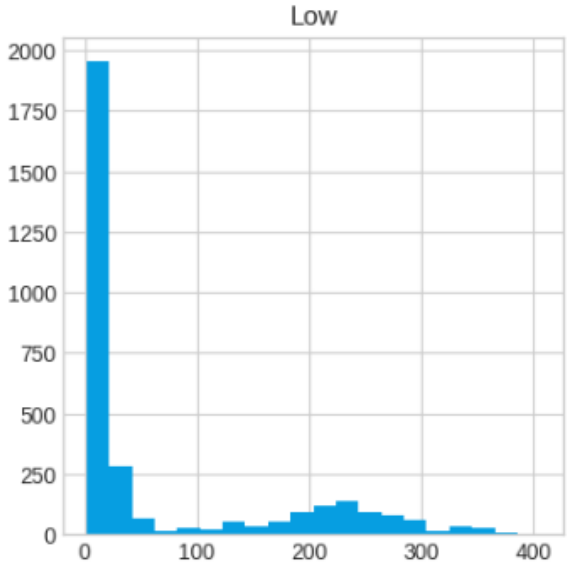
Correlation Matrix of the features of TSLA

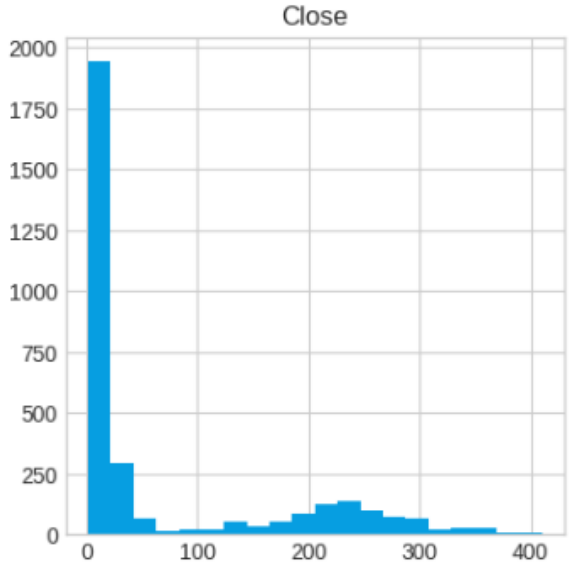
Correlation Matrix of the features of TSLA

