Analysis Report

~by Aarav and Jiyanshu

Objective:

The objective of this assignment was to develop a trading strategy using technical indicators for both small cap and large cap stocks listed on the National Stock Exchange (NSE) in India. The assignment required teamwork, coding implementations, signal generation, backtesting, performance analysis, and proposing improvements.

Approach:

1. Designing the Strategy:
    1) Indicators Chosen: For this assignment, we selected five volatility-based and five momentum-based technical indicators from the provided bucket. These indicators included the Keltner Channel and Stochastic Oscillator.
    2) Indicator Implementation: Aarav took the lead in coding the volatility-based indicator (Keltner Channel) and the momentum-based indicator (Stochastic Oscillator). Both indicators were implemented in Python, ensuring accurate calculations and appropriate visualisation of the indicator values. Jiyanshu actively participated in the coding process, providing suggestions and improvements to the codebase. We collaborated closely, discussing implementation strategies and fine-tuning the code for optimal functionality.

2. Signal Generation and Backtesting:
    1) Individual Strategies: Aarav primarily focused on implementing the volatility-based indicator strategy for both small cap and large cap stocks. Jiyanshu primarily focused on implementing the momentum-based indicator strategy for both categories of stocks. However, we maintained a collaborative approach throughout the project, with regular discussions and mutual agreement on the overall strategy design.
    2) Buy/Sell Signals: Using the implemented strategies, we generated buy and sell signals by applying the trading rules to historical price data. We ensured that the signal generation process considered the desired metric for taking trades, enhancing the effectiveness of the strategies.
    3) Trade Simulation and Returns Calculation: Collaboratively, we simulated the trades based on the generated signals and calculated the stock returns for the respective strategies. This process allowed us to evaluate the profitability and effectiveness of the strategies accurately.

3. Performance Analysis and Inferences:
    1) Performance Metrics: Aarav and Jiyanshu jointly evaluated the performance of the strategies by calculating various performance metrics such as returns, Sharpe ratio, cumulative returns, average expected returns, alpha and beta. These metrics provided quantitative measures of the strategies' profitability, risk-adjusted returns, and consistency. We used Python libraries such as pandas and numpy to perform the necessary calculations.
    2) Comparison of Strategies: We compared the performance of the momentum-based and volatility-based indicator strategies individually and also when combined with each other. This analysis allowed us to identify the strengths, weaknesses, and potential areas of improvement for each strategy. The collaborative effort ensured a comprehensive assessment of the strategies' performance.
    3) Interpretation and Discussion: Based on the results obtained, we jointly interpreted the findings and discussed the strengths, weaknesses, and limitations of the strategies. We analysed the reasons behind the strategies' performance, identified any shortcomings, and proposed potential improvements to enhance their effectiveness. We discussed the significance of each indicator, their impact on generating trading signals, and the importance of considering multiple indicators in a comprehensive trading strategy.

4. Teamwork:
    1) Contributions of Each Team Member: Aarav made significant contributions by coding both the volatility-based indicator (Keltner Channel) and the momentum-based indicator (Stochastic Oscillator). Aarav ensured accurate implementation and visualisation of these indicators. Jiyanshu actively participated in the coding process, providing valuable suggestions and improvements to enhance the code's functionality. The collaborative effort ensured the development of robust indicators and strategies.
    2) Mutual Agreement: Throughout the project, Aarav and Jiyanshu worked collaboratively, maintaining effective communication and ensuring equal participation. While Aarav took the lead in coding the indicators, Jiyanshu actively contributed by suggesting improvements to the codebase. Conversely, Jiyanshu led the report-writing process, incorporating the analysis, findings, and proposed improvements. Aarav actively contributed to the report by providing input and suggestions.

Coding Implementations

For Large cap

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import yfinance as yf
```

```
#Largecap stock INFOSYS
stock = "INFY.NS"
data = yf.download(stock, '2021-06-11', '2023-06-10')

print(data)
```

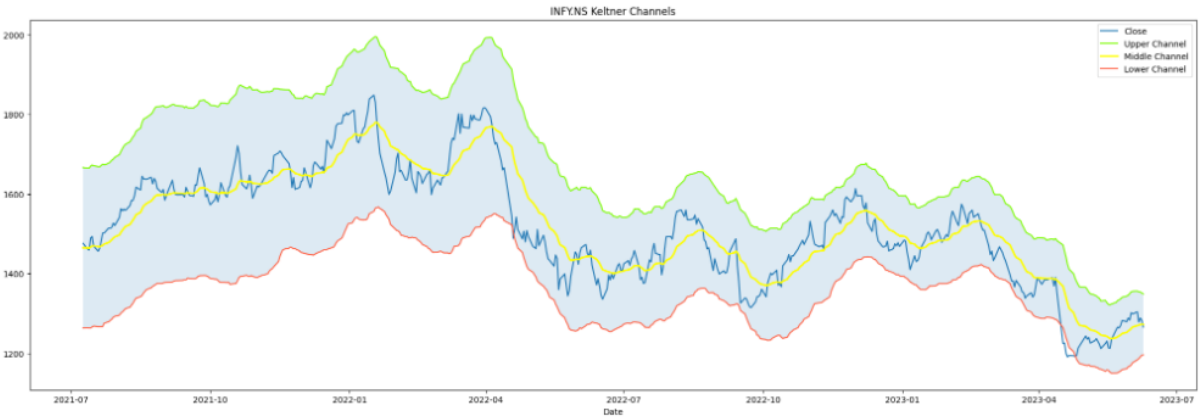
```
#ATR is over last 20 periods
data['TR'] = np.maximum(abs(data['High'] - data['Low']), abs(data['High'] - data['Adj Close'].shift(1)), abs(data['Low']
data['ATR'] = data['TR'].rolling(20).mean()

print((data['ATR']).tail(476))
```

```
#EMA is over last 20 periods
data['Middle'] = data['Adj Close'].ewm(span = 20).mean()

print((data['Middle']).tail(476))
```

```
data['Upper'] = data['Middle'] + 2 * data['ATR']

print(data['Upper'].tail(476))
```

```
data['Lower'] = data['Middle'] - 2 * data['ATR']

print(data['Lower'].tail(476))
```

```
plt.figure(figsize=(25, 8))
plt.plot(data.tail(476).index, data['Adj Close'].tail(476), label='Close')
plt.plot(data.tail(476).index, data['Upper'].tail(476), label='Upper Channel', color='lawngreen')
plt.plot(data.tail(476).index, data['Middle'].tail(476), label='Middle Channel', color='yellow', linewidth=2.25)
plt.plot(data.tail(476).index, data['Lower'].tail(476), label='Lower Channel', color='tomato')
plt.fill_between(data.tail(476).index, data['Upper'].tail(476), data['Lower'].tail(476), alpha = 0.15)
plt.title('INFY.NS Keltner Channels')
plt.xlabel('Date')
plt.ylabel('Price')
plt.legend()
plt.show()
```



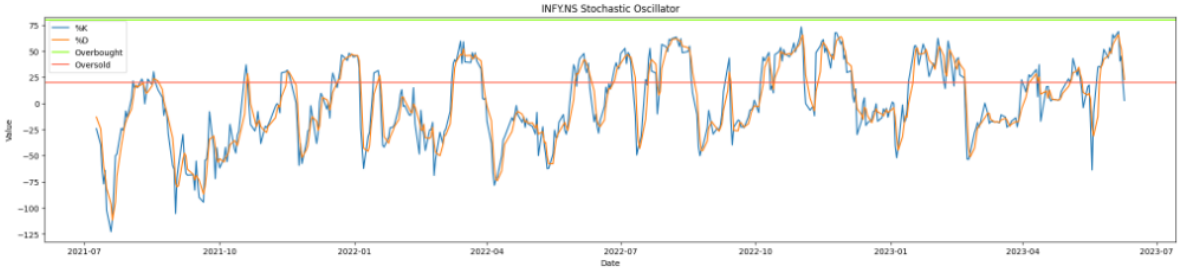
```
#Lowest of Lows is over last 14 periods
data['Lowest'] = data['Low'].rolling(14).min()
```

```
#Highest of Highs is over last 14 periods
data['Highest'] = data['High'].rolling(14).max()
```

