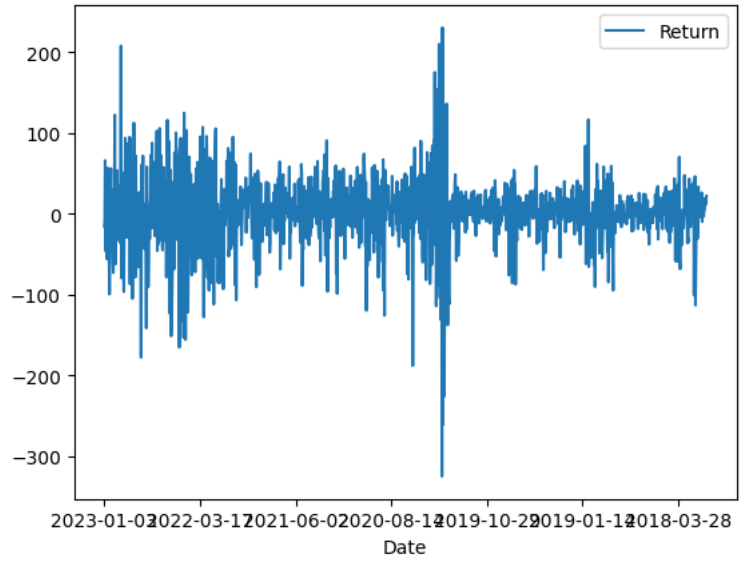
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error


data_index = pd.read_csv('sp500.csv')
data_index = data_index.drop(['Open', 'High', 'Low', 'Close*', 'Volume', 'Adj Close**'], axis='columns')
data_index.set_index('Date', inplace=True)
data_index.plot()
```

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




```
# read and merge datasets
data_ohlc = pd.read_csv('sp500.csv')
# data_ohlc['Date'] = pd.to_datetime(data_ohlc['Date'])
# data_ohlc['Date'] = data_ohlc['Date'].dt.strftime("%Y-%m-%d")
```



```
merged[['Open', 'High', 'Low', 'Close', 'Volume', 'Adj Close']] = merged[['Open', 'High', 'Low', 'Close*', 'Volume', 'Adj Close**']].replace(
# merged[['Close', 'Volume']] = merged[['Close*', 'Volume']].astype(float)
merged = merged.drop(['Close*', 'S&P500', 'Adj Close**'], axis='columns')
merged['Date'] = [int(datetime.strptime(date, '%Y-%m-%d').timestamp()) for date in merged['Date']]

# merged.dropna(inplace=True)
merged.to_csv('form1.csv',index=False)
# pd.read_csv('form1.csv').info()
set = pd.read_csv('form1.csv')
set.head(10)
```

	Date	Open	High	Low	Volume	Return	Close	Adj Close
0	1672704000	3853.29	3878.46	3794.33	3.959140e+09	-15.36	3824.14	3824.14

```
X = set[['Date', 'Open', 'High', 'Low', 'Volume', 'Close', 'Adj Close']]
# X = set[['Volume', 'S&P500', 'Close', 'Day', 'Month', 'Year']]
y = set['Return']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# svm_model = SVR(kernel='poly', degree=3, C=100)# best parameter
svm_model = SVR(kernel='linear')
# svm_model = SVR(kernel='rbf', C=1.0, epsilon=0.1)
svm_model.fit(X_train_scaled, y_train)
```

```
SVR
SVR(kernel='linear')
```

```
y_pred = svm_model.predict(X_test_scaled)
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)

print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"mean Absolute Error (MAE): {mae:.2f}")
```