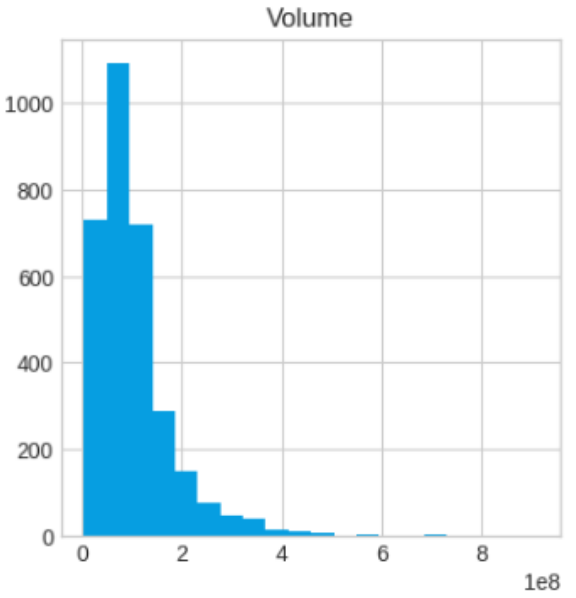


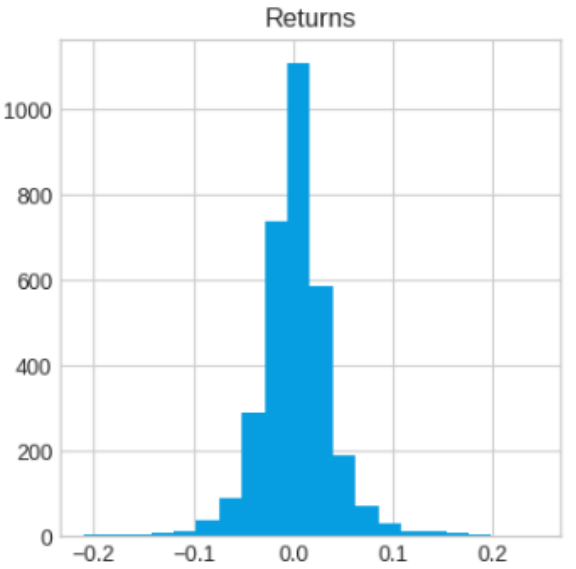

Open
2000
1750
1500
1250
1000
750
500
250
0
0
100
200
300
400

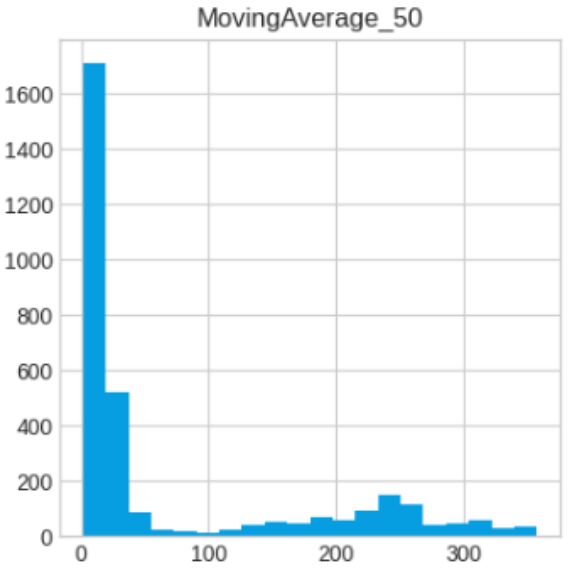

High
2000
1750
1500
1250
1000
750
500
250
0
0
100
200
300
400

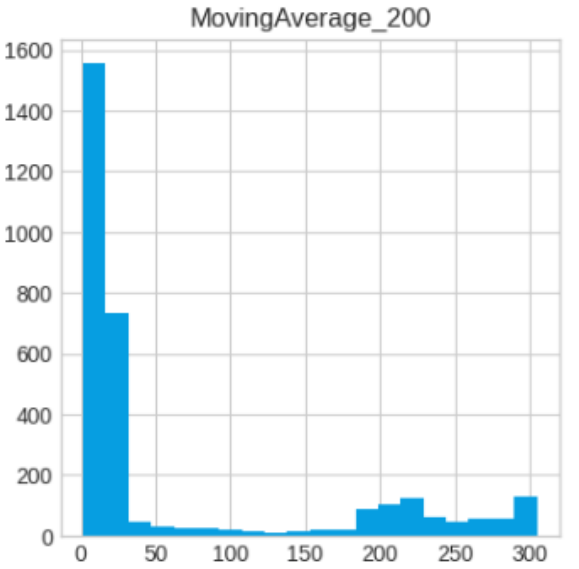

Low
2000
1750
1500
1250
1000
750
500
250
0
0
100
200
300
400

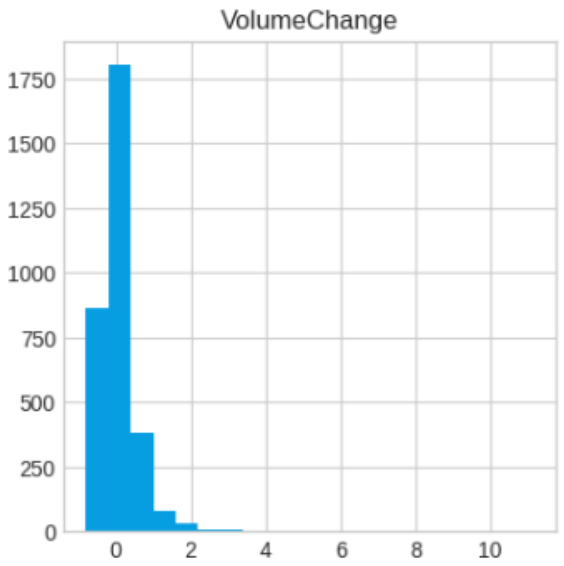

Close
2000
1750
1500
1250
1000
750
500
250
0
0
100
200
300
400

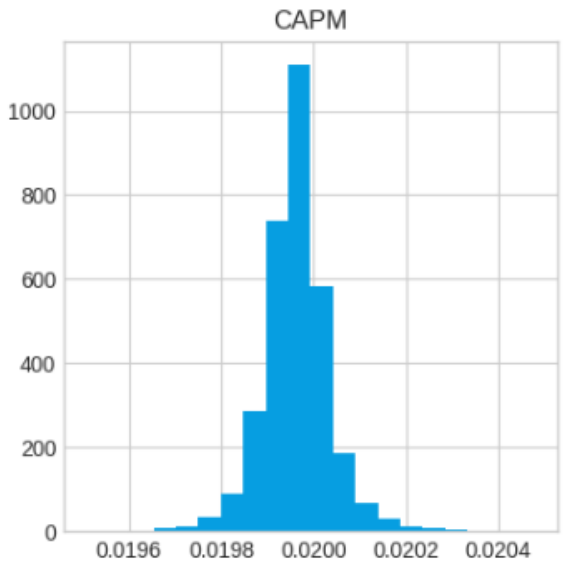

Volume
1000
800
600
400
200
0
0
2
4
6
8
1e8

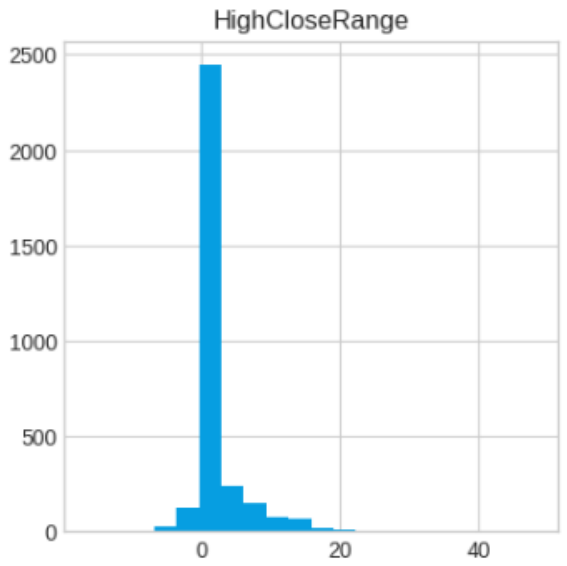

Returns
1000
800
600
400
200
0
−0.2
−0.1
0.0
0.1
0.2

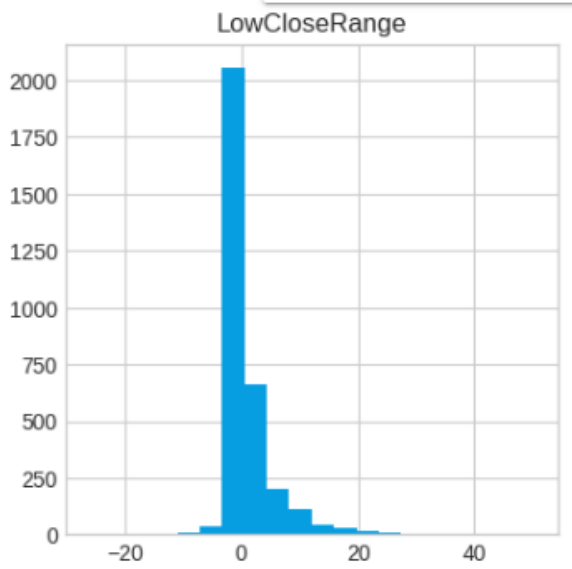

MovingAverage_50
1600
1400
1200
1000
800
600
400
200
0
0
100
200
300

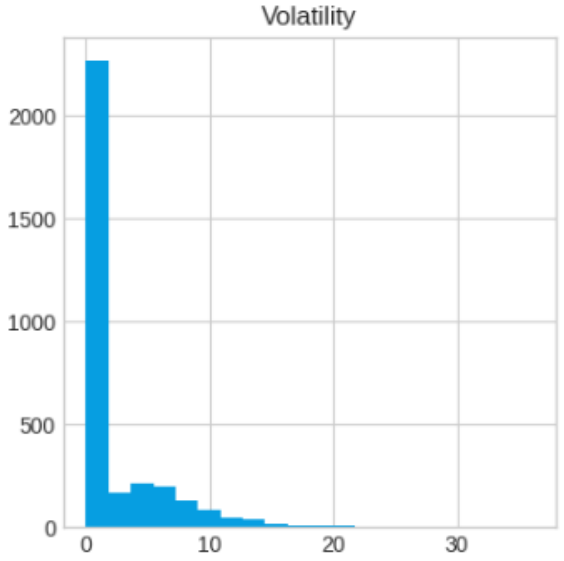

MovingAverage_200
1600
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250
300