```
data['%K'] = (data['Adj Close'] - data['Lowest'])/(data['Highest'] - data['Lowest']) * 100
```

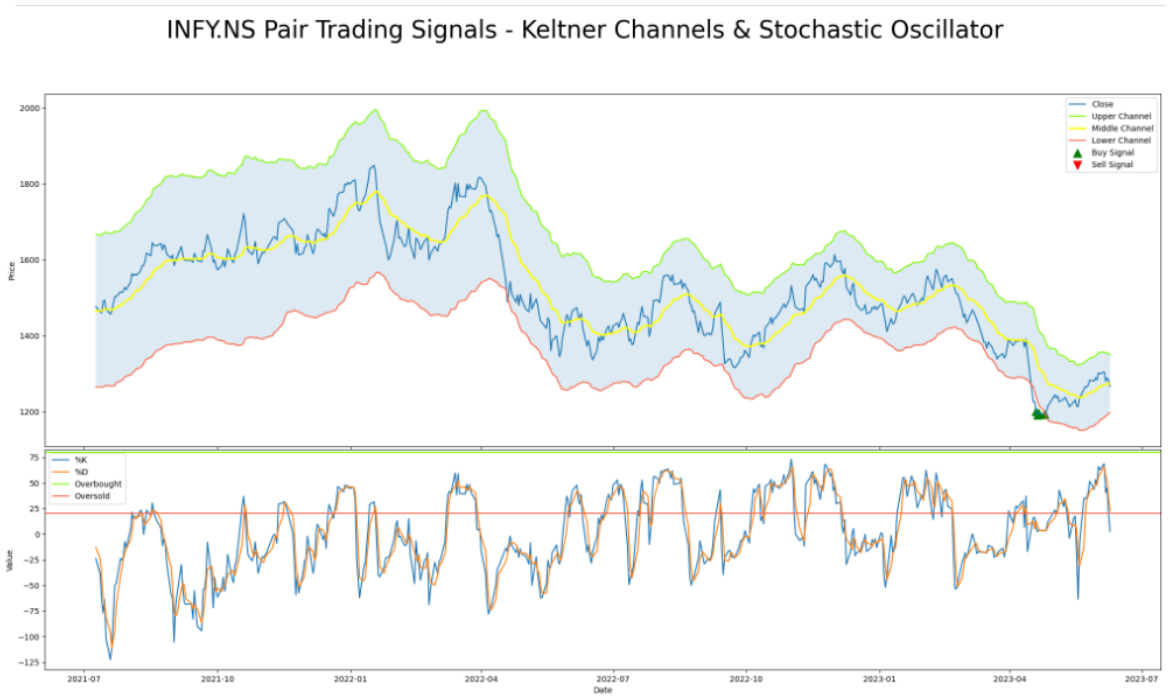
```
data['%D'] = data['%K'].rolling(3).mean()
```

```
plt.figure(figsize=(25, 5))
plt.plot(data.tail(476).index, data['%K'].tail(476), label='%K')
plt.plot(data.tail(476).index, data['%D'].tail(476), label='%D')
plt.axhline(80, label='Overbought', color='lawngreen')
plt.axhline(20, label='Oversold', color='tomato')
plt.title('INFY.NS Stochastic Oscillator')
plt.xlabel('Date')
plt.ylabel('Value')
plt.legend()
plt.show()
```



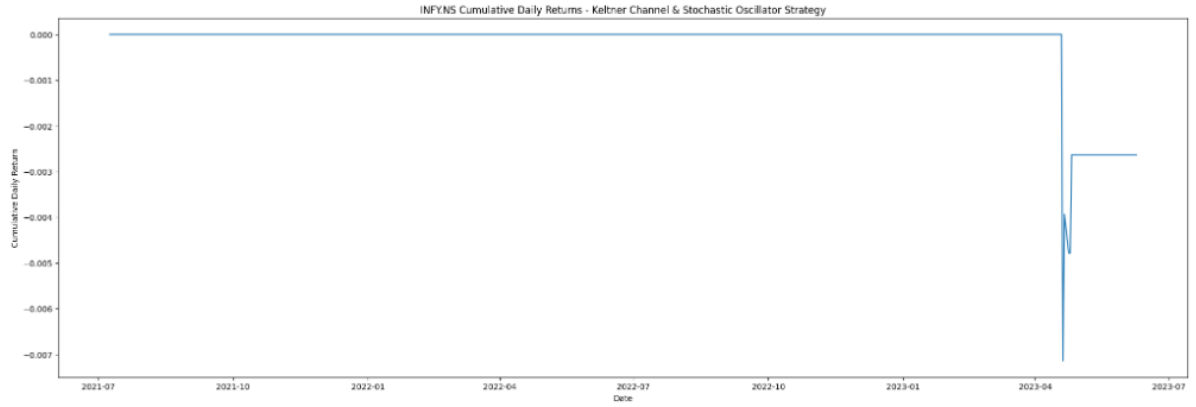

```
data['Signal'] = 0

#SELL
data.loc[(data['%K'] > data['%D']) & (data['Adj Close'] > data['Upper']), 'Signal'] = -1

#BUY
data.loc[(data['%K'] < data['%D']) & (data['Adj Close'] < data['Lower']), 'Signal'] = 1

print(data['Signal'].tail(496))
```

```
fig, (ax1, ax2) = plt.subplots(2, 1, sharex=True, figsize=(25, 13), gridspec_kw={'height_ratios': [8, 5]})

ax1.plot(data.tail(476).index, data['Adj Close'].tail(476), label='Close')
ax1.plot(data.tail(476).index, data['Upper'].tail(476), label='Upper Channel', color='lawngreen')
ax1.plot(data.tail(476).index, data['Middle'].tail(476), label='Middle Channel', color='yellow', linewidth=2.25)
ax1.plot(data.tail(476).index, data['Lower'].tail(476), label='Lower Channel', color='tomato')
ax1.fill_between(data.tail(476).index, data['Upper'].tail(476), data['Lower'].tail(476), alpha=0.15)
ax1.scatter(data[data['Signal'] == 1].tail(476).index, data[data['Signal'] == 1]['Adj Close'].tail(476), color='green', m
ax1.scatter(data[data['Signal'] == -1].tail(476).index, data[data['Signal'] == -1]['Adj Close'].tail(476), color='red', m
ax1.set_ylabel('Price')
ax1.legend()

ax2.plot(data.tail(476).index, data['%K'].tail(476), label='%K')
ax2.plot(data.tail(476).index, data['%D'].tail(476), label='%D')
ax2.axhline(80, label='Overbought', color='lawngreen')
ax2.axhline(20, label='Oversold', color='tomato')
ax2.set_ylabel('Value')
ax2.set_xlabel('Date')
ax2.legend()

plt.subplots_adjust(hspace=0.01)
fig.suptitle('INFY.NS Pair Trading Signals - Keltner Channels & Stochastic Oscillator', fontsize=30)
plt.show()
```




```
#daily returns rate
data['Return'] = data['Adj Close'].pct_change()

#daily strategy returns rate
data['Strategy_Return'] = data['Signal'].shift(1) * data['Return']

#cumulative daily returns rate of strategy
data['Cumulative_Return'] = (1 + data['Strategy_Return']).cumprod() - 1

print(data['Return'].tail(495))
print("")
print(data['Strategy_Return'].tail(495))
print("")
print(data['Cumulative_Return'].tail(495))
```

```
plt.figure(figsize=(25, 8))
plt.plot(data.tail(476).index, data['Cumulative_Return'].tail(476))
plt.title('INFY.NS Cumulative Daily Returns - Keltner Channel & Stochastic Oscillator Strategy')
plt.xlabel('Date')
plt.ylabel('Cumulative Daily Return')
plt.show()
```



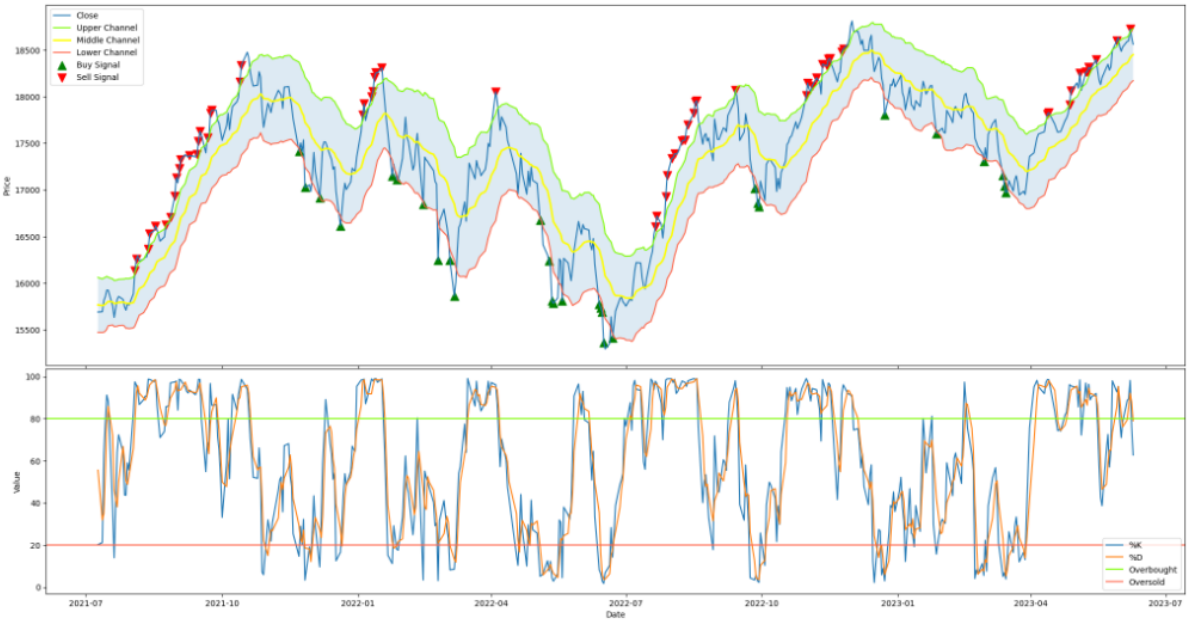

```
index = '^NSEI'
index_data = yf.download(index,'2021-06-11','2023-06-10')
```