```
Mean Squared Error (MSE): 1790.62
mean Absolute Error (MAE): 28.78
```

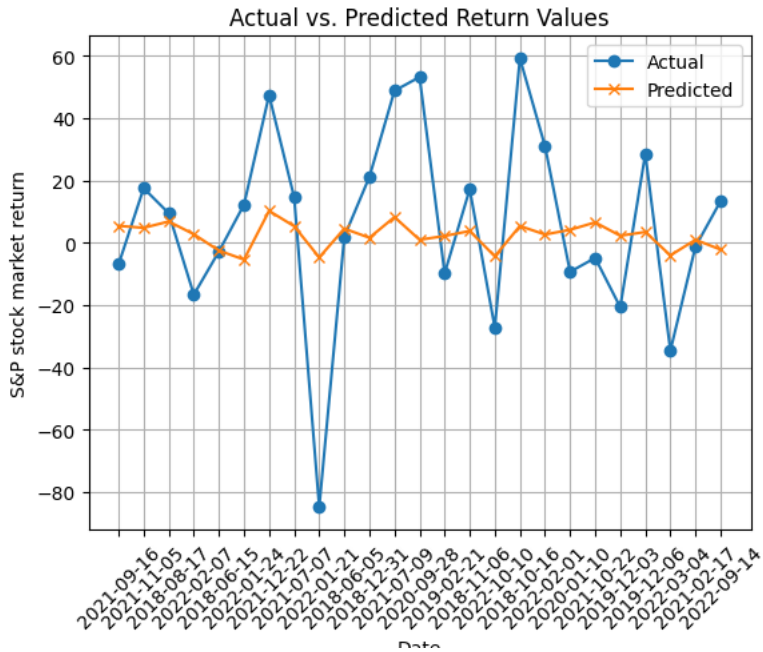
```
X_test['Date'] = [datetime.fromtimestamp(timestamp).strftime('%Y-%m-%d') for timestamp in X_test['Date']]
quick_test = pd.DataFrame({'Actual': y_test, 'Predicted' : y_pred})


merged_df = quick_test.join(X_test)
merged_df = merged_df.drop([ 'Open','High', 'Low', 'Volume', 'Adj Close', 'Close'], axis='columns')
merged_df
```

	Actual	Predicted	Date
326	-6.95	5.397390	2021-09-16
290	17.47	4.769143	2021-11-05
1101	0.44	6.794012	2018-08-17
1145	-2.83	-2.534731	2018-06-15
...	...	...	...
578	-15.71	-3.077816	2020-09-16
925	-6.21	1.825251	2019-05-02
163	9.81	-8.229415	2022-05-10
354	7.42	7.806443	2021-08-06
1096	17.71	8.584455	2018-08-24

252 rows × 3 columns

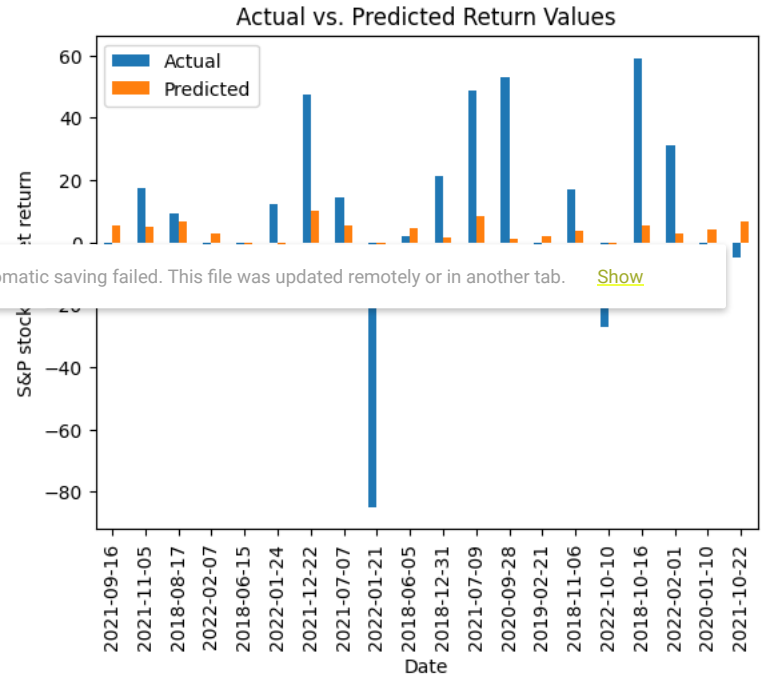
Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
merged_df = quick_test.join(X_test)
merged_df = merged_df.head(25)
plt.plot(merged_df['Date'], merged_df['Actual'], label='Actual', marker='o')
plt.plot(merged_df['Date'], merged_df['Predicted'], label='Predicted', marker='x')
plt.xlabel('Date')
plt.ylabel('S&P stock market return')
plt.title('Actual vs. Predicted Return Values')
plt.legend()
plt.xticks(rotation=45)
plt.grid()
plt.show()
```




```
# X_test['Date'] = [datetime.fromtimestamp(timestamp).strftime('%Y-%m-%d') for timestamp in X_test['Date']]
# merged_df = quick_test.join(X_test)
merged_df.set_index('Date', inplace=True)
graph = merged_df.head(20)
graph.plot(kind='bar')
plt.xlabel('Date')
plt.ylabel('S&P stock market return')
plt.title('Actual vs. Predicted Return Values')
```

```
Text(0.5, 1.0, 'Actual vs. Predicted Return Values')
```