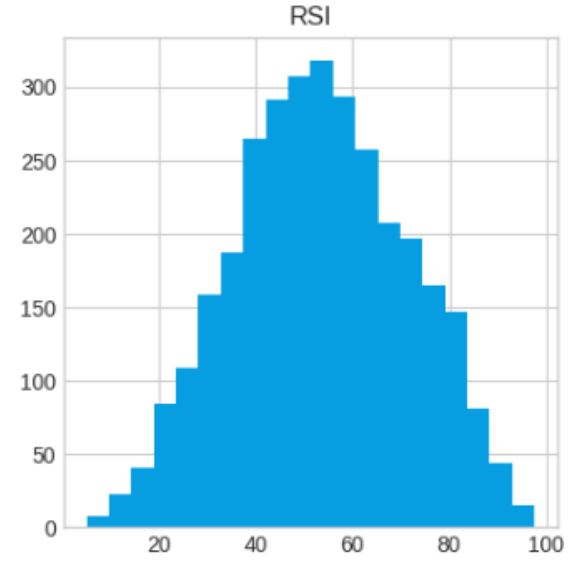

VolumeChange
1750
1500
1250
1000
750
500
250
0
0
2
4
6
8
10

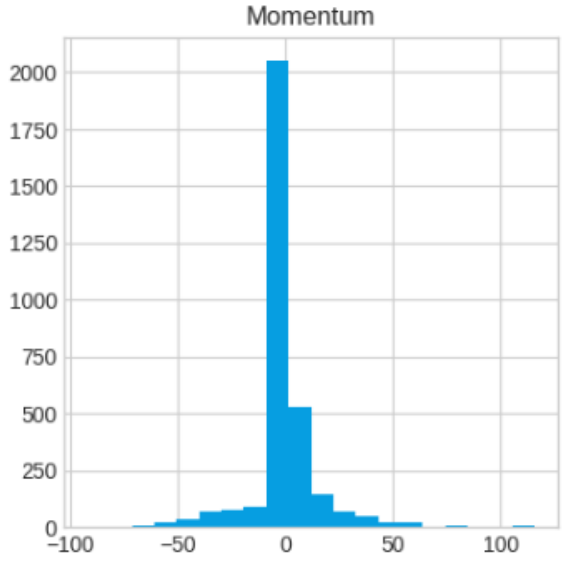

CAPM
1000
800
600
400
200
0
0.0196
0.0198
0.0200
0.0202
0.0204

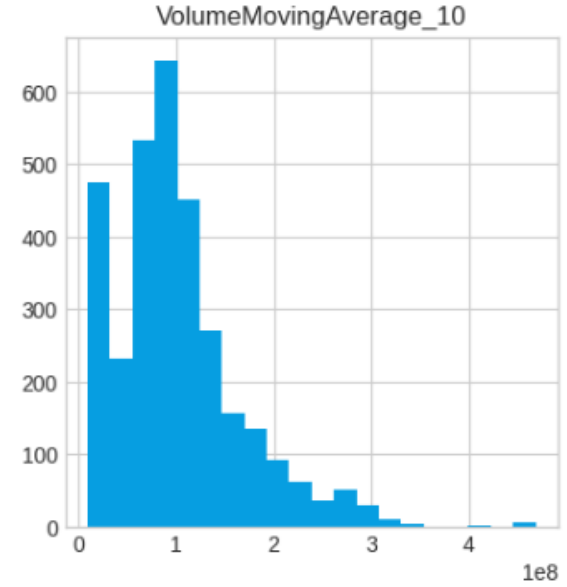

HighCloseRange
2500
2000
1500
1000
500
0
0
20
40

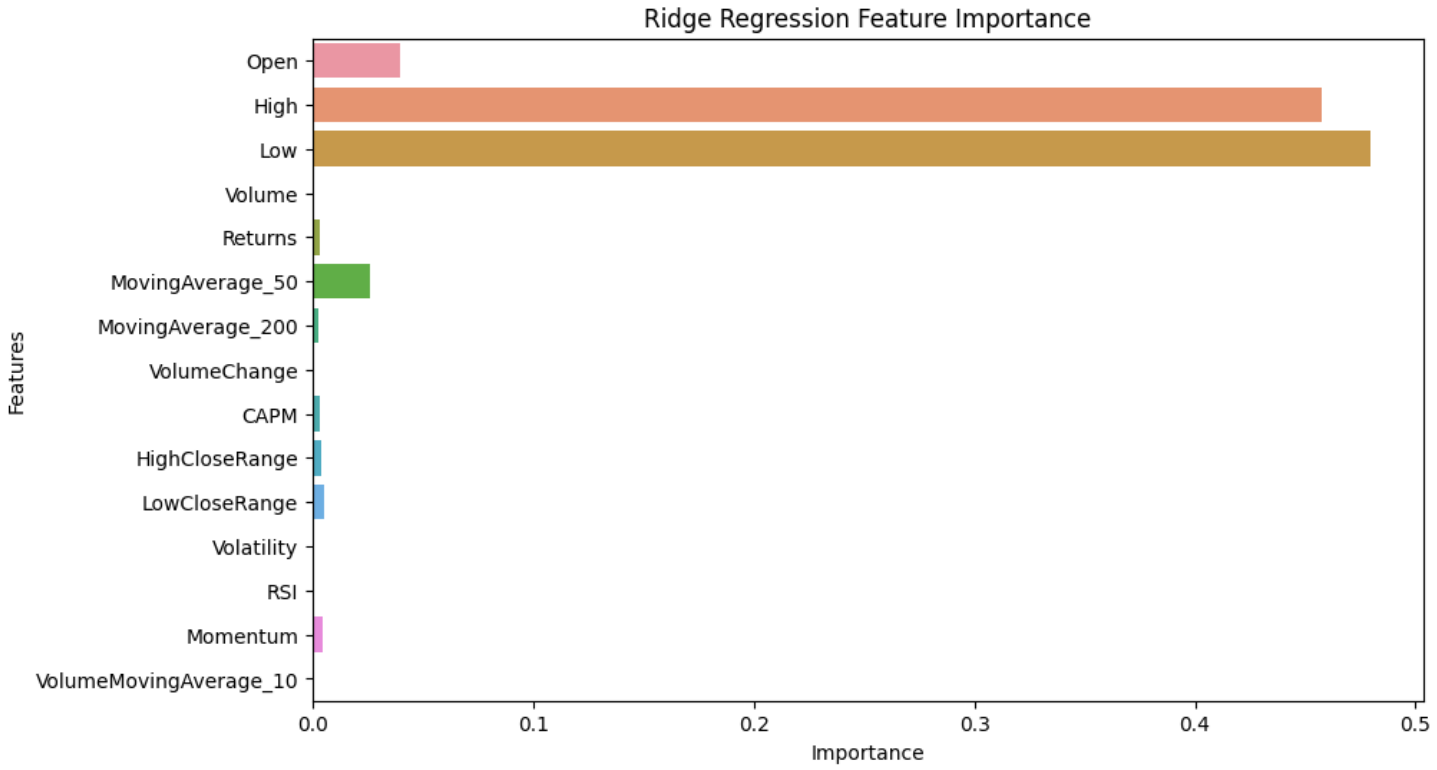

LowCloseRange
2000
1750
1500
1250
1000
750
500
250
0
−20
0
20
40


Volatility
2000
1500
1000
500
0
0
10
20
30

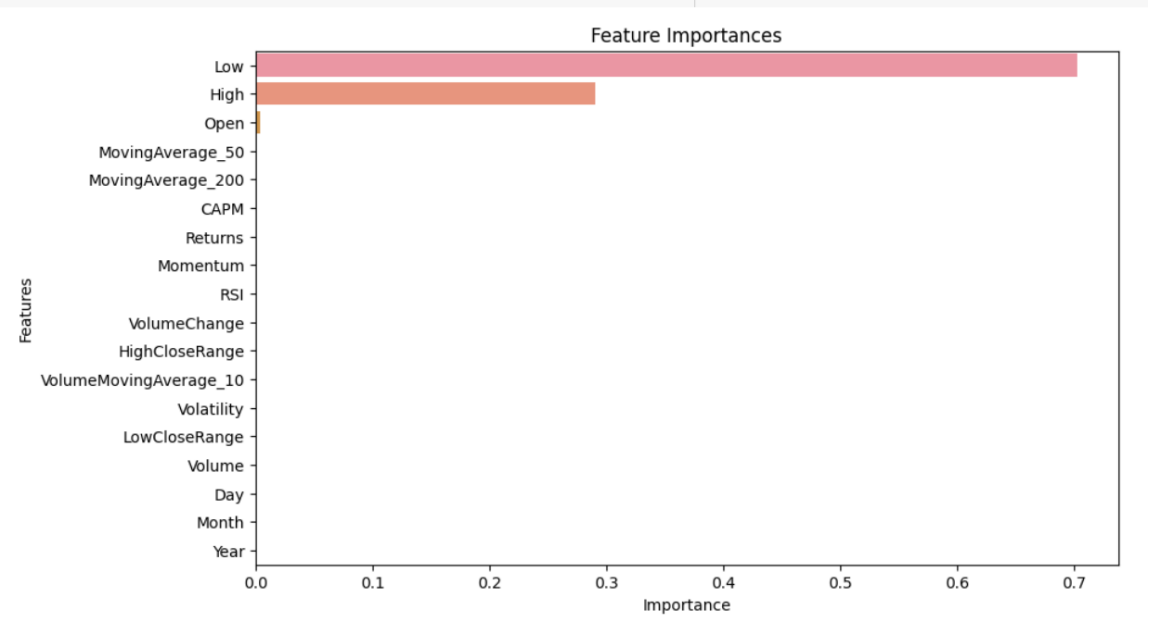

RSI
300
250
200
150
100
50
0
20
40
60
80
100


Momentum
2000
1750
1500
1250
1000
750
500
250
0
−100
−50
0
50
100

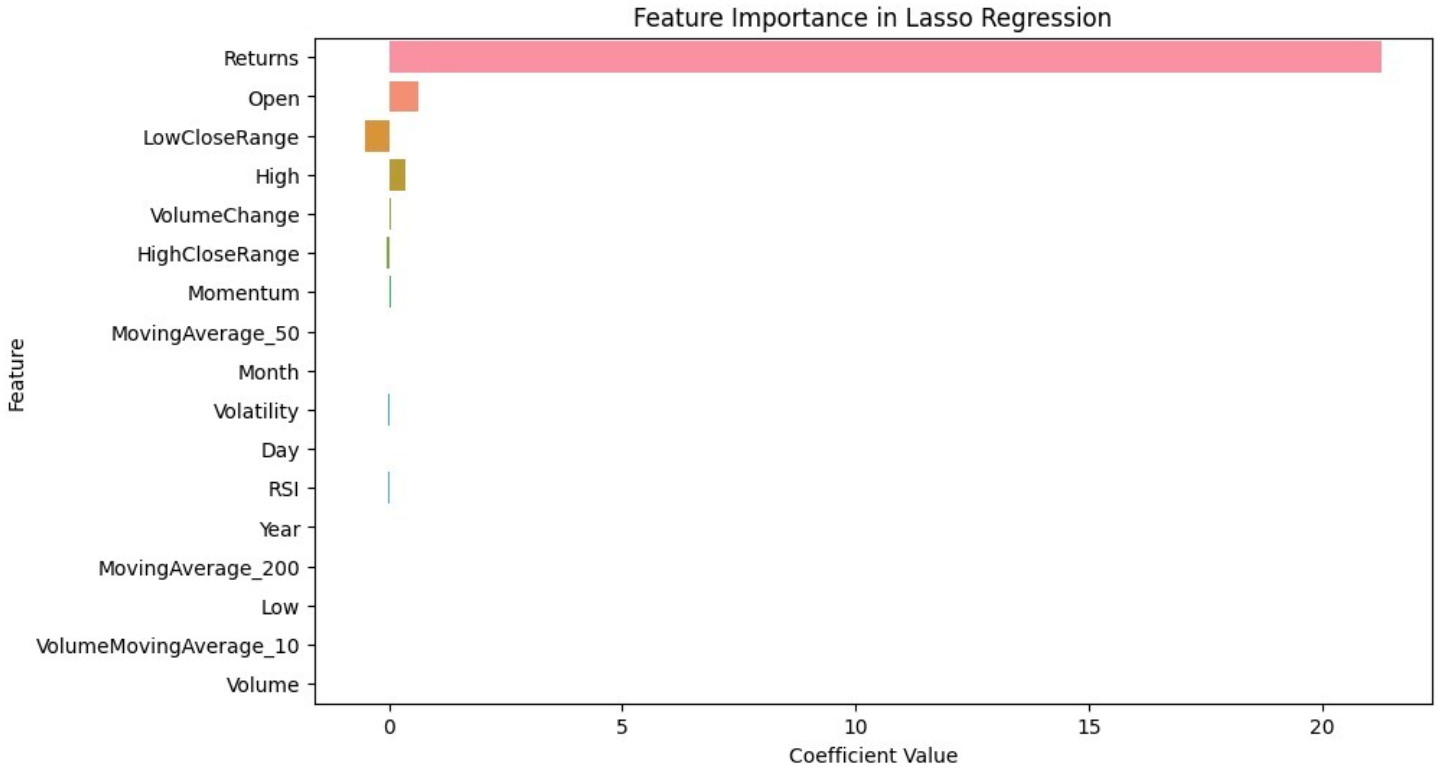

VolumeMovingAverage_10
600
500
400
300
200
100
0
0
1
2
3
4
1e8



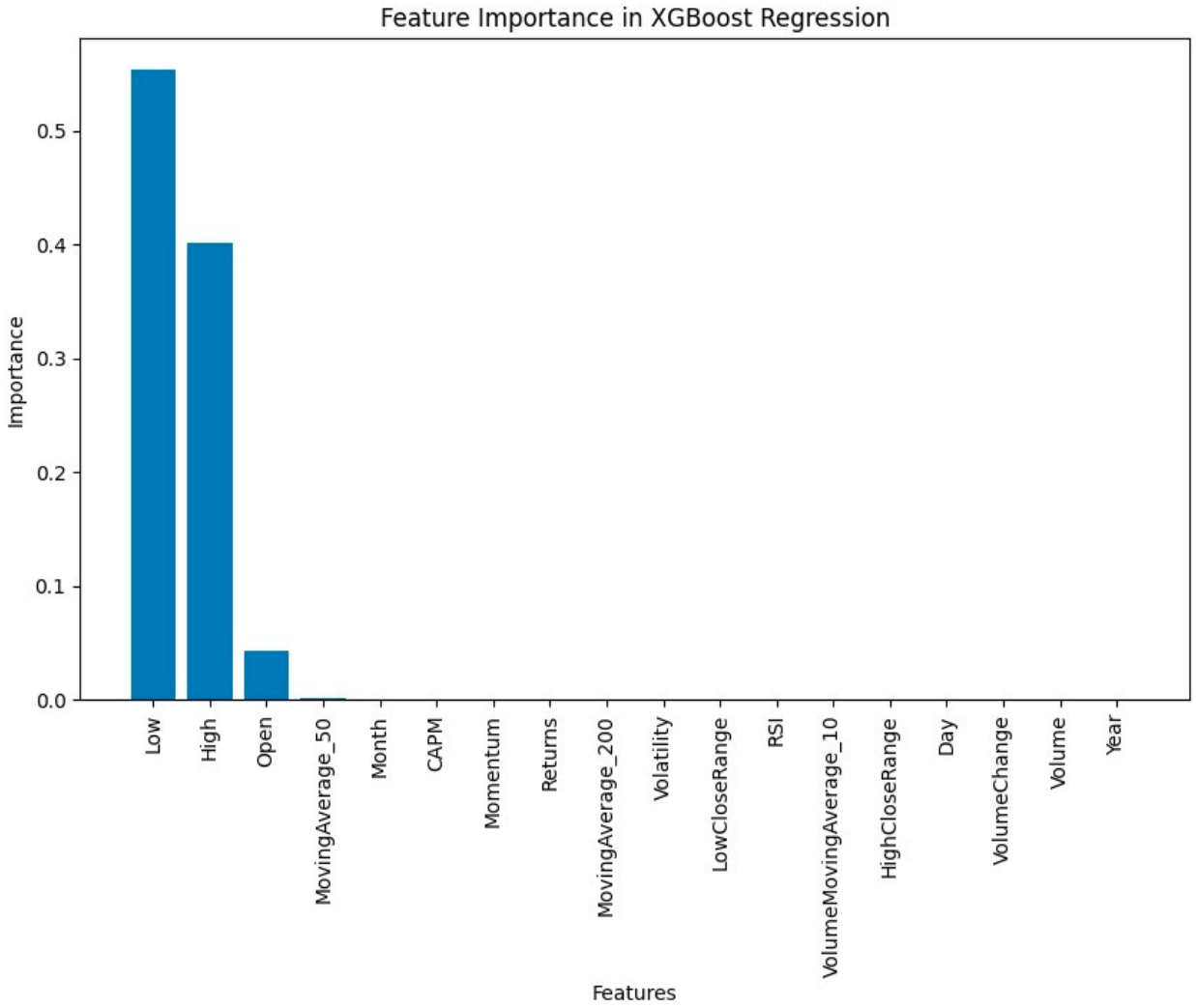
Feature Importance Ridge Regression



Feature Importance Random Forest Regression



Feature Importance LASSO Regression



Feature Importance XGB Regression

OLS TABLE

```
[13] # OLS table for Ridge regression
     X_train_ols = sm.add_constant(X_train)
     ridge_ols = sm.OLS(y_train, X_train_ols).fit()
     print(ridge_ols.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Close   R-squared:                       1.000
Model:                            OLS   Adj. R-squared:                  1.000
Method:                 Least Squares   F-statistic:                 6.671e+05
Date:                Fri, 15 Dec 2023   Prob (F-statistic):               0.00
Time:                        17:03:18   Log-Likelihood:                 6939.7
No. Observations:                2552   AIC:                        -1.385e+04
Df Residuals:                    2536   BIC:                        -1.375e+04
Df Model:                          15
Covariance Type:            nonrobust
================================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
const                0.0002      0.000      0.544      0.586      -0.000       0.001