```
index_data['TR'] = np.maximum(abs(index_data['High'] - index_data['Low']), abs(index_data['High'] - index_data['Adj Close
index_data['ATR'] = index_data['TR'].rolling(20).mean()
index_data['Middle'] = index_data['Adj Close'].ewm(span = 20).mean()
index_data['Upper'] = index_data['Middle'] + 2 * index_data['ATR']
index_data['Lower'] = index_data['Middle'] - 2 * index_data['ATR']

index_data['Lowest'] = index_data['Low'].rolling(14).min()
index_data['Highest'] = index_data['High'].rolling(14).max()
index_data['%K'] = (index_data['Adj Close'] - index_data['Lowest'])/(index_data['Highest'] - index_data['Lowest']) * 100
index_data['%D'] = index_data['%K'].rolling(3).mean()
```

```
index_data['Signal'] = 0

#SELL
index_data.loc[(index_data['%K'] > index_data['%D']) & (index_data['Adj Close'] > index_data['Upper']), 'Signal'] = -1

#BUY
index_data.loc[(index_data['%K'] < index_data['%D']) & (index_data['Adj Close'] < index_data['Lower']), 'Signal'] = 1
```

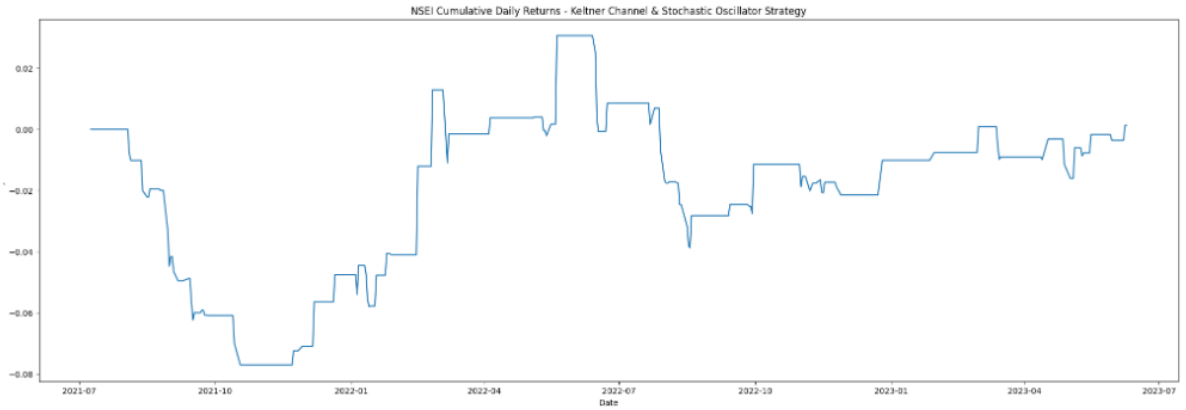
```
fig, (ax1, ax2) = plt.subplots(2, 1, sharex=True, figsize=(25, 13), gridspec_kw={'height_ratios': [8, 5]})

ax1.plot(index_data.tail(476).index, index_data['Adj Close'].tail(476), label='Close')
ax1.plot(index_data.tail(476).index, index_data['Upper'].tail(476), label='Upper Channel', color='lawngreen')
ax1.plot(index_data.tail(476).index, index_data['Middle'].tail(476), label='Middle Channel', color='yellow', linewidth=2.
ax1.plot(index_data.tail(476).index, index_data['Lower'].tail(476), label='Lower Channel', color='tomato')
ax1.fill_between(index_data.tail(476).index, index_data['Upper'].tail(476), index_data['Lower'].tail(476), alpha=0.15)
ax1.scatter(index_data[index_data['Signal'] == 1].tail(476).index, index_data[index_data['Signal'] == 1]['Adj Close'].tai
ax1.scatter(index_data[index_data['Signal'] == -1].tail(476).index, index_data[index_data['Signal'] == -1]['Adj Close'].t
ax1.set_ylabel('Price')
ax1.legend()

ax2.plot(index_data.tail(476).index, index_data['%K'].tail(476), label='%K')
ax2.plot(index_data.tail(476).index, index_data['%D'].tail(476), label='%D')
ax2.axhline(80, label='Overbought', color='lawngreen')
ax2.axhline(20, label='Oversold', color='tomato')
ax2.set_ylabel('Value')
ax2.set_xlabel('Date')
ax2.legend()

plt.subplots_adjust(hspace=0.01)
fig.suptitle('NSEI Pair Trading Signals - Keltner Channels & Stochastic Oscillator', fontsize=30)
plt.show()
```

NSEI Pair Trading Signals - Keltner Channels & Stochastic Oscillator




```
#daily returns rate
index_data['Return'] = index_data['Adj Close'].pct_change()

#daily strategy returns rate
index_data['Strategy_Return'] = index_data['Signal'].shift(1) * index_data['Return']

#cumulative daily returns rate of strategy
index_data['Cumulative_Return'] = (1 + index_data['Strategy_Return']).cumprod() - 1
```

```
plt.figure(figsize=(25, 8))
plt.plot(index_data.tail(476).index, index_data['Cumulative_Return'].tail(476))
plt.title('NSEI Cumulative Daily Returns - Keltner Channel & Stochastic Oscillator Strategy')
plt.xlabel('Date')
plt.ylabel('Cumulative Daily Return')
plt.show()
```




```
data['Mom_Signal'] = 0

#SELL
data.loc[(data['%K'] > data['%D']), 'Mom_Signal'] = -1

#BUY
data.loc[(data['%K'] < data['%D']), 'Mom_Signal'] = 1
```

```
index_data['Mom_Signal'] = 0

#SELL
index_data.loc[(index_data['%K'] > index_data['%D']), 'Mom_Signal'] = -1

#BUY
index_data.loc[(index_data['%K'] < index_data['%D']), 'Mom_Signal'] = 1
```

```
#daily strategy returns rate
data['Mom_Return'] = data['Mom_Signal'].shift(1) * data['Return']
index_data['Mom_Return'] = index_data['Mom_Signal'].shift(1) * index_data['Return']

#cumulative daily returns rate of strategy
data['Mom_Cumulative_Return'] = (1 + data['Mom_Return']).cumprod() - 1
index_data['Mom_Cumulative_Return'] = (1 + index_data['Mom_Return']).cumprod() - 1
```

```
data['Vol_Signal'] = 0

#SELL
data.loc[(data['Adj Close'] > data['Upper']), 'Vol_Signal'] = -1

#BUY
data.loc[(data['Adj Close'] < data['Lower']), 'Vol_Signal'] = 1
```

```
index_data['Vol_Signal'] = 0

#SELL
index_data.loc[(index_data['Adj Close'] > index_data['Upper']), 'Vol_Signal'] = -1

#BUY
index_data.loc[(index_data['Adj Close'] < index_data['Lower']), 'Vol_Signal'] = 1

print(index_data['Vol_Signal'].tail(496))
```

```
#daily strategy returns rate
data['Vol_Return'] = data['Vol_Signal'].shift(1) * data['Return']
index_data['Vol_Return'] = index_data['Vol_Signal'].shift(1) * index_data['Return']

#cumulative daily returns rate of strategy
data['Vol_Cumulative_Return'] = (1 + data['Vol_Return']).cumprod() - 1
index_data['Vol_Cumulative_Return'] = (1 + index_data['Vol_Return']).cumprod() - 1
```

```
#return is basically profit
#return rate at end of overall period is last value of 'Cumulative_Return'
total_profit_percenatge = (data['Strategy_Return'].tail(495)).sum() * 100
total_profit_percenatge_index = (index_data['Strategy_Return'].tail(495)).sum() * 100
total_profit_percenatge_Mom = (data['Mom_Return'].tail(495)).sum() * 100
total_profit_percenatge_Vol = (data['Vol_Return'].tail(495)).sum() * 100
total_profit_percenatge_Mom_index = (index_data['Mom_Return'].tail(495)).sum() * 100
total_profit_percenatge_Vol_index = (index_data['Vol_Return'].tail(495)).sum() * 100

print('Total Profit of stock using the combined strategy is:', total_profit_percenatge, '%')
print('Total Profit of index using the combined strategy is:', total_profit_percenatge_index, '%')
print('Total Profit of stock using just Momentum Indicator is:', total_profit_percenatge_Mom, '%')
print('Total Profit of index using just Momentum Indicator is:', total_profit_percenatge_Mom_index, '%')
print('Total Profit of stock using just Volatality Indicator is:', total_profit_percenatge_Vol, '%')
print('Total Profit of index using just Volatality Indicator is:', total_profit_percenatge_Vol_index, '%')
```

```
Total Profit of stock using the combined strategy is: -0.2604932655554171 %
Total Profit of index using the combined strategy is: 0.47760992596918017 %
Total Profit of stock using just Momentum Indicator is: -9.07281742830417 %
Total Profit of index using just Momentum Indicator is: -40.40652965831859 %
Total Profit of stock using just Volatality Indicator is: -0.8017808012066641 %
Total Profit of index using just Volatality Indicator is: 6.627289771988554 %
```

```
#beta
covariance = np.cov(data['Strategy_Return'].tail(495),index_data['Strategy_Return'].tail(495))[0,1]
variance = np.var(index_data['Strategy_Return'].tail(495))
beta = covariance/variance
covariance_Mom = np.cov(data['Mom_Return'].tail(495),index_data['Mom_Return'].tail(495))[0,1]
variance_Mom = np.var(index_data['Mom_Return'].tail(495))
beta_Mom = covariance/variance
covariance_Vol = np.cov(data['Vol_Return'].tail(495),index_data['Vol_Return'].tail(495))[0,1]
variance_Vol = np.var(index_data['Vol_Return'].tail(495))
beta_Vol = covariance/variance

print('Beta for combined strategy:', beta)
print('Beta for just Momentum Indicator:', beta_Mom)
print('Beta for just Volatality Indicator:', beta_Vol)
```