**** WITH OTHER DATA ****

```
# read and merge datasets
data_ohlc = pd.read_csv('sp500.csv')
# data_ohlc['Date'] = pd.to_datetime(data_ohlc['Date'])
# data_ohlc['Date'] = data_ohlc['Date'].dt.strftime("%Y-%m-%d")

data_index = pd.read_csv('sp500_index.csv')
merged = pd.merge(data_ohlc, data_index, on='Date', how='inner')
```

```
merged[['Open','High', 'Low', 'Adj Close', 'Close', 'Volume']] = merged[['Open','High', 'Low', 'Adj Close**', 'Close*', 'Volume']].replace({'
# merged[['Open','High', 'Low', 'Adj Close**', 'Close', 'Volume']] = merged[['Close*', 'Volume']].astype(float)
merged = merged.drop(['Adj Close**', 'Close*', 'S&P500'], axis='columns')

# merged.dropna(inplace=True)
print(merged)
```

```
            Date     Open     High      Low        Volume  Return  Adj Close  \
0     2023-01-03  3853.29  3878.46  3794.33  3.959140e+09  -15.36    3824.14
1     2022-12-30  3829.06  3839.85  3800.34  2.979870e+09   -9.78    3839.50
2     2022-12-29  3805.45  3858.19  3805.45  3.003680e+09   66.06    3849.28
3     2022-12-28  3829.56  3848.32  3780.78  3.083520e+09  -46.03    3783.22
4     2022-12-27  3843.34  3846.65  3813.22  3.030300e+09  -15.57    3829.25
...          ...      ...      ...      ...           ...     ...        ...
1255  2018-01-08  2742.67  2748.51  2737.60  3.246160e+09    4.56    2747.71
1256  2018-01-05  2731.33  2743.45  2727.92  3.239280e+09   19.16    2743.15
1257  2018-01-04  2719.31  2729.29  2719.07  3.697340e+09   10.93    2723.99
1258  2018-01-03  2697.85  2714.37  2697.77  3.544030e+09   17.25    2713.06
1259  2018-01-02  2683.73  2695.89  2682.36  3.397430e+09   22.20    2695.81

        Close
0     3824.14
1     3839.50
2     3849.28
3     3783.22
4     3829.25
...       ...
1255  2747.71
1256  2743.15
1257  2723.99
1258  2713.06
1259  2695.81

[1260 rows x 8 columns]
```

```
data_news = pd.read_csv('sp500_news.csv')
merged_data = pd.merge(merged, data_news, on='Date', how='inner')
# merged_data['Date'] = pd.to_datetime(merged_data['Date'])
# merged_data['Day'] = merged_data['Date'].dt.day
# merged_data['Month'] = merged_data['Date'].dt.month
# merged_data['Year'] = merged_data['Date'].dt.year
merged_data['Date'] = [int(datetime.strptime(date, '%Y-%m-%d').timestamp()) for date in merged_data['Date']]
print(merged_data)

merged_data.info()
merged_data.isnull().sum()
```

```
           Date     Open     High      Low        Volume  Return  Adj Close  \
0   [illegible]                                                       [illegible]2.10
1   [illegible]                                                       [illegible]0.14
2   [illegible]                                                       [illegible]7.18
3    1591574400  3199.92  3233.13  3196.00  8.498300e+09   38.46    3232.39
4    1591315200  3163.84  3211.72  3163.84  8.639300e+09   81.58    3193.93
..          ...      ...      ...      ...           ...     ...        ...
610  1515369600  2742.67  2748.51  2737.60  3.246160e+09    4.56    2747.71
611  1515110400  2731.33  2743.45  2727.92  3.239280e+09   19.16    2743.15
612  1515024000  2719.31  2729.29  2719.07  3.697340e+09   10.93    2723.99
613  1514937600  2697.85  2714.37  2697.77  3.544030e+09   17.25    2713.06
614  1514851200  2683.73  2695.89  2682.36  3.397430e+09   22.20    2695.81

       Close  Positive  Negative   Neutral  Price_change
0    3002.10  0.067369  0.743126  0.189504            -1
1    3190.14  0.390972  0.240399  0.368628            -1
2    3207.18  0.296441  0.337487  0.366072            -1
3    3232.39  0.523481  0.124646  0.351873             1
4    3193.93  0.567891  0.070503  0.361606             1
..       ...       ...       ...       ...           ...
610  2747.71  0.281969  0.176601  0.541430             1
611  2743.15  0.134174  0.150020  0.715806             1
612  2723.99  0.201664  0.272806  0.525530             1
613  2713.06  0.203532  0.295476  0.500992             1
614  2695.81  0.255685  0.267352  0.476963             1

[615 rows x 12 columns]
<class 'pandas.core.frame.DataFrame'>
Int64Index: 615 entries, 0 to 614
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Date          615 non-null    int64
 1   Open          615 non-null    float64
 2   High          615 non-null    float64
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
 3   Low           615 non-null    float64
 4   Volume        615 non-null    float64
 5   Return        615 non-null    float64
 6   Adj Close     615 non-null    float64
 7   Close         615 non-null    float64
 8   Positive      615 non-null    float64
 9   Negative      615 non-null    float64
 10  Neutral       615 non-null    float64
 11  Price_change  615 non-null    int64
dtypes: float64(10), int64(2)
memory usage: 62.5 KB
Date            0
Open            0
High            0
Low             0
Volume          0
Return          0
Adj Close       0
Close           0
Positive        0
Negative        0
Neutral         0
Price_change    0
```

```
merged_data.loc[:, 'ma20'] = merged_data.Close.rolling(20).mean()
merged_data.dropna(inplace=True)

# merged_data.loc[:, "rsi"] = talib.RSI(merged_data.Close, 14)
# print(merged_data)

merged_data.to_csv('form1.csv',index=False)
# pd.read_csv('form1.csv').info()
set = pd.read_csv('form1.csv')
set.head(10)
```