Open                -0.5816      0.019    -30.236      0.000      -0.619      -0.544
High              1.901e+07   6.71e+06      2.833      0.005    5.85e+06    3.22e+07
Low              -1.813e+07    6.4e+06     -2.833      0.005   -3.07e+07   -5.58e+06
Volume              -0.0002      0.001     -0.328      0.743      -0.001       0.001
Returns            134.3550     84.256      1.595      0.111     -30.863     299.573
MovingAverage_50     0.0111      0.003      3.840      0.000       0.005       0.017
MovingAverage_200   -0.0026      0.001     -2.011      0.044      -0.005   -6.54e-05
VolumeChange         0.0003      0.000      0.746      0.456      -0.000       0.001
CAPM              -134.3494     84.256     -1.595      0.111    -299.568      30.869
HighCloseRange    -1.662e+06   2.12e+06     -0.783      0.433   -5.82e+06    2.5e+06
LowCloseRange     -1.761e+06   2.25e+06     -0.783      0.433   -6.17e+06    2.65e+06
Volatility          1.2e+06   2.81e+06      0.427      0.670   -4.31e+06    6.71e+06
RSI                 -0.0010      0.000     -2.621      0.009      -0.002      -0.000
Momentum             0.0025      0.000      5.248      0.000       0.002       0.003
VolumeMovingAverage_10     0.0004      0.001      0.696      0.487      -0.001       0.002
==============================================================================
Omnibus:                      550.007   Durbin-Watson:                   2.021
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            20408.162
Skew:                           0.150   Prob(JB):                         0.00
Kurtosis:                      16.850   Cond. No.                     7.43e+10
==============================================================================
```

```
# OLS table for LASSO
X_train_ols = sm.add_constant(X_train)
lasso_ols = sm.OLS(y_train, X_train_ols).fit()
print(lasso_ols.summary())
```