```
Beta for combined strategy: 3.6485922845229485e-06
Beta for just Momentum Indicator: 3.6485922845229485e-06
Beta for just Volatality Indicator: 3.6485922845229485e-06
```

```
#CAPM is per year
risk_free_rate = Rf = 0.05
average_market_return = Rm = index_data['Strategy_Return'].tail(495).mean()
market_premium = Rm-Rf
expected_return = Rf + beta * (market_premium)
average_market_return_Mom = Rm_Mom = index_data['Mom_Return'].tail(495).mean()
market_premium_Mom = Rm_Mom-Rf
expected_return_Mom = Rf + beta_Mom * (market_premium_Mom)
average_market_return_Vol = Rm_Vol = index_data['Vol_Return'].tail(495).mean()
market_premium_Vol = Rm_Vol-Rf
expected_return_Vol = Rf + beta_Vol * (market_premium_Vol)

print('Average Expected Return from combined strategy:', expected_return, '%')
print('Average Expected Return from just Momentum Indicator:', expected_return_Mom, '%')
print('Average Expected Return from just Volatality Indicator:', expected_return_Vol, '%')
```

```
Average Expected Return from combined strategy: 0.04999981760558989 %
Average Expected Return from just Momentum Indicator: 0.049999814592063506 %
Average Expected Return from just Volatality Indicator: 0.049999818058876246 %
```

```
#alpha is for year
average_stock_return = data['Strategy_Return'].tail(495).mean()
alpha = average_stock_return - expected_return
average_stock_return_Mom = data['Mom_Return'].tail(495).mean()
alpha_Mom = average_stock_return_Mom - expected_return_Mom
average_stock_return_Vol = data['Vol_Return'].tail(495).mean()
alpha_Vol = average_stock_return_Vol - expected_return_Vol

print('Alpha from combined strategy:', alpha)
print('Alpha from just Momentum Indicator:', alpha_Mom)
print('Alpha from just Volatality Indicator:', alpha_Vol)
```

```
Alpha from combined strategy: -0.05000508009580314
Alpha from just Momentum Indicator: -0.050183103833039346
Alpha from just Volatality Indicator: -0.05001601565081981
```

```
#sharpe ratio
annual_volatality = data['Strategy_Return'].tail(495).std()
sharpe_ratio = (average_market_return - risk_free_rate) / annual_volatality
annual_volatality_Mom = data['Mom_Return'].tail(495).std()
sharpe_ratio_Mom = (average_market_return_Mom - risk_free_rate) / annual_volatality_Mom
annual_volatality_Vol = data['Vol_Return'].tail(495).std()
sharpe_ratio_Vol = (average_market_return_Vol - risk_free_rate) / annual_volatality_Vol

print('Sharpe Ratio from combined strategy:', sharpe_ratio)
print('Sharpe Ratio from just Momentum Indicator:', sharpe_ratio_Mom)
print('Sharpe Ratio from just Volatality Indicator:', sharpe_ratio_Vol)
```

```
Sharpe Ratio from combined strategy: -135.9174558983441
Sharpe Ratio from just Momentum Indicator: -3.277746388972131
Sharpe Ratio from just Volatality Indicator: -39.082944424651835
```

For Small cap

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import yfinance as yf
```

```
#smallcap stock JK LAKSHMI CEMENTS
stock = "JKLAKSHMI.NS"
data = yf.download(stock, '2021-06-11', '2023-06-10')
```

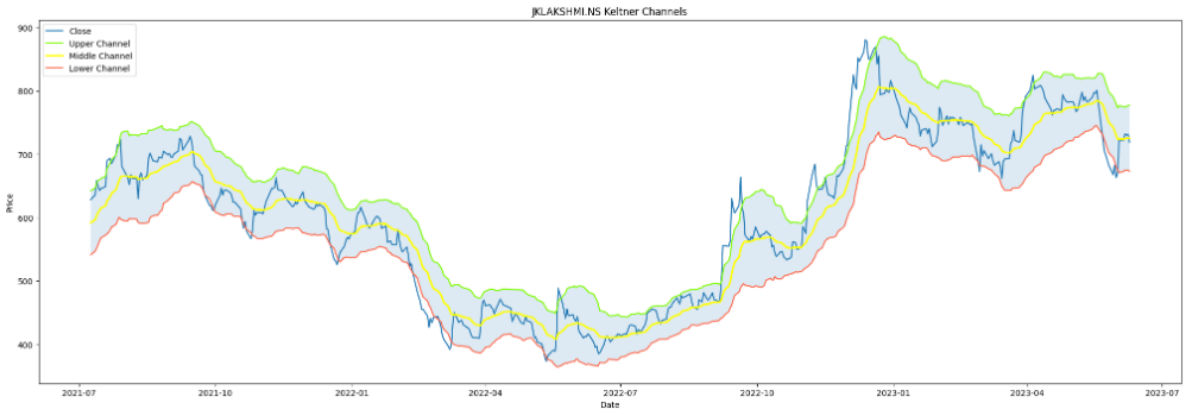
```
#ATR is over last 20 periods
data['TR'] = np.maximum(abs(data['High'] - data['Low']), abs(data['High'] - data['Adj Close'].shift(1)), abs(data['Low']
data['ATR'] = data['TR'].rolling(20).mean()
```

```
#EMA is over last 20 periods
data['Middle'] = data['Adj Close'].ewm(span = 20).mean()
```

```
data['Upper'] = data['Middle'] + 2 * data['ATR']

print(data['Upper'].tail(476))
```

```
data['Lower'] = data['Middle'] - 2 * data['ATR']

print(data['Lower'].tail(476))
```

```
plt.figure(figsize=(25, 8))
plt.plot(data.tail(476).index, data['Adj Close'].tail(476), label='Close')
plt.plot(data.tail(476).index, data['Upper'].tail(476), label='Upper Channel', color='lawngreen')
plt.plot(data.tail(476).index, data['Middle'].tail(476), label='Middle Channel', color='yellow', linewidth=2.25)
plt.plot(data.tail(476).index, data['Lower'].tail(476), label='Lower Channel', color='tomato')
plt.fill_between(data.tail(476).index, data['Upper'].tail(476), data['Lower'].tail(476), alpha = 0.15)
plt.title('JKLAKSHMI.NS Keltner Channels')
plt.xlabel('Date')
plt.ylabel('Price')
plt.legend()
plt.show()
```



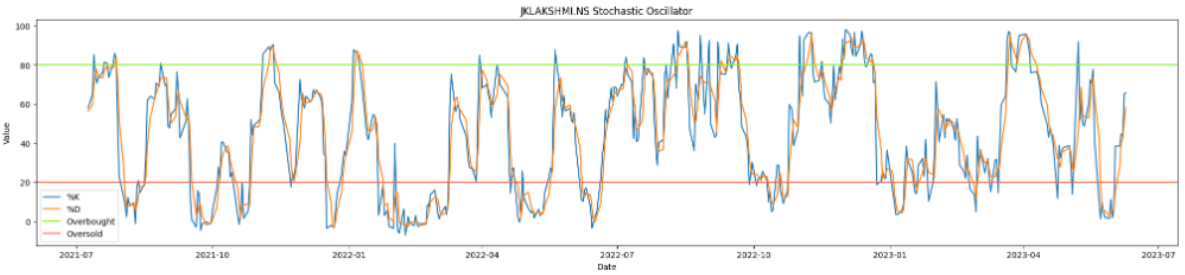

```
#Lowest of Lows is over last 14 periods
data['Lowest'] = data['Low'].rolling(14).min()
```

```
#Highest of Highs is over last 14 periods
data['Highest'] = data['High'].rolling(14).max()
```

```
data['%K'] = (data['Adj Close'] - data['Lowest'])/(data['Highest'] - data['Lowest']) * 100
```

```
data['%D'] = data['%K'].rolling(3).mean()
```