	Date	Open	High	Low	Volume	Return	Adj Close	Clos
0	1589414400	2794.54	2852.80	2766.64	5.651130e+09	32.50	2852.50	2852.5
1	1589328000	2865.86	2874.14	2793.15	6.151650e+09	-50.12	2820.00	2820.0
2	1589241600	2939.50	2945.82	2869.59	5.119630e+09	-60.07	2870.12	2870.1
3	1589155200	2915.46	2944.25	2903.44	4.819730e+09	0.39	2930.19	2930.1
4	1588896000	2908.83	2932.16	2902.88	4.876030e+09	48.61	2929.80	2929.8
5	1588809600	2878.26	2901.92	2876.48	5.178790e+09	32.77	2881.19	2881.1
6	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2848.4
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2868.4
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
# merged['Date'] = pd.to_datetime(merged['Date'])
# merged['Day'] = merged['Date'].dt.day
# merged['Month'] = merged['Date'].dt.month
# merged['Year'] = merged['Date'].dt.year
# print(merged)

# merged.to_csv('form1.csv',index=False)
# # pd.read_csv('form1.csv').info()
# set = pd.read_csv('form1.csv')
# set.head()


X = set[['Date', 'Open','High', 'Low', 'Volume', 'Adj Close', 'Positive', 'Negative', 'Neutral', 'Price_change', 'ma20', 'Close']]
# X = set[['Volume', 'S&P500', 'Close', 'Day', 'Month', 'Year']]
y = set['Return']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# svm_model = SVR(kernel='poly', degree=2, C=100)# best parameter
svm_model = SVR(kernel='linear')
```

```
# svm_model = SVR(kernel='rbf', C=1.0, epsilon=0.1)
svm_model.fit(X_train_scaled, y_train)
```

```
SVR
SVR(kernel='linear')
```

```
y_pred = svm_model.predict(X_test_scaled)
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)

print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"Mean Absolute Error (MAE): {mae:.2f}")
```