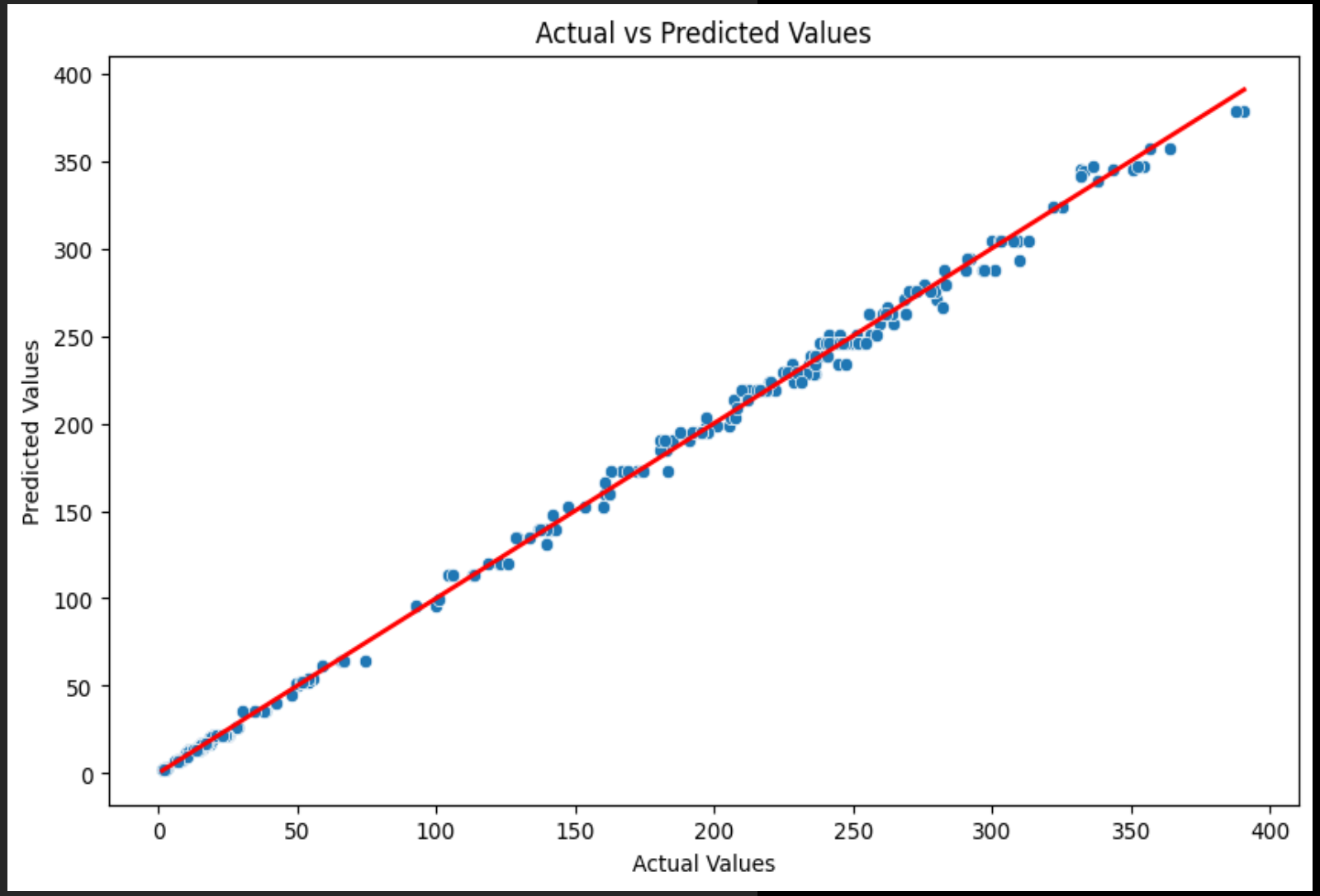
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Close   R-squared:                       1.000
Model:                            OLS   Adj. R-squared:                  1.000
Method:                 Least Squares   F-statistic:                 6.671e+05
Date:                Fri, 15 Dec 2023   Prob (F-statistic):               0.00
Time:                        17:03:21   Log-Likelihood:                 6939.7
No. Observations:                2552   AIC:                        -1.385e+04
Df Residuals:                    2536   BIC:                        -1.375e+04
Df Model:                          15
Covariance Type:            nonrobust
================================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
const                0.0002      0.000      0.544      0.586      -0.000       0.001
Open                -0.5816      0.019    -30.236      0.000      -0.619      -0.544
High              1.901e+07   6.71e+06      2.833      0.005    5.85e+06    3.22e+07
Low              -1.813e+07    6.4e+06     -2.833      0.005   -3.07e+07   -5.58e+06
Volume              -0.0002      0.001     -0.328      0.743      -0.001       0.001
Returns            134.3550     84.256      1.595      0.111     -30.863     299.573
MovingAverage_50     0.0111      0.003      3.840      0.000       0.005       0.017
MovingAverage_200   -0.0026      0.001     -2.011      0.044      -0.005   -6.54e-05
VolumeChange         0.0003      0.000      0.746      0.456      -0.000       0.001
CAPM              -134.3494     84.256     -1.595      0.111    -299.568      30.869
HighCloseRange    -1.662e+06   2.12e+06     -0.783      0.433   -5.82e+06    2.5e+06
LowCloseRange     -1.761e+06   2.25e+06     -0.783      0.433   -6.17e+06    2.65e+06
Volatility          1.2e+06   2.81e+06      0.427      0.670   -4.31e+06    6.71e+06
RSI                 -0.0010      0.000     -2.621      0.009      -0.002      -0.000
Momentum             0.0025      0.000      5.248      0.000       0.002       0.003
VolumeMovingAverage_10     0.0004      0.001      0.696      0.487      -0.001       0.002
==============================================================================
Omnibus:                      550.007   Durbin-Watson:                   2.021
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            20408.162
Skew:                           0.150   Prob(JB):                         0.00
Kurtosis:                      16.850   Cond. No.                     7.43e+10
==============================================================================
```

MODEL TRAINING

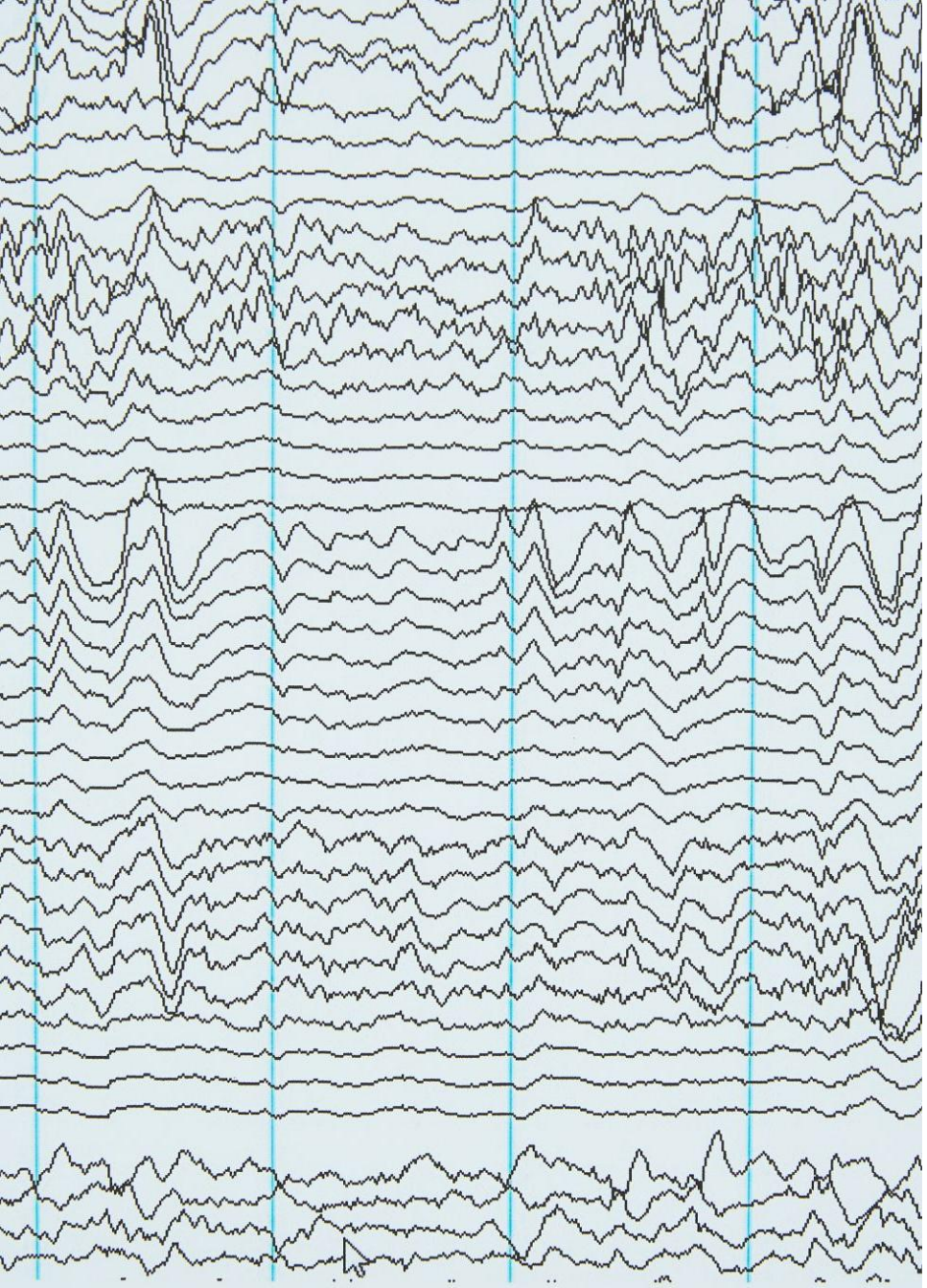
Random Forest Regressor

- The code starts by loading a dataset (TSLA_Feature_Mart_IMP.csv) and performing basic preprocessing steps. It converts the 'Date' column to datetime, extracts year, month, and day as separate features, and drops the original 'Date' column. The target variable (y) is defined as the 'Close' column, and the feature matrix (X) is constructed by excluding the 'Close' column.
- The data is split into training and testing sets using train_test_split. 80% of the data is used for training (X_train, y_train), and 20% is reserved for testing (X_test, y_test).
- A RandomizedSearchCV object is created to perform hyperparameter tuning for a Random Forest Regressor. The hyperparameters considered include the number of estimators, maximum depth of trees, minimum samples required to split an internal node, and minimum samples required at a leaf node. The search is configured to randomly sample 10 combinations from the specified parameter space, using 3-fold cross-validation.
- The RandomizedSearchCV object is fitted to the training data, searching for the best hyperparameters. The best hyperparameters and the corresponding best model are printed. The best model is then used to make predictions on the test data (X_test), and the root mean squared error (RMSE) is calculated to evaluate the model's performance.

```
# Create the grid search object
# Use RandomizedSearchCV
random_search = RandomizedSearchCV(
    estimator=RandomForestRegressor(random_state=42),
    param_distributions=param_grid,
    n_iter=10,  # Number of parameter settings sampled
    cv=3,  # Reduced number of cross-validation folds
    verbose=2,
    random_state=42,
    n_jobs=-1
)

# Fit the grid search to the data
random_search.fit(X_train, y_train)

# Best parameters
print("Best parameters found: ", random_search.best_params_)

# Best model
best_model = random_search.best_estimator_

# Predict and evaluate using the best model
y_pred = best_model.predict(X_test)
rmse = math.sqrt(mean_squared_error(y_test, y_pred))

print(f"Optimized Random Forest Regressor RMSE: {rmse}")
```

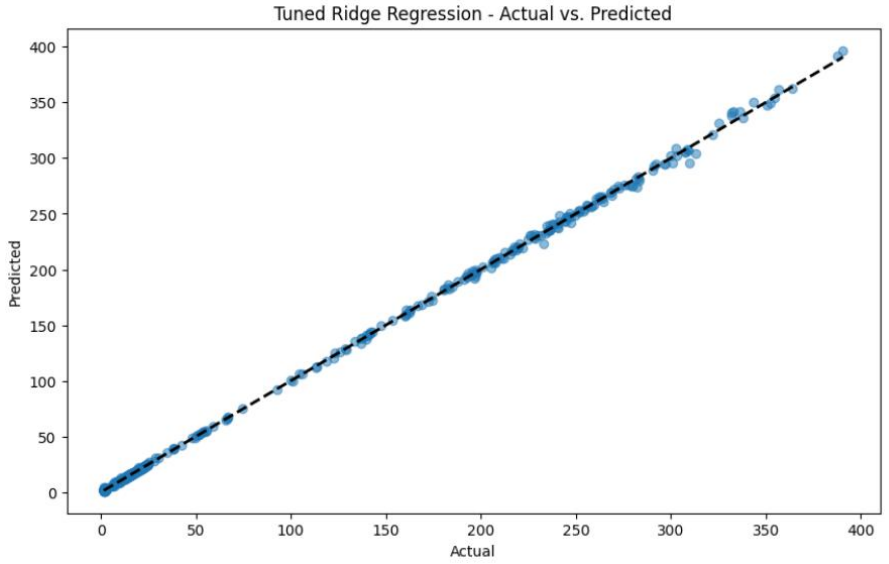



```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
Best parameters found:  {'n_estimators': 50, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_depth': 10}
Optimized Random Forest Regressor RMSE: 2.1152895548092325
```



Ridge Regression

- Ridge regression, a linear regression variant with L2 regularization, to model the relationship between the features (X_train) and the target variable (y_train). Ridge introduces a penalty term (controlled by the hyperparameter alpha) to prevent overfitting.
- A grid search is performed to find the optimal value for the hyperparameter alpha. The range of alpha values to explore is defined using np.logspace(-4, 4, 20), which spans a range of values from 0.0001 to 10000 in a logarithmic scale. The grid search uses 5-fold cross-validation and employs the negative mean squared error as the scoring metric.
- The best alpha value for Ridge is identified from the grid search results (best_alpha). A new Ridge model (best_ridge_model) is then created with the optimal alpha and trained on the training data (X_train, y_train).
- The trained Ridge model is used to make predictions (y_pred_best_ridge) on the test data (X_test). These predictions can be evaluated to assess the model's performance on unseen data.



Ridge Regression

```
rmse_best_ridge = np.sqrt(mean_squared_error(y_test, y_pred_best_ridge))
print(f'Tuned Ridge Regression RMSE: {rmse_best_ridge}')
```

```
Tuned Ridge Regression RMSE: 1.7144318378140322
```

LASSO (Least Absolute Shrinkage and Selection Operator) Regression

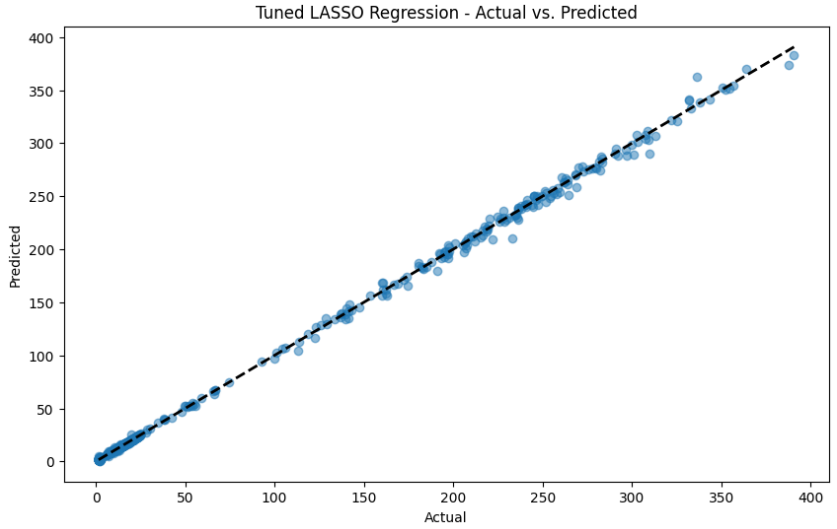
Lasso regression, a linear regression variant with L1 regularization, to model the relationship between the features (X_train) and the target variable (y_train). Lasso introduces a penalty term (controlled by the hyperparameter alpha) to encourage sparsity in the coefficient estimates

A grid search is performed to find the optimal value for the hyperparameter alpha. The range of alpha values to explore is defined using np.logspace(-4, 4, 20), which spans a range of values from 0.0001 to 10000 in a logarithmic scale. The grid search uses 5-fold cross-validation and employs the negative mean squared error as the scoring metric.

The best alpha value for Lasso is identified from the grid search results (best_alpha_lasso). A new Lasso model (best_lasso_model) is then created with the optimal alpha and trained on the training data (X_train, y_train).

The trained Lasso model is used to make predictions (y_pred_best_lasso) on the test data (X_test). These predictions can be evaluated to assess the model's performance on unseen data.

Tuned Lasso Regression



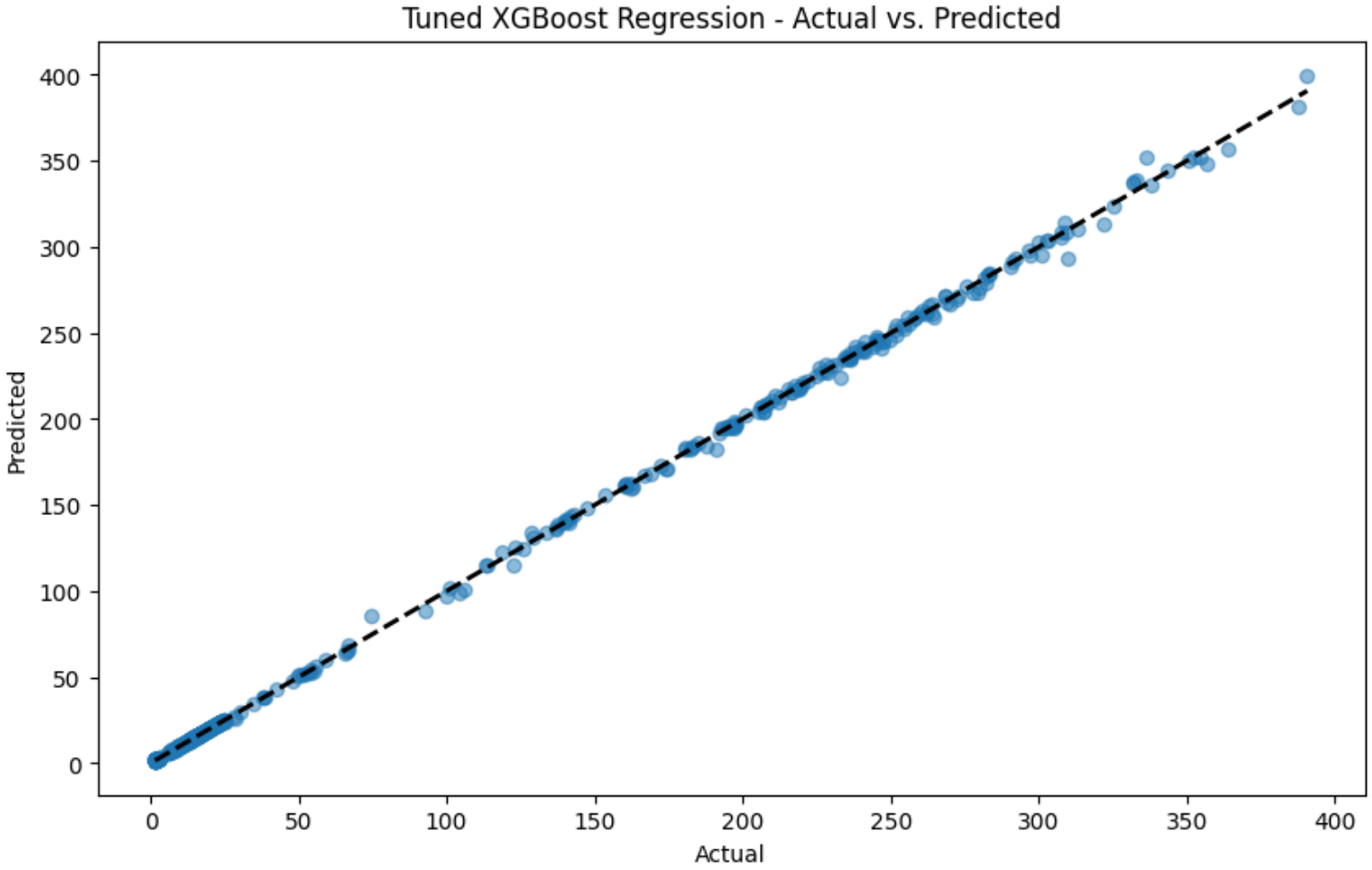

```
rmse_best_lasso = np.sqrt(mean_squared_error(y_test, y_pred_best_lasso))
print(f'Tuned LASSO Regression RMSE: {rmse_best_lasso}')
```

```
Tuned LASSO Regression RMSE: 2.679694189533512
```

Extreme Gradient Boosting (XGBoost) Regressors

- A RandomizedSearchCV object is created with the XGBoost regressor as the estimator. The search space for hyperparameters is defined, specifying potential values for each hyperparameter. The search is configured to randomly sample 10 combinations from this space and perform 3-fold cross-validation.
- The RandomizedSearchCV object is fitted to the training data (X_train and y_train). This involves training and evaluating XGBoost models with different hyperparameter combinations to find the set of hyperparameters that yields the best performance based on cross-validation scores.
- The best hyperparameters found by the randomized search are printed. Additionally, the best XGBoost model (with the optimal hyperparameters) is stored in the variable best_xgb_model. This model can then be used for making predictions on new data.

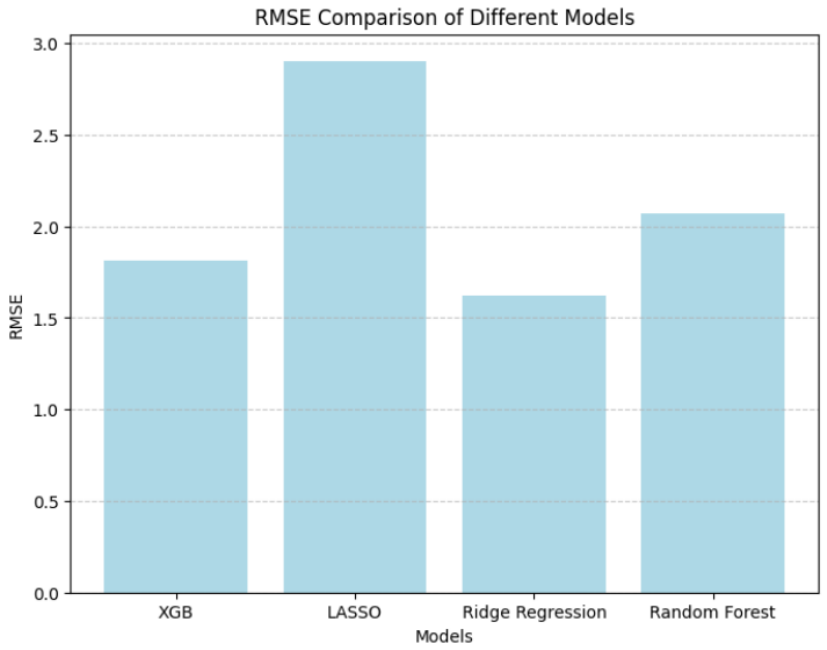
XGBoost Regressors




```
y_pred_best_xgb = best_xgb_model.predict(X_test)
rmse_best_xgb = np.sqrt(mean_squared_error(y_test, y_pred_best_xgb))
print(f'Tuned XGBoost Regression RMSE: {rmse_best_xgb}')
```

```
Tuned XGBoost Regression RMSE: 1.7311977348345764
```

RMSE Comparison of different models

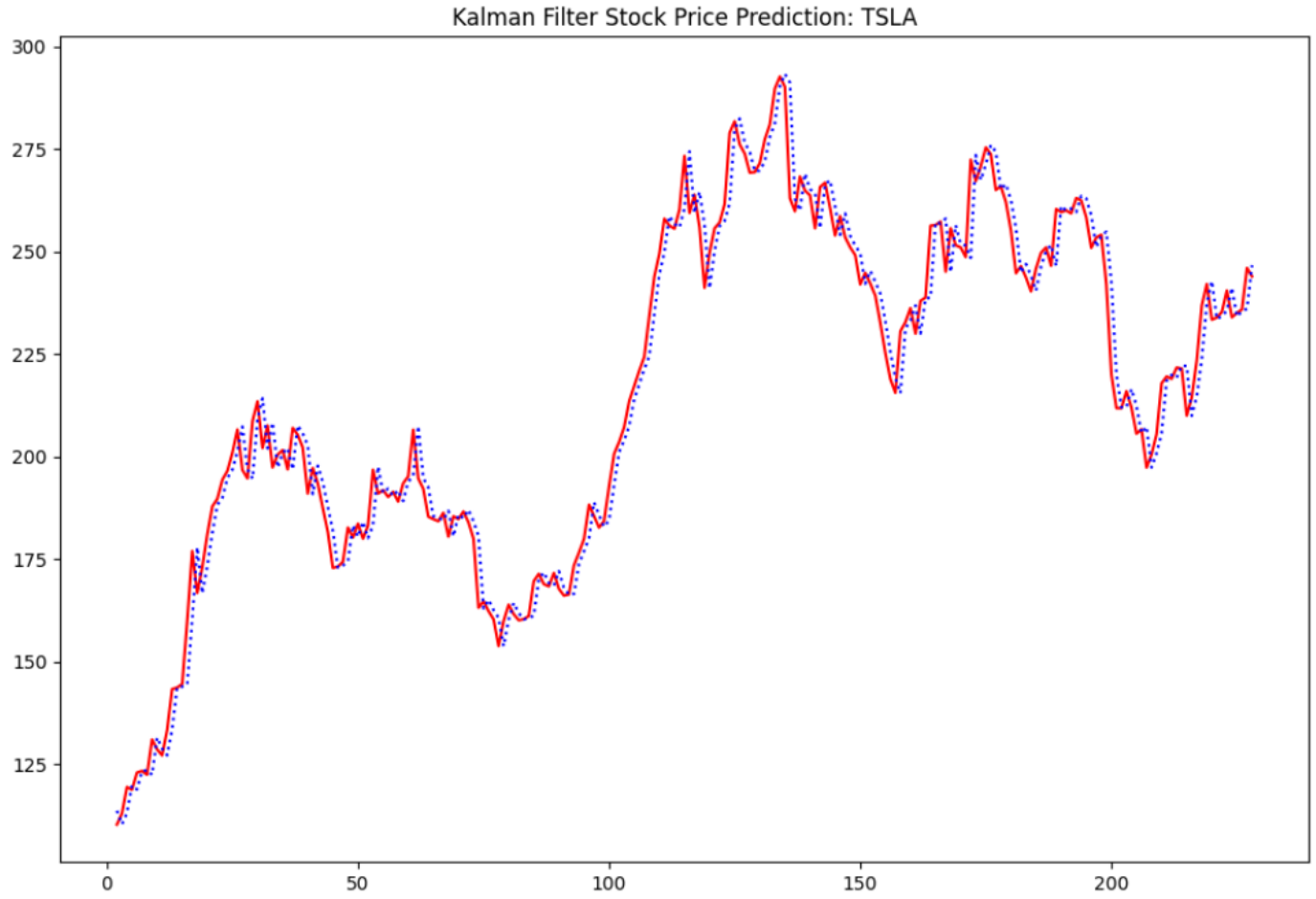


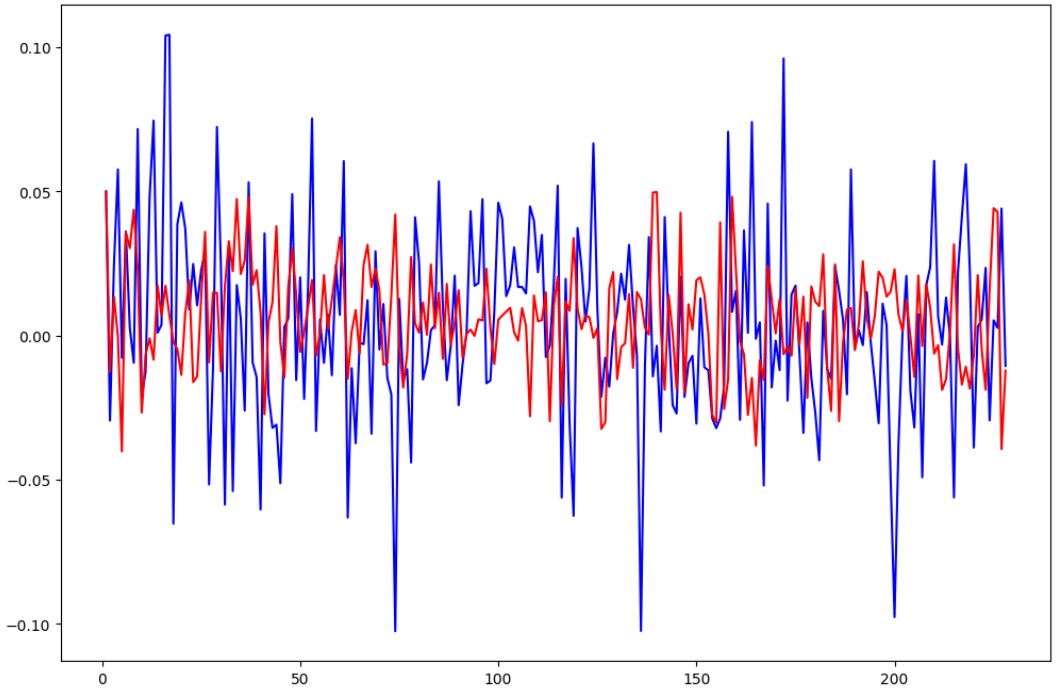

```
XGB RMSE: 1.8124813628535832
LASSO RMSE: 2.9045378379956985
Ridge Regression RMSE: 1.6240902104196515
Random Forest Regression RMSE: 2.06878456471084
```

Kalman filter

- Tesla's stock data from a specified date range and extracts the adjusted closing prices.
- It initializes parameters for the Kalman Filter and

  optimizes them
  to minimize the output of the Kalman_Filter function.
- The optimized parameters are then used in the Kalman_Smoother function to generate smoothed estimates of the stock price.
- Plotting the original and smoothed stock prices, and calculates the root mean square error (RMSE) between the original and smoothed prices.





GARCH

The model provide a framework for capturing and modeling the time-varying volatility observed in financial time series data, making them valuable tools for risk management and financial analysis.