In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import plotly.express as px
import plotly.graph_objects as go
import plotly.io as pio
pio.templates.default = "plotly_white"
```

In [2]:
```
df = pd.read_csv('tsla_2014_2023.csv')
```

In [3]:
```
df.head()
```

Out[3]:

| | date | open | high | low | close | volume | rsi_7 | rsi_14 | cci_7 | cci_14 | sma_50 | ema_50 | sma_100 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2014-01-02 | 9.986667 | 10.165333 | 9.770000 | 10.006667 | 92826000 | 55.344071 | 54.440118 | -37.373644 | 15.213422 | 9.682107 | 9.820167 | 10.494240 |
| **1** | 2014-01-03 | 10.000000 | 10.146000 | 9.906667 | 9.970667 | 70425000 | 53.742629 | 53.821521 | -81.304471 | 17.481130 | 9.652800 | 9.826069 | 10.495693 |
| **2** | 2014-01-06 | 10.000000 | 10.026667 | 9.682667 | 9.800000 | 80416500 | 46.328174 | 50.870410 | -123.427544 | -37.824708 | 9.629467 | 9.825047 | 10.496740 |
| **3** | 2014-01-07 | 9.841333 | 10.026667 | 9.683333 | 9.957333 | 75511500 | 53.263037 | 53.406750 | -84.784651 | -20.779431 | 9.597747 | 9.830235 | 10.503407 |
| **4** | 2014-01-08 | 9.923333 | 10.246667 | 9.917333 | 10.085333 | 92448000 | 58.368660 | 55.423026 | 60.799662 | 43.570559 | 9.573240 | 9.840239 | 10.511147 |

In [4]:
```
df.tail()
```

Out[4]:

| | date | open | high | low | close | volume | rsi_7 | rsi_14 | cci_7 | cci_14 | sma_50 | ema_ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **2511** | 2023-12-22 | 256.760010 | 258.220001 | 251.369995 | 252.539993 | 93249800 | 58.296612 | 58.137456 | 50.821325 | 80.672033 | 232.553000 | 240.3605 |
| **2512** | 2023-12-26 | 254.490005 | 257.970001 | 252.910004 | 256.609985 | 86892400 | 63.570549 | 60.824035 | 93.909968 | 86.446838 | 232.662800 | 240.9978 |
| **2513** | 2023-12-27 | 258.350006 | 263.339996 | 257.519989 | 261.440002 | 106494400 | 68.998630 | 63.793639 | 171.938770 | 119.554558 | 232.813200 | 241.7994 |
| **2514** | 2023-12-28 | 263.660004 | 265.130005 | 252.710007 | 253.179993 | 113619900 | 53.186966 | 55.978816 | 45.772983 | 73.958135 | 232.779799 | 242.2457 |
| **2515** | 2023-12-29 | 255.100006 | 255.190002 | 247.429993 | 248.479996 | 100615300 | 46.164227 | 52.070118 | -98.880167 | -0.320098 | 232.895800 | 242.4902 |

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
date              0
open              0
high              0
low               0
close             0
volume            0
rsi_7             0
rsi_14            0
cci_7             0
cci_14            0
sma_50            0
ema_50            0
sma_100           0
ema_100           0
macd              0
bollinger         0
TrueRange         0
atr_7             0
atr_14            0
next_day_close    0
dtype: int64
```

In [6]:
```
df.describe()
```

In [6]:
```
df.describe()
```

Out[6]:

|  | open | high | low | close | volume | rsi_7 | rsi_14 | cci_7 | cci_14 | sma_50 |
|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 2516.000000 | 2516.000000 | 2516.000000 | 2516.000000 | 2.516000e+03 | 2516.000000 | 2516.000000 | 2516.000000 | 2516.000000 | 2516.000000 |
| **mean** | 94.098510 | 96.172733 | 91.865096 | 94.072491 | 1.131986e+08 | 53.058382 | 52.862457 | 9.809933 | 13.202457 | 91.810735 |
| **std** | 108.593936 | 111.022486 | 105.911918 | 108.500301 | 7.547433e+07 | 18.239752 | 13.352063 | 100.975002 | 109.285239 | 106.581797 |
| **min** | 9.366667 | 9.800000 | 9.111333 | 9.289333 | 1.062000e+07 | 6.395305 | 16.564126 | -233.333333 | -297.930166 | 9.490973 |
| **25%** | 15.763167 | 16.082168 | 15.491167 | 15.814167 | 6.643185e+07 | 39.859440 | 43.595435 | -76.876737 | -78.543937 | 15.496080 |
| **50%** | 21.801001 | 22.198334 | 21.487666 | 21.877667 | 9.320775e+07 | 53.226417 | 51.621434 | 19.823624 | 24.702835 | 21.563733 |
| **75%** | 200.017505 | 204.525829 | 194.482498 | 200.049999 | 1.323710e+08 | 65.900330 | 61.937068 | 94.426550 | 99.180514 | 192.341650 |
| **max** | 411.470001 | 414.496674 | 405.666656 | 409.970001 | 9.140820e+08 | 97.460910 | 94.197983 | 233.333333 | 350.643337 | 357.870532 |

In [7]:
```
df.shape
```

Out[7]:
```
(2516, 20)
```

In [8]:
```
df.dtypes
```

Out[8]:
```
date                  object
open                 float64
high                 float64
low                  float64
close                float64
volume                 int64
rsi_7                float64
rsi_14               float64
cci_7                float64
cci_14               float64
sma_50               float64
ema_50               float64
sma_100              float64
ema_100              float64
macd                 float64
bollinger            float64
TrueRange            float64
atr_7                float64
atr_14               float64
next_day_close       float64
dtype: object
```

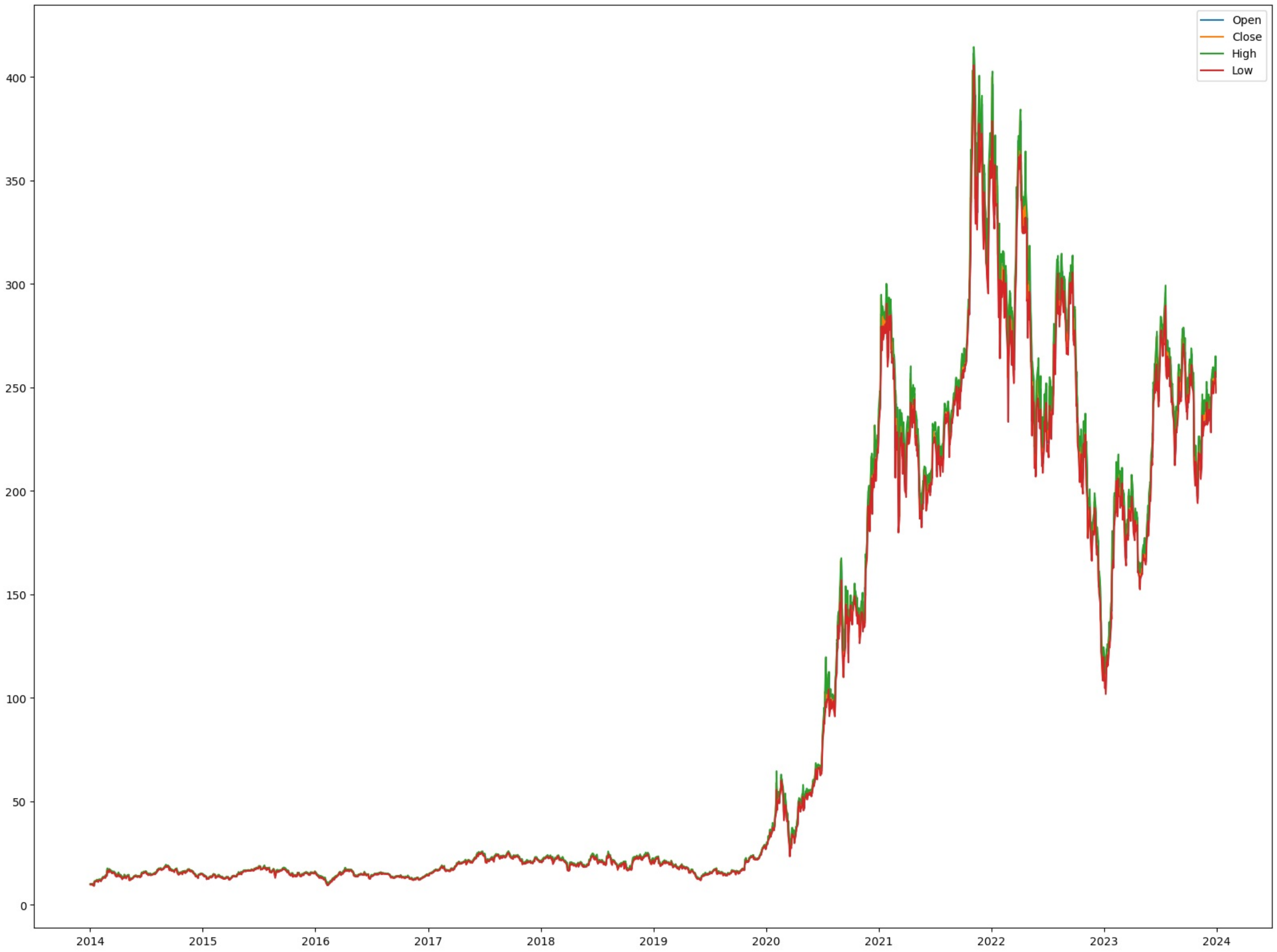
In [9]:
```
df['date'] = pd.to_datetime(df['date'])
df.set_index('date', inplace=True)

df.head()
```