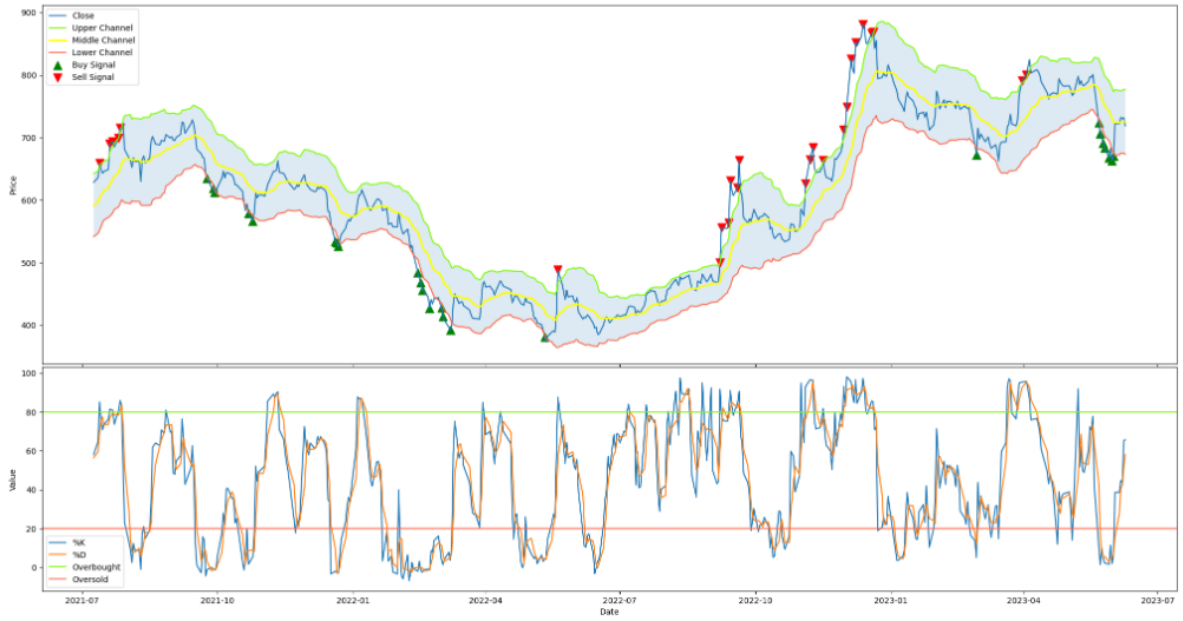
```
plt.figure(figsize=(25, 5))
plt.plot(data.tail(476).index, data['%K'].tail(476), label='%K')
plt.plot(data.tail(476).index, data['%D'].tail(476), label='%D')
plt.axhline(80, label='Overbought', color='lawngreen')
plt.axhline(20, label='Oversold', color='tomato')
plt.title('JKLAKSHMI.NS Stochastic Oscillator')
plt.xlabel('Date')
plt.ylabel('Value')
plt.legend()
plt.show()
```




```
data['Signal'] = 0

#SELL
data.loc[(data['%K'] > data['%D']) & (data['Adj Close'] > data['Upper']), 'Signal'] = -1

#BUY
data.loc[(data['%K'] < data['%D']) & (data['Adj Close'] < data['Lower']), 'Signal'] = 1
```

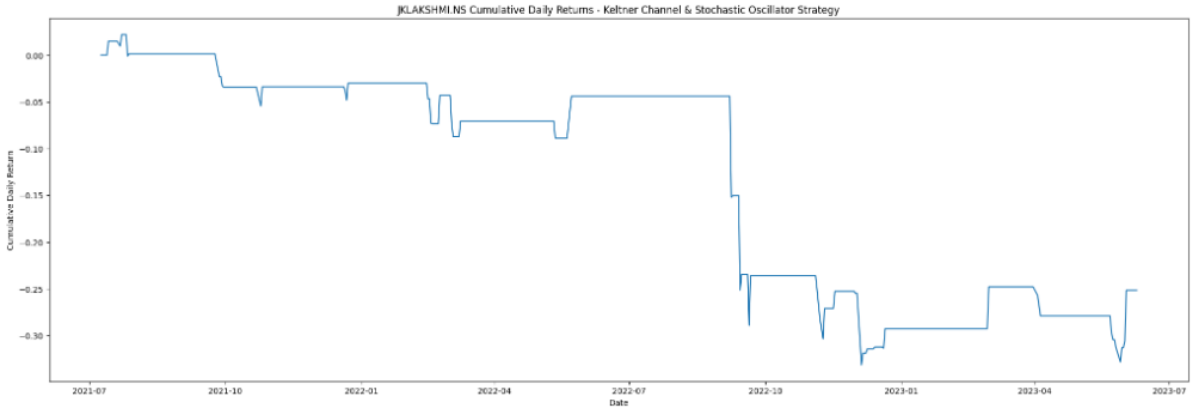
```
fig, (ax1, ax2) = plt.subplots(2, 1, sharex=True, figsize=(25, 13), gridspec_kw={'height_ratios': [8, 5]})

ax1.plot(data.tail(476).index, data['Adj Close'].tail(476), label='Close')
ax1.plot(data.tail(476).index, data['Upper'].tail(476), label='Upper Channel', color='lawngreen')
ax1.plot(data.tail(476).index, data['Middle'].tail(476), label='Middle Channel', color='yellow', linewidth=2.25)
ax1.plot(data.tail(476).index, data['Lower'].tail(476), label='Lower Channel', color='tomato')
ax1.fill_between(data.tail(476).index, data['Upper'].tail(476), data['Lower'].tail(476), alpha=0.15)
ax1.scatter(data[data['Signal'] == 1].tail(476).index, data[data['Signal'] == 1]['Adj Close'].tail(476), color='green', m
ax1.scatter(data[data['Signal'] == -1].tail(476).index, data[data['Signal'] == -1]['Adj Close'].tail(476), color='red', m
ax1.set_ylabel('Price')
ax1.legend()

ax2.plot(data.tail(476).index, data['%K'].tail(476), label='%K')
ax2.plot(data.tail(476).index, data['%D'].tail(476), label='%D')
ax2.axhline(80, label='Overbought', color='lawngreen')
ax2.axhline(20, label='Oversold', color='tomato')
ax2.set_ylabel('Value')
ax2.set_xlabel('Date')
ax2.legend()

plt.subplots_adjust(hspace=0.01)
fig.suptitle('JKLAKSHMI.NS Pair Trading Signals - Keltner Channels & Stochastic Oscillator', fontsize=30)
plt.show()
```

JKLAKSHMI.NS Pair Trading Signals - Keltner Channels & Stochastic Oscillator




```
#daily returns rate
data['Return'] = data['Adj Close'].pct_change()

#daily strategy returns rate
data['Strategy_Return'] = data['Signal'].shift(1) * data['Return']

#cumulative daily returns rate of strategy
data['Cumulative_Return'] = (1 + data['Strategy_Return']).cumprod() - 1
```

```
plt.figure(figsize=(25, 8))
plt.plot(data.tail(476).index, data['Cumulative_Return'].tail(476))
plt.title('JKLAKSHMI.NS Cumulative Daily Returns - Keltner Channel & Stochastic Oscillator Strategy')
plt.xlabel('Date')
plt.ylabel('Cumulative Daily Return')
plt.show()
```

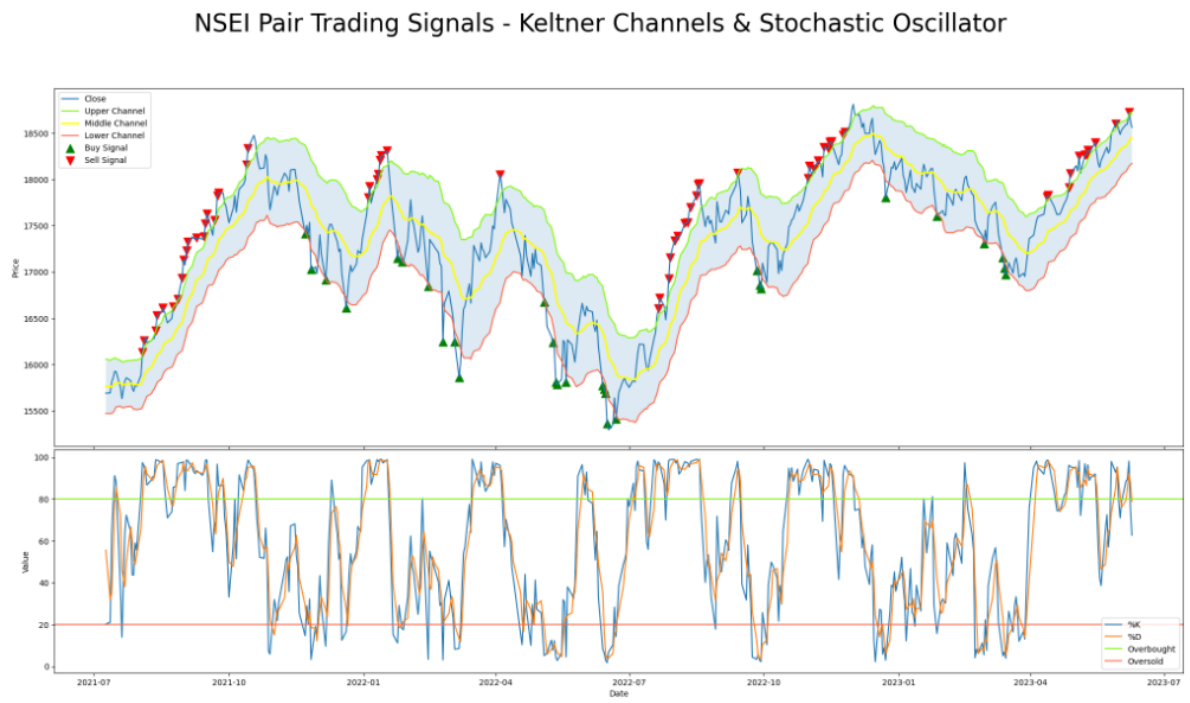



```
index = '^NSEI'
index_data = yf.download(index,'2021-06-11','2023-06-10')
```

```
index_data['TR'] = np.maximum(abs(index_data['High'] - index_data['Low']), abs(index_data['High'] - index_data['Adj Close
index_data['ATR'] = index_data['TR'].rolling(20).mean()
index_data['Middle'] = index_data['Adj Close'].ewm(span = 20).mean()
index_data['Upper'] = index_data['Middle'] + 2 * index_data['ATR']
index_data['Lower'] = index_data['Middle'] - 2 * index_data['ATR']

index_data['Lowest'] = index_data['Low'].rolling(14).min()
index_data['Highest'] = index_data['High'].rolling(14).max()
index_data['%K'] = (index_data['Adj Close'] - index_data['Lowest'])/(index_data['Highest'] - index_data['Lowest']) * 100
index_data['%D'] = index_data['%K'].rolling(3).mean()
```