```
Mean Squared Error (MSE): 415.98
Mean Absolute Error (MAE): 13.93
```

```
quick_test = pd.DataFrame({'Actual': y_test, 'Predicted' : y_pred})
quick_test.head(10)
```

	Actual	Predicted
228	8.71	12.470199
157	14.95	10.933563
543	-39.09	-14.925421
467	23.21	18.101486
163	-14.72	-11.529882
345	21.11	17.920783
310	-9.82	-9.871308
65	5.66	11.605153
53	-24.54	-38.064555
363	30.20	17.454317

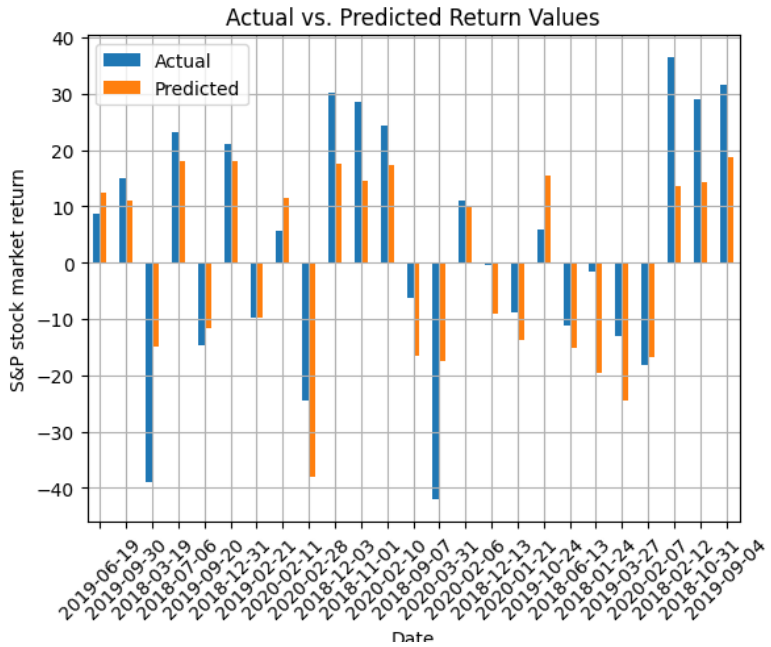
```
merged_df = quick_test.join(X_test)
merged_df = merged_df.drop([ 'Open','High', 'Low', 'Volume', 'Adj Close', 'Positive', 'Negative', 'Neutral', 'Price_change', 'ma20', 'Close']
merged_df['Date'] = [datetime.fromtimestamp(timestamp).strftime('%Y-%m-%d') for timestamp in merged_df['Date']]
merged_df
```


157	14.95	10.933563	2019-09-30
543	-39.09	-14.925421	2018-03-19
467	23.21	18.101486	2018-07-06
163	-14.72	-11.529882	2019-09-20
...	...	...	...
348	116.60	33.186884	2018-12-26
480	-5.91	-1.609366	2018-06-18
398	38.76	16.891868	2018-10-12
117	23.35	16.620362	2019-11-25
159	-7.25	-6.650064	2019-09-26

120 rows × 3 columns

```
merged_df.set_index('Date', inplace=True)

# Create the bar chart
graph = merged_df.head(25)
graph.plot(kind='bar')
plt.xlabel('Date')
plt.ylabel('S&P stock market return')
plt.title('Actual vs. Predicted Return Values')
```

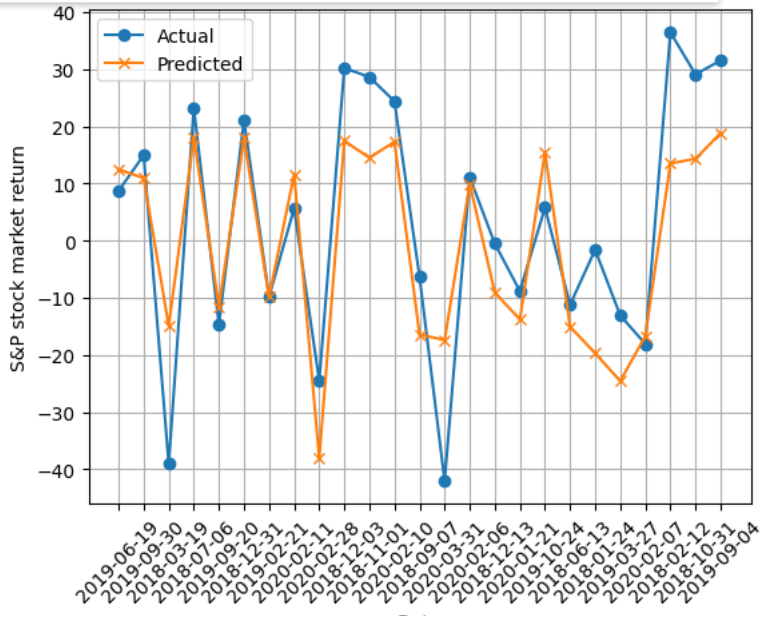
```
plt.legend()
plt.xticks(rotation=45)
plt.grid()
plt.show()
```




```
X_test['Date'] = [datetime.fromtimestamp(timestamp).strftime('%Y-%m-%d') for timestamp in X_test['Date']]
```

```
merged_df = quick_test.join(X_test)
merged_df = merged_df.head(25)
plt.plot(merged_df['Date'], merged_df['Actual'], label='Actual', marker='o')
plt.plot(merged_df['Date'], merged_df['Predicted'], label='Predicted', marker='x')
plt.xlabel('Date')
plt.ylabel('S&P stock market return')
plt.title('Actual vs. Predicted Return Values')
plt.legend()
plt.xticks(rotation=45)
plt.grid()
plt.show()
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff



✓ 0s completed at 21:17

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.

Automatic saving failed. This file was updated remotely or in another tab. Show diff