Out[9]:

|  | open | high | low | close | volume | rsi_7 | rsi_14 | cci_7 | cci_14 | sma_50 | ema_50 | sma_100 | er |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **date** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **2014-01-02** | 9.986667 | 10.165333 | 9.770000 | 10.006667 | 92826000 | 55.344071 | 54.440118 | -37.373644 | 15.213422 | 9.682107 | 9.820167 | 10.494240 | 9. |
| **2014-01-03** | 10.000000 | 10.146000 | 9.906667 | 9.970667 | 70425000 | 53.742629 | 53.821521 | -81.304471 | 17.481130 | 9.652800 | 9.826069 | 10.495693 | 9. |
| **2014-01-06** | 10.000000 | 10.026667 | 9.682667 | 9.800000 | 80416500 | 46.328174 | 50.870410 | -123.427544 | -37.824708 | 9.629467 | 9.825047 | 10.496740 | 9. |
| **2014-01-07** | 9.841333 | 10.026667 | 9.683333 | 9.957333 | 75511500 | 53.263037 | 53.406750 | -84.784651 | -20.779431 | 9.597747 | 9.830235 | 10.503407 | 9. |
| **2014-01-08** | 9.923333 | 10.246667 | 9.917333 | 10.085333 | 92448000 | 58.368660 | 55.423026 | 60.799662 | 43.570559 | 9.573240 | 9.840239 | 10.511147 | 9. |