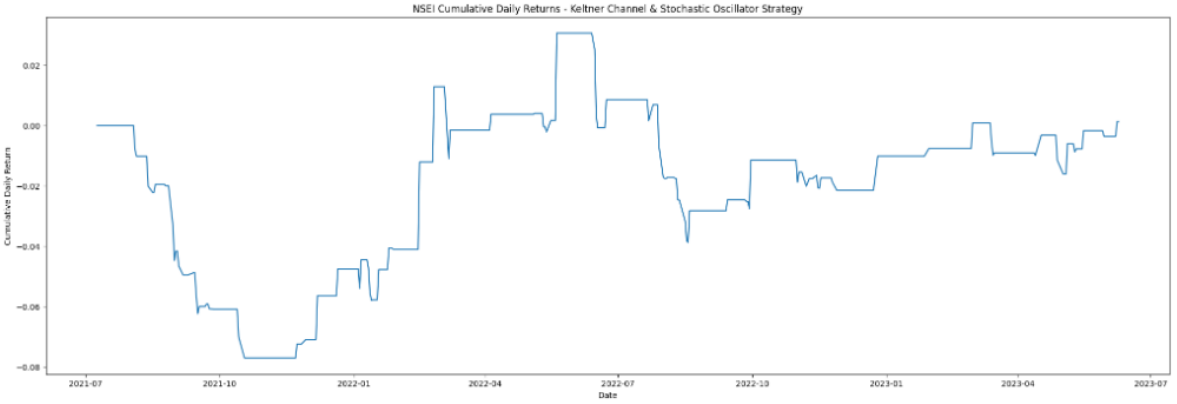
```
index_data['Signal'] = 0

#SELL
index_data.loc[(index_data['%K'] > index_data['%D']) & (index_data['Adj Close'] > index_data['Upper']), 'Signal'] = -1

#BUY
index_data.loc[(index_data['%K'] < index_data['%D']) & (index_data['Adj Close'] < index_data['Lower']), 'Signal'] = 1
```

```
fig, (ax1, ax2) = plt.subplots(2, 1, sharex=True, figsize=(25, 13), gridspec_kw={'height_ratios': [8, 5]})

ax1.plot(index_data.tail(476).index, index_data['Adj Close'].tail(476), label='Close')
ax1.plot(index_data.tail(476).index, index_data['Upper'].tail(476), label='Upper Channel', color='lawngreen')
ax1.plot(index_data.tail(476).index, index_data['Middle'].tail(476), label='Middle Channel', color='yellow', linewidth=2.
ax1.plot(index_data.tail(476).index, index_data['Lower'].tail(476), label='Lower Channel', color='tomato')
ax1.fill_between(index_data.tail(476).index, index_data['Upper'].tail(476), index_data['Lower'].tail(476), alpha=0.15)
ax1.scatter(index_data[index_data['Signal'] == 1].tail(476).index, index_data[index_data['Signal'] == 1]['Adj Close'].tai
ax1.scatter(index_data[index_data['Signal'] == -1].tail(476).index, index_data[index_data['Signal'] == -1]['Adj Close'].t
ax1.set_ylabel('Price')
ax1.legend()

ax2.plot(index_data.tail(476).index, index_data['%K'].tail(476), label='%K')
ax2.plot(index_data.tail(476).index, index_data['%D'].tail(476), label='%D')
ax2.axhline(80, label='Overbought', color='lawngreen')
ax2.axhline(20, label='Oversold', color='tomato')
ax2.set_ylabel('Value')
ax2.set_xlabel('Date')
ax2.legend()

plt.subplots_adjust(hspace=0.01)
fig.suptitle('NSEI Pair Trading Signals - Keltner Channels & Stochastic Oscillator', fontsize=30)
plt.show()
```



```
#daily returns rate
index_data['Return'] = index_data['Adj Close'].pct_change()

#daily strategy returns rate
index_data['Strategy_Return'] = index_data['Signal'].shift(1) * index_data['Return']

#cumulative daily returns rate of strategy
index_data['Cumulative_Return'] = (1 + index_data['Strategy_Return']).cumprod() - 1
```

```
plt.figure(figsize=(25, 8))
plt.plot(index_data.tail(476).index, index_data['Cumulative_Return'].tail(476))
plt.title('NSEI Cumulative Daily Returns - Keltner Channel & Stochastic Oscillator Strategy')
plt.xlabel('Date')
plt.ylabel('Cumulative Daily Return')
plt.show()
```




```
data['Mom_Signal'] = 0

#SELL
data.loc[(data['%K'] > data['%D']), 'Mom_Signal'] = -1

#BUY
data.loc[(data['%K'] < data['%D']), 'Mom_Signal'] = 1
```

```
index_data['Mom_Signal'] = 0

#SELL
index_data.loc[(index_data['%K'] > index_data['%D']), 'Mom_Signal'] = -1

#BUY
index_data.loc[(index_data['%K'] < index_data['%D']), 'Mom_Signal'] = 1
```

```
#daily strategy returns rate
data['Mom_Return'] = data['Mom_Signal'].shift(1) * data['Return']
index_data['Mom_Return'] = index_data['Mom_Signal'].shift(1) * index_data['Return']

#cumulative daily returns rate of strategy
data['Mom_Cumulative_Return'] = (1 + data['Mom_Return']).cumprod() - 1
index_data['Mom_Cumulative_Return'] = (1 + index_data['Mom_Return']).cumprod() - 1
```

```
data['Vol_Signal'] = 0

#SELL
data.loc[(data['Adj Close'] > data['Upper']), 'Vol_Signal'] = -1

#BUY
data.loc[(data['Adj Close'] < data['Lower']), 'Vol_Signal'] = 1
```

```
index_data['Vol_Signal'] = 0

#SELL
index_data.loc[(index_data['Adj Close'] > index_data['Upper']), 'Vol_Signal'] = -1

#BUY
index_data.loc[(index_data['Adj Close'] < index_data['Lower']), 'Vol_Signal'] = 1
```

```
#daily strategy returns rate
data['Vol_Return'] = data['Vol_Signal'].shift(1) * data['Return']
index_data['Vol_Return'] = index_data['Vol_Signal'].shift(1) * index_data['Return']

#cumulative daily returns rate of strategy
data['Vol_Cumulative_Return'] = (1 + data['Vol_Return']).cumprod() - 1
index_data['Vol_Cumulative_Return'] = (1 + index_data['Vol_Return']).cumprod() - 1
```

```
#return is basically profit
#return rate at end of overall period is last value of 'Cumulative_Return'
total_profit_percenatge = (data['Strategy_Return'].tail(495)).sum() * 100
total_profit_percenatge_index = (index_data['Strategy_Return'].tail(495)).sum() * 100
total_profit_percenatge_Mom = (data['Mom_Return'].tail(495)).sum() * 100
total_profit_percenatge_Vol = (data['Vol_Return'].tail(495)).sum() * 100
total_profit_percenatge_Mom_index = (index_data['Mom_Return'].tail(495)).sum() * 100
total_profit_percenatge_Vol_index = (index_data['Vol_Return'].tail(495)).sum() * 100

print('Total Profit of stock using the combined strategy is:', total_profit_percenatge, '%')
print('Total Profit of index using the combined strategy is:', total_profit_percenatge_index, '%')
print('Total Profit of stock using just Momentum Indicator is:', total_profit_percenatge_Mom, '%')
print('Total Profit of index using just Momentum Indicator is:', total_profit_percenatge_Mom_index, '%')
print('Total Profit of stock using just Volatality Indicator is:', total_profit_percenatge_Vol, '%')
print('Total Profit of index using just Volatality Indicator is:', total_profit_percenatge_Vol_index, '%')
```

```
Total Profit of stock using the combined strategy is: -24.84474944210573 %
Total Profit of index using the combined strategy is: 0.47760992596918017 %
Total Profit of stock using just Momentum Indicator is: -105.68136154112095 %
Total Profit of index using just Momentum Indicator is: -40.40652965831859 %
Total Profit of stock using just Volatality Indicator is: -36.91822966198855 %
Total Profit of index using just Volatality Indicator is: 6.627289771988554 %
```

```
#beta
covariance = np.cov(data['Strategy_Return'].tail(495),index_data['Strategy_Return'].tail(495))[0,1]
variance = np.var(index_data['Strategy_Return'].tail(495))
beta = covariance/variance
covariance_Mom = np.cov(data['Mom_Return'].tail(495),index_data['Mom_Return'].tail(495))[0,1]
variance_Mom = np.var(index_data['Mom_Return'].tail(495))
beta_Mom = covariance/variance
covariance_Vol = np.cov(data['Vol_Return'].tail(495),index_data['Vol_Return'].tail(495))[0,1]
variance_Vol = np.var(index_data['Vol_Return'].tail(495))
beta_Vol = covariance/variance

print('Beta for combined strategy:', beta)
print('Beta for just Momentum Indicator:', beta_Mom)
print('Beta for just Volatality Indicator:', beta_Vol)
```

```
Beta for combined strategy: -0.02889682892431965
Beta for just Momentum Indicator: -0.02889682892431965
Beta for just Volatality Indicator: -0.02889682892431965
```

```
#CAPM is per year
risk_free_rate = Rf = 0.05
average_market_return = Rm = index_data['Strategy_Return'].tail(495).mean()
market_premium = Rm-Rf
expected_return = Rf + beta * (market_premium)
average_market_return_Mom = Rm_Mom = index_data['Mom_Return'].tail(495).mean()
market_premium_Mom = Rm_Mom-Rf
expected_return_Mom = Rf + beta_Mom * (market_premium_Mom)
average_market_return_Vol = Rm_Vol = index_data['Vol_Return'].tail(495).mean()
market_premium_Vol = Rm_Vol-Rf
expected_return_Vol = Rf + beta_Vol * (market_premium_Vol)

print('Average Expected Return from combined strategy:', expected_return, '%')
print('Average Expected Return from just Momentum Indicator:', expected_return_Mom, '%')
print('Average Expected Return from just Volatality Indicator:', expected_return_Vol, '%')
```

```
Average Expected Return from combined strategy: 0.051444562629805415 %
Average Expected Return from just Momentum Indicator: 0.05146842974065966 %
Average Expected Return from just Volatality Indicator: 0.05144097260462461 %
```

```
#alpha is for year
average_stock_return = data['Strategy_Return'].tail(495).mean()
alpha = average_stock_return - expected_return
average_stock_return_Mom = data['Mom_Return'].tail(495).mean()
alpha_Mom = average_stock_return_Mom - expected_return_Mom
average_stock_return_Vol = data['Vol_Return'].tail(495).mean()
alpha_Vol = average_stock_return_Vol - expected_return_Vol

print('Alpha from combined strategy:', alpha)
print('Alpha from just Momentum Indicator:', alpha_Mom)
print('Alpha from just Volatality Indicator:', alpha_Vol)
```

```
Alpha from combined strategy: -0.05194647675994896
Alpha from just Momentum Indicator: -0.05360340674149039
Alpha from just Volatality Indicator: -0.05218679542607892
```

```
#sharpe ratio
annual_volatality = data['Strategy_Return'].tail(495).std()
sharpe_ratio = (average_market_return - risk_free_rate) / annual_volatality
annual_volatality_Mom = data['Mom_Return'].tail(495).std()
sharpe_ratio_Mom = (average_market_return_Mom - risk_free_rate) / annual_volatality_Mom
annual_volatality_Vol = data['Vol_Return'].tail(495).std()
sharpe_ratio_Vol = (average_market_return_Vol - risk_free_rate) / annual_volatality_Vol

print('Sharpe Ratio from combined strategy:', sharpe_ratio)
print('Sharpe Ratio from just Momentum Indicator:', sharpe_ratio_Mom)
print('Sharpe Ratio from just Volatality Indicator:', sharpe_ratio_Vol)
```

```
Sharpe Ratio from combined strategy: -3.9294117086793325
Sharpe Ratio from just Momentum Indicator: -1.9744284520497497
Sharpe Ratio from just Volatality Indicator: -3.453735465474106
```