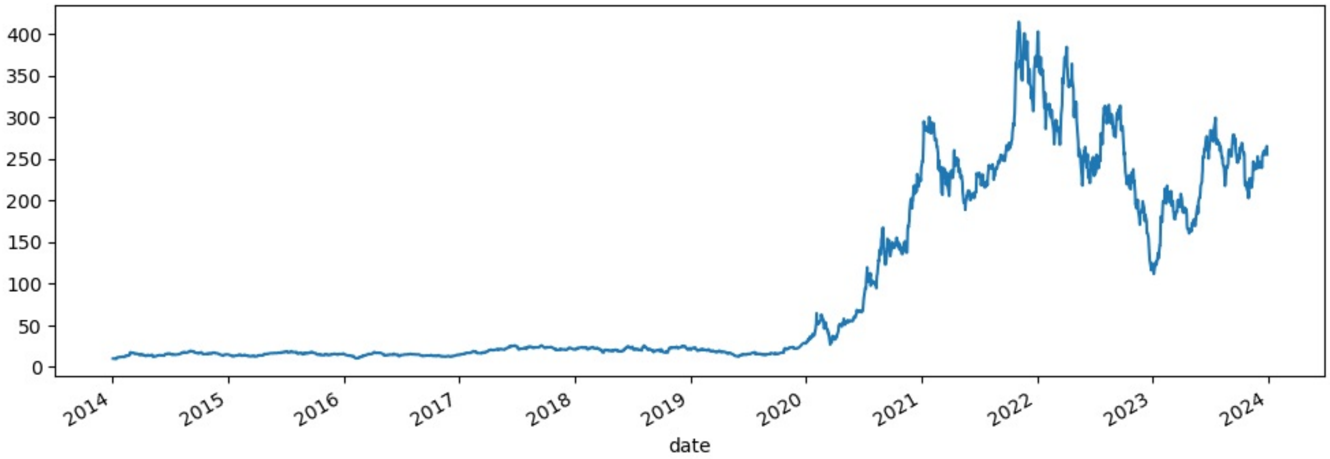
In [10]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(20, 15))
plt.plot(df.index, df['open'], label='Open')
plt.plot(df.index, df['close'], label='Close')
plt.plot(df.index, df['high'], label='High')
plt.plot(df.index, df['low'], label='Low')
plt.legend()
```

Out[10]:
```
<matplotlib.legend.Legend at 0x26f19305190>
```



```
In [11]: df['high'].plot(figsize=(12,4))
```

```
Out[11]: <Axes: xlabel='date'>
```



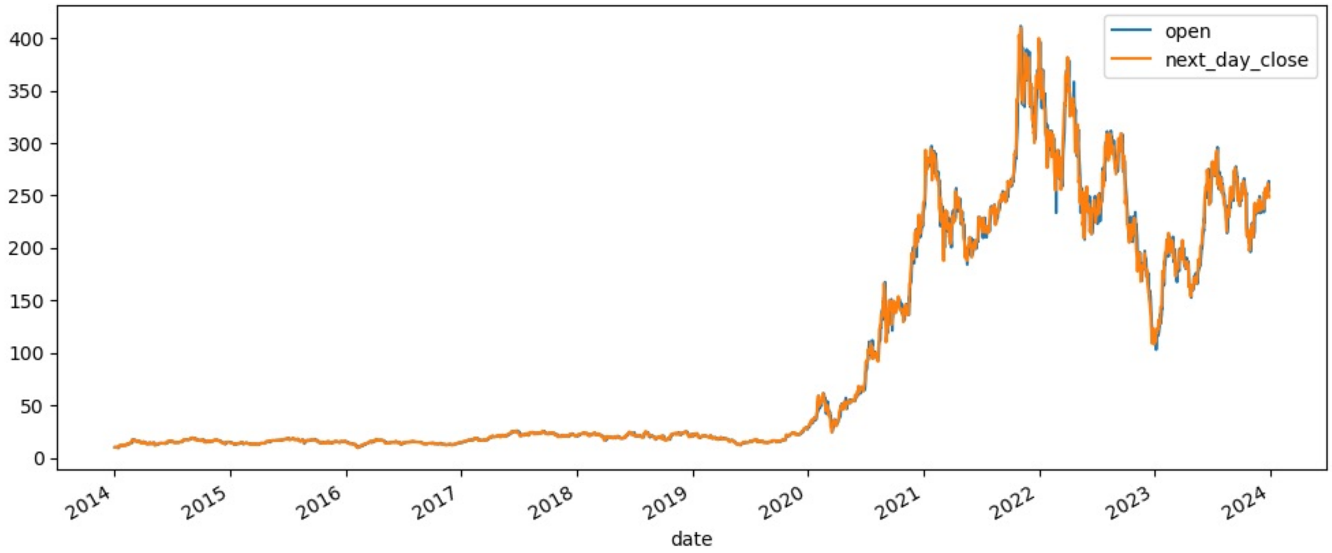
```
In [12]: df['high'].head()
```

```
Out[12]: date
2014-01-02    10.165333
2014-01-03    10.146000
2014-01-06    10.026667
2014-01-07    10.026667
2014-01-08    10.246667
Name: high, dtype: float64
```

```
In [13]: df[['open','next_day_close']].plot(figsize=(12,5))
```

Out[13]:

```
<Axes: xlabel='date'>
```

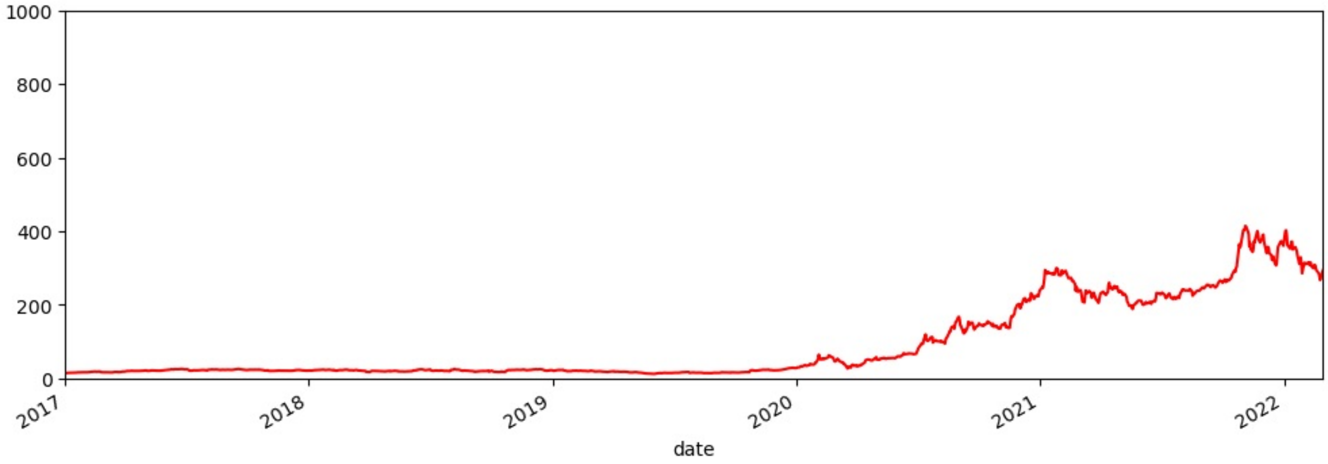


In [14]:

```
df['high'].plot(xlim=['2017-01-01','2022-02-28'],ylim=[0,1000],figsize=(12,4),c='red')
```

Out[14]:

```
<Axes: xlabel='date'>
```

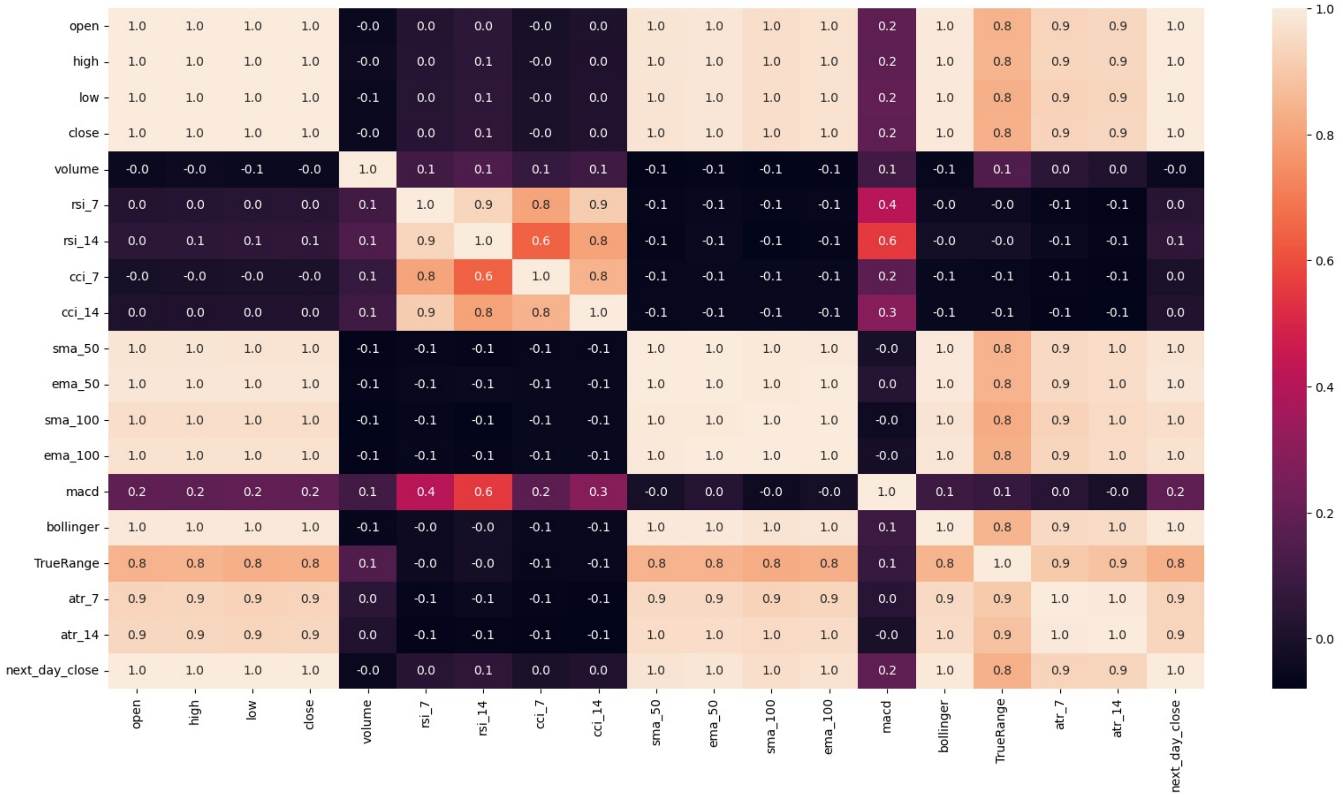


In [15]:

```
df_corr = df.corr()

import seaborn as sns
plt.figure(figsize=(20, 10))
sns.heatmap(df_corr, annot=True, fmt='.1f')
```

Out[15]:

```
<Axes: >
```



| | open | high | low | close | volume | rsi_7 | rsi_14 | cci_7 | cci_14 | sma_50 | ema_50 | sma_100 | ema_100 | macd | bollinger | TrueRange | atr_7 | atr_14 | next_day_close |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| open | 1.0 | 1.0 | 1.0 | 1.0 | -0.0 | 0.0 | 0.0 | -0.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.2 | 1.0 | 0.8 | 0.9 | 0.9 | 1.0 |
| high | 1.0 | 1.0 | 1.0 | 1.0 | -0.0 | 0.0 | 0.1 | -0.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.2 | 1.0 | 0.8 | 0.9 | 0.9 | 1.0 |
| low | 1.0 | 1.0 | 1.0 | 1.0 | -0.1 | 0.0 | 0.1 | -0.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.2 | 1.0 | 0.8 | 0.9 | 0.9 | 1.0 |
| close | 1.0 | 1.0 | 1.0 | 1.0 | -0.0 | 0.0 | 0.1 | -0.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.2 | 1.0 | 0.8 | 0.9 | 0.9 | 1.0 |
| volume | -0.0 | -0.0 | -0.1 | -0.0 | 1.0 | 0.1 | 0.1 | 0.1 | 0.1 | -0.1 | -0.1 | -0.1 | -0.1 | 0.1 | -0.1 | 0.1 | 0.0 | 0.0 | -0.0 |
| rsi_7 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 1.0 | 0.9 | 0.8 | 0.9 | -0.1 | -0.1 | -0.1 | -0.1 | 0.4 | -0.0 | -0.0 | -0.1 | -0.1 | 0.0 |
| rsi_14 | 0.0 | 0.1 | 0.1 | 0.1 | 0.1 | 0.9 | 1.0 | 0.6 | 0.8 | -0.1 | -0.1 | -0.1 | -0.1 | 0.6 | -0.0 | -0.0 | -0.1 | -0.1 | 0.1 |
| cci_7 | -0.0 | -0.0 | -0.0 | -0.0 | 0.1 | 0.8 | 0.6 | 1.0 | 0.8 | -0.1 | -0.1 | -0.1 | -0.1 | 0.2 | -0.1 | -0.1 | -0.1 | -0.1 | 0.0 |
| cci_14 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.9 | 0.8 | 0.8 | 1.0 | -0.1 | -0.1 | -0.1 | -0.1 | 0.3 | -0.1 | -0.1 | -0.1 | -0.1 | 0.0 |
| sma_50 | 1.0 | 1.0 | 1.0 | 1.0 | -0.1 | -0.1 | -0.1 | -0.1 | -0.1 | 1.0 | 1.0 | 1.0 | 1.0 | -0.0 | 1.0 | 0.8 | 0.9 | 1.0 | 1.0 |
| ema_50 | 1.0 | 1.0 | 1.0 | 1.0 | -0.1 | -0.1 | -0.1 | -0.1 | -0.1 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0 | 1.0 | 0.8 | 0.9 | 1.0 | 1.0 |
| sma_100 | 1.0 | 1.0 | 1.0 | 1.0 | -0.1 | -0.1 | -0.1 | -0.1 | -0.1 | 1.0 | 1.0 | 1.0 | 1.0 | -0.0 | 1.0 | 0.8 | 0.9 | 1.0 | 1.0 |
| ema_100 | 1.0 | 1.0 | 1.0 | 1.0 | -0.1 | -0.1 | -0.1 | -0.1 | -0.1 | 1.0 | 1.0 | 1.0 | 1.0 | -0.0 | 1.0 | 0.8 | 0.9 | 1.0 | 1.0 |
| macd | 0.2 | 0.2 | 0.2 | 0.2 | 0.1 | 0.4 | 0.6 | 0.2 | 0.3 | -0.0 | 0.0 | -0.0 | -0.0 | 1.0 | 0.1 | 0.1 | 0.0 | -0.0 | 0.2 |
| bollinger | 1.0 | 1.0 | 1.0 | 1.0 | -0.1 | -0.0 | -0.0 | -0.1 | -0.1 | 1.0 | 1.0 | 1.0 | 1.0 | 0.1 | 1.0 | 0.8 | 0.9 | 1.0 | 1.0 |
| TrueRange | 0.8 | 0.8 | 0.8 | 0.8 | 0.1 | -0.0 | -0.0 | -0.1 | -0.1 | 0.8 | 0.8 | 0.8 | 0.8 | 0.1 | 0.8 | 1.0 | 0.9 | 0.9 | 0.8 |
| atr_7 | 0.9 | 0.9 | 0.9 | 0.9 | 0.0 | -0.1 | -0.1 | -0.1 | -0.1 | 0.9 | 0.9 | 0.9 | 0.9 | 0.0 | 0.9 | 0.9 | 1.0 | 1.0 | 0.9 |