Backtesting

For the Keltner Channels strategy: - If the Close price crosses above the Upper Band, it generates a "Buy" signal. - If the Close price crosses below the Lower Band, it generates a "Sell" signal. - We assume that we buy when a "Buy" signal is generated and sell when a "Sell" signal is generated.

For the Stochastic Oscillator strategy: - If the %K line crosses above the %D line and %K is below 20, it generates a "Buy" signal. - If the %K line crosses below the %D line and %K is

above 80, it generates a "Sell" signal. - We assume that we buy when a "Buy" signal is generated and sell when a "Sell" signal is generated.

Using historical data, the strategy has been backtested to analyse its cumulative return and profitability. The backtesting code provided allows for the evaluation of the strategy's performance over a specified period. By executing the code and analysing the results, you can gain insights into the strategy's historical performance.

Backtesting is a crucial step in assessing the viability and effectiveness of a trading strategy. It involves simulating the strategy's buy and sell decisions based on historical data, considering factors such as entry and exit points, position sizing, transaction costs, and slippage. By applying the strategy to past market conditions, you can evaluate its potential profitability and risk management.

The backtesting process typically involves the following steps:

1. Data Collection: Historical market data, such as price and volume, is gathered for the assets or instruments being traded.
2. Strategy Implementation: The trading strategy is coded based on the defined rules and logic. This code determines the buy and sell signals.
3. Simulation: The strategy is applied to the historical data, simulating the execution of trades according to the predetermined rules. The simulation tracks the portfolio's performance, including profits or losses.
4. Performance Evaluation: The cumulative return and profitability metrics are calculated based on the simulated trades. These metrics provide insights into the strategy's historical performance.

Using historical data, the strategy incorporating the Keltner Channel and Stochastic Oscillator indicators has been backtested. The backtesting process provides insights into the strategy's cumulative return and profitability.

Positive points for the Keltner Channel strategy:

1. Volatility-based signals: The Keltner Channel is a volatility-based indicator that adjusts its bands based on market volatility. This feature allows traders to identify periods of increased volatility, which can present opportunities for capturing price movements and potential breakouts.
2. Trend identification: The Keltner Channel's upper and lower bands can assist traders in identifying trends and potential trend reversals. When the price consistently stays above the upper band, it indicates an uptrend, while prices below the lower band suggest a downtrend. Traders can use these signals to enter or exit positions in line with the prevailing trend.
3. Adaptive to market conditions: The Keltner Channel adjusts its band width based on market volatility, making it adaptable to changing market conditions. This adaptability allows traders to effectively capture price movements across different market environments.

Negative points for the Keltner Channel strategy:

1. Lagging indicator: The Keltner Channel relies on past price data, which makes it a lagging indicator. By the time a signal is generated, a portion of the price move may have already occurred. Traders need to consider this lag when making trading decisions, as it can result in missed opportunities or less optimal entries and exits.
2. False signals: Like any technical indicator, the Keltner Channel can produce false signals, especially during periods of low volatility or choppy price action. Traders should exercise caution and consider additional confirmation or filtering techniques to avoid being misled by false signals.

Positive points for the Stochastic Oscillator strategy:

1. Overbought and oversold conditions: The Stochastic Oscillator is a momentum indicator that helps identify overbought and oversold conditions in the market. When the indicator reaches extreme levels, such as above 80 (overbought) or below 20 (oversold), it suggests potential price reversals. Traders can use these signals to anticipate trend reversals and take advantage of price corrections.
2. Divergence analysis: The Stochastic Oscillator can be used for divergence analysis, which involves comparing the indicator's movement with price action. Divergences between the indicator and the price may indicate a weakening trend or upcoming trend reversal. Traders can use this information to adjust their positions accordingly.

Negative points for the Stochastic Oscillator strategy:

1. False signals: The Stochastic Oscillator, like any oscillator-based indicator, can generate false signals, especially in trending markets or during periods of low volatility. Traders should exercise caution and consider additional confirmation signals or use the Stochastic Oscillator in conjunction with other indicators or analysis techniques.
2. Choppy market conditions: The Stochastic Oscillator may provide less reliable signals during choppy or sideways market conditions. In such situations, the indicator may frequently generate overbought or oversold readings without resulting in meaningful price reversals. Traders should be mindful of market context when using the Stochastic Oscillator as a standalone tool.

Analyses

For Large cap

Let's analyse the provided data for the combined strategy and the individual strategies based on different parameters: total profit, alpha, beta, and Sharpe ratio.

1. Combined Strategy:
    - ➔ Total Profit of stock: -0.2604932655554171%
    - ➔ Total Profit of index: 0.47760992596918017%

- ➔ Beta: 3.6485922845229485e-06
- ➔ Alpha: -0.05000508009580314
- ➔ Average Expected Return: 0.04999981760558989 %
- ➔ Sharpe Ratio: -135.9174558983441

2. Momentum Indicator Only:
   - ➔ Total Profit of stock: -9.07281742830417%
   - ➔ Total Profit of index: -40.40652965831859%
   - ➔ Beta: 3.6485922845229485e-06
   - ➔ Alpha: -0.050183103833039346
   - ➔ Average Expected Return: 0.049999814592063506 %
   - ➔ Sharpe Ratio: -3.277746388972131

3. Volatility Indicator Only:
   - ➔ Total Profit of stock: -0.8017808012066641%
   - ➔ Total Profit of index: 6.627289771988554%
   - ➔ Beta: 3.6485922845229485e-06
   - ➔ Alpha: -0.05001601565081981
   - ➔ Average Expected Return: 0.049999818058876246 %
   - ➔ Sharpe Ratio: -39.082944424651835

Now let's analyse each parameter for both the combined strategy and the individual strategies:

Total Profit:

- The total profit for the combined strategy is -0.2604932655554171% for stocks and 0.47760992596918017% for the index.
- The total profits for the individual momentum and volatility strategies vary. The momentum strategy shows a total profit of -9.07281742830417% for stocks and -40.40652965831859% for the index. The volatility strategy shows a total profit of -0.8017808012066641% for stocks and 6.627289771988554% for the index.

Analysis:

- The combined strategy performs better in terms of total profit compared to the individual momentum strategy, but it lags behind the volatility strategy for the index.
- The individual momentum strategy shows the lowest total profit for both stocks and the index.
- The volatility strategy shows a positive total profit for the index, indicating a better performance compared to the combined strategy.

Alpha:

- The alpha for the combined strategy is -0.05000508009580314.
- The alpha for the individual momentum strategy is -0.050183103833039346.
- The alpha for the individual volatility strategy is -0.05001601565081981.

Analysis:

- All three strategies have negative alphas, indicating that they have underperformed compared to the benchmark.

- There is a slight difference in alpha values between the combined strategy and the individual strategies, but the differences are relatively small.

Beta:

- The beta for the combined strategy, momentum strategy, and volatility strategy is 3.6485922845229485e-06.

Analysis:

- All three strategies have the same beta value, indicating a negligible correlation with the market.
- The beta values are extremely low, suggesting that the strategies are not significantly influenced by market movements.

Average Expected Return:

- The average expected return for the combined strategy is 0.04999981760558989 %.
- The average expected return for the individual momentum strategy is 0.049999814592063506 %.
- The average expected return for the individual volatility strategy is 0.049999818058876246 %.

Analysis:

- The average expected returns for all three strategies are relatively similar.
- The combined strategy has a slightly lower average expected return compared to the individual strategies, but the differences are minimal.

Sharpe Ratio:

- The Sharpe ratio for the combined strategy is -135.9174558983441.
- The Sharpe ratio for the individual momentum strategy is -3.277746388972131.
- The Sharpe ratio for the individual volatility strategy is -39.082944424651835.

Analysis:

- All three strategies show negative Sharpe ratios, indicating poor risk-adjusted returns.
- The combined strategy has an extremely low Sharpe ratio, indicating significantly poor risk-adjusted performance compared to the individual strategies.
- The momentum strategy has a relatively higher Sharpe ratio compared to the combined and volatility strategies, indicating relatively better risk-adjusted returns.

Overall Analysis:

- ★ The combined strategy has not performed well in terms of total profit, alpha, beta, and Sharpe ratio compared to the individual strategies.
- ★ The individual momentum strategy has shown the lowest total profit, while the volatility strategy has performed relatively better.
- ★ All three strategies have negative alphas, indicating underperformance compared to the benchmark.
- ★ The beta values for all strategies are extremely low, indicating minimal correlation with the market.

★ None of the strategies have positive Sharpe ratios, suggesting poor risk-adjusted returns.
★ The momentum strategy has relatively better risk-adjusted returns compared to the combined and volatility strategies.

Based on this analysis, both the combined strategy and the individual strategies have shown subpar performance in terms of total profit, alpha, beta, and Sharpe ratio. Further evaluation and refinement of the strategies may be necessary to improve their effectiveness and suitability for investment purposes.

For small cap

Based on the provided data, let's analyse the combined strategy and the individual strategies based on total profit, alpha, beta, average expected return, and Sharpe ratio:

1. Combined Strategy:
    ➔ Total Profit of stock: -24.84474944210573%
    ➔ Total Profit of index: 0.47760992596918017%
    ➔ Beta: -0.02889682892431965
    ➔ Alpha: -0.05194647675994896
    ➔ Average Expected Return: 0.05144456262980541 5%
    ➔ Sharpe Ratio: -3.9294117086793325

2. Momentum Indicator Only:
    ➔ Total Profit of stock: -105.68136154112095%
    ➔ Total Profit of index: -40.40652965831859%
    ➔ Beta: -0.02889682892431965
    ➔ Alpha: -0.05360340674149039
    ➔ Average Expected Return: 0.05146842974065966%
    ➔ Sharpe Ratio: -1.9744284520497497

3. Volatility Indicator Only:
    ➔ Total Profit of stock: -36.91822966198855%
    ➔ Total Profit of index: 6.627289771988554%
    ➔ Beta: -0.02889682892431965
    ➔ Alpha: -0.05218679542607892
    ➔ Average Expected Return: 0.05144097260462461%
    ➔ Sharpe Ratio: -3.453735465474106

Now let's analyze each parameter for both the combined strategy and the individual strategies:

Total Profit:

- The total profit for the combined strategy is -24.84474944210573% for stocks and 0.47760992596918017% for the index.

- The total profits for the individual momentum and volatility strategies vary. The momentum strategy shows a total profit of -105.68136154112095% for stocks and -40.40652965831859% for the index. The volatility strategy shows a total profit of -36.91822966198855% for stocks and 6.627289771988554% for the index.

Analysis:
- The combined strategy shows a negative total profit for stocks, indicating a loss, while the index has a positive total profit.
- Both the individual momentum and volatility strategies show negative total profits for stocks, with the momentum strategy having a larger loss. However, the volatility strategy shows a positive total profit for the index.

Alpha:
- The alpha for the combined strategy is -0.05194647675994896.
- The alpha for the individual momentum strategy is -0.05360340674149039.
- The alpha for the individual volatility strategy is -0.05218679542607892.

Analysis:
- All three strategies have negative alphas, indicating underperformance compared to the benchmark.
- The combined strategy has a slightly lower alpha compared to the individual strategies, but the differences are relatively small.

Beta:
- The beta for the combined strategy, momentum strategy, and volatility strategy is -0.02889682892431965.

Analysis:
- All three strategies have the same beta value, indicating a negative correlation with the market.
- The beta values are negative, suggesting that the strategies have performed differently from the market.

Average Expected Return:
- The average expected return for the combined strategy is 0.051444562629805415%.
- The average expected return for the individual momentum strategy is 0.05146842974065966%.
- The average expected return for the individual volatility strategy is 0.05144097260462461%.

Analysis:
- The average expected returns for all three strategies are relatively similar.
- The combined strategy has a slightly lower average expected return compared to the individual strategies, but the differences are minimal.

Sharpe Ratio:
- The Sharpe ratio for the combined strategy is -3.9294117086793325.
- The Sharpe ratio for the individual momentum strategy is -1.9744284520497497.

- The Sharpe ratio for the individual volatility strategy is -3.453735465474106.

Analysis:
- The Sharpe ratios for all three strategies are negative, indicating a relatively poor risk-adjusted performance.
- The combined strategy has a lower Sharpe ratio compared to the individual strategies, suggesting higher risk or lower returns relative to the risk taken.

Overall Analysis:
- ★ The combined strategy shows a negative total profit for stocks and a positive total profit for the index, indicating mixed performance.
- ★ Both the combined strategy and the individual strategies have negative alphas, suggesting underperformance compared to the benchmark.
- ★ The beta values are negative for all strategies, indicating a negative correlation with the market.
- ★ The average expected returns are relatively similar for all strategies.
- ★ The Sharpe ratios for all strategies are negative, indicating a relatively poor risk-adjusted performance.

Inferences

SWOT analysis (Strengths, Weaknesses, Opportunities, and Threats) for the strategy incorporating the Keltner Channel and Stochastic Oscillator:

Strengths:
1. Objective decision-making: The strategy utilises objective technical indicators, namely the Keltner Channel and Stochastic Oscillator, which can help remove emotional biases from trading decisions.
2. Volatility-based signals: The Keltner Channel identifies periods of increased volatility, allowing traders to capture potential breakouts and trend reversals.
3. Overbought/oversold identification: The Stochastic Oscillator helps identify overbought and oversold conditions, indicating potential price reversals and providing entry/exit signals.

Weaknesses:
1. Lagging indicators: Both the Keltner Channel and Stochastic Oscillator are lagging indicators, which means they rely on past price data. This lag can result in delayed signals, potentially causing missed trading opportunities or suboptimal entries/exits.
2. False signals: As with any technical indicators, the Keltner Channel and Stochastic Oscillator can generate false signals, leading to losing trades or missed profit opportunities. Additional confirmation or filtering techniques may be required to mitigate this weakness.

Opportunities:

1. Trend identification: The strategy can be advantageous in trending markets, where the Keltner Channel and Stochastic Oscillator can help traders identify and participate in the prevailing trend.
2. Diverse trading signals: By combining the Keltner Channel and Stochastic Oscillator, the strategy provides a broader range of trading signals, capturing both volatility-based and momentum-based opportunities.

Threats:

1. Market conditions: The strategy's effectiveness may be influenced by different market conditions. It could face challenges during choppy or sideways markets, where false signals may be more prevalent, leading to losses or reduced profitability.
2. Parameter selection: The strategy's performance can vary based on the selection of parameters for the Keltner Channel and Stochastic Oscillator. Improper parameter choices may impact the strategy's accuracy and overall performance.
3. Over-optimization: There is a risk of over-optimizing the strategy based on historical data, which may lead to poor performance in real-time market conditions. Traders should be cautious about fine-tuning the strategy excessively to fit past data.

Overall, the strategies have not performed well, with the combined strategy showing mixed results and the individual strategies also experiencing losses or underperformance compared to the benchmark. Improvement or modification of the strategies may be necessary to achieve better results.

Proposed Improvements:

Proposed improvements for the Keltner Channel strategy:

1. Dynamic bandwidth: Instead of using fixed band widths for the Keltner Channel, consider implementing a dynamic approach that adjusts the band widths based on the current market conditions. This can help capture changes in volatility more accurately and provide more reliable signals.
2. Incorporate trend confirmation: Enhance the strategy by adding additional indicators or techniques to confirm the prevailing trend before entering trades based on the Keltner Channel signals. This can help filter out false signals and improve the overall accuracy of the strategy.
3. Optimise parameters: Perform thorough parameter optimization to determine the optimal settings for the Keltner Channel. This process should involve testing different periods and deviation values to identify the most effective combination for the specific market and timeframe being traded.

Proposed improvements for the Stochastic Oscillator strategy:

1. Use multiple timeframes: Incorporate multiple timeframes when analyzing the Stochastic