| atr_14 | 0.9 | 0.9 | 0.9 | 0.9 | 0.0 | -0.1 | -0.1 | -0.1 | -0.1 | 1.0 | 1.0 | 1.0 | 1.0 | -0.0 | 1.0 | 0.9 | 1.0 | 1.0 | 0.9 |
| next_day_close | 1.0 | 1.0 | 1.0 | 1.0 | -0.0 | 0.0 | 0.1 | 0.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.2 | 1.0 | 0.8 | 0.9 | 0.9 | 1.0 |

The “next_day_close” column has a 100% percent correlation with “open”, “close”, “low”, “high”, “sma_50”, and so on. This means if values in those columns increase the values in the “next_day_close” column will also increase.

In [16]:
```
y = df['next_day_close']
X = df.drop(columns='next_day_close')
```

In [17]:
```
X
```

Out[17]:

| | open | high | low | close | volume | rsi_7 | rsi_14 | cci_7 | cci_14 | sma_50 | ema_50 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| date | | | | | | | | | | | | |
| 2014-01-02 | 9.986667 | 10.165333 | 9.770000 | 10.006667 | 92826000 | 55.344071 | 54.440118 | -37.373644 | 15.213422 | 9.682107 | 9.820167 | 1 |
| 2014-01-03 | 10.000000 | 10.146000 | 9.906667 | 9.970667 | 70425000 | 53.742629 | 53.821521 | -81.304471 | 17.481130 | 9.652800 | 9.826069 | 1 |
| 2014-01-06 | 10.000000 | 10.026667 | 9.682667 | 9.800000 | 80416500 | 46.328174 | 50.870410 | -123.427544 | -37.824708 | 9.629467 | 9.825047 | 1 |
| 2014-01-07 | 9.841333 | 10.026667 | 9.683333 | 9.957333 | 75511500 | 53.263037 | 53.406750 | -84.784651 | -20.779431 | 9.597747 | 9.830235 | 1 |
| 2014-01-08 | 9.923333 | 10.246667 | 9.917333 | 10.085333 | 92448000 | 58.368660 | 55.423026 | 60.799662 | 43.570559 | 9.573240 | 9.840239 | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 2023-12-22 | 256.760010 | 258.220001 | 251.369995 | 252.539993 | 93249800 | 58.296612 | 58.137456 | 50.821325 | 80.672033 | 232.553000 | 240.360582 | 24 |
| 2023-12-26 | 254.490005 | 257.970001 | 252.910004 | 256.609985 | 86892400 | 63.570549 | 60.824035 | 93.909968 | 86.446838 | 232.662800 | 240.997814 | 24 |
| 2023-12-27 | 258.350006 | 263.339996 | 257.519989 | 261.440002 | 106494400 | 68.998630 | 63.793639 | 171.938770 | 119.554558 | 232.813200 | 241.799468 | 24 |
| 2023-12-28 | 263.660004 | 265.130005 | 252.710007 | 253.179993 | 113619900 | 53.186966 | 55.978816 | 45.772983 | 73.958135 | 232.779799 | 242.245763 | 24 |
| 2023-12-29 | 255.100006 | 255.190002 | 247.429993 | 248.479996 | 100615300 | 46.164227 | 52.070118 | -98.880167 | -0.320098 | 232.895800 | 242.490243 | 24 |

2516 rows × 18 columns

In [18]:
```
y
```

Out[18]:
```
date
2014-01-02      9.970667
2014-01-03      9.800000
2014-01-06      9.957333
2014-01-07     10.085333
2014-01-08      9.835333
                 ...
2023-12-22    256.609985
2023-12-26    261.440002
2023-12-27    253.179993
2023-12-28    248.479996
2023-12-29    248.419998
Name: next_day_close, Length: 2516, dtype: float64
```

In [19]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=777)
```

In [20]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.fit_transform(X_test)
X_train_scaled.shape, X_test_scaled.shape # ((2012, 18), (504, 18))
```

Out[20]:
```
((2012, 18), (504, 18))
```

In [21]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X_train, y_train)
lr_prediction = lr.predict(X_test)

from sklearn.metrics import mean_absolute_error
lr_mae = mean_absolute_error(y_test, lr_prediction)
```

In [22]:
```
lr_mae
```

Out[22]:
```
2.3867493379877676
```

## Our model is performing amazingly well. Our linear regression model is just giving a very low error.

2.38*meansthat*, *iftheactualvalueis*12.38 our prediction would be $10.

It’s not that bad. Let’s check other models’ performance as well.

In [23]:
```
from sklearn.tree import DecisionTreeRegressor
dtr = DecisionTreeRegressor()
dtr.fit(X_train, y_train)
dtr_prediction = dtr.predict(X_test)
```

```
dtr_mae = mean_absolute_error(y_test, dtr_prediction)
dtr_mae
```

Out[23]:
```
3.588810936507936
```

In [24]:
```
from sklearn.ensemble import RandomForestRegressor
rfr = RandomForestRegressor()
rfr.fit(X_train, y_train)
rfr_prediction = rfr.predict(X_test)
rfr_mae = mean_absolute_error(y_test, rfr_prediction)
rfr_mae
```

Out[24]:
```
2.662456560972218
```

In [25]:
```
from sklearn.svm import SVR
svr = SVR()
svr.fit(X_train, y_train)
svr_prediction = svr.predict(X_test)
svr_mae = mean_absolute_error(y_test, svr_prediction)
svr_mae
```

Out[25]:
```
74.6380006751906
```

In [ ]:

Processing math: 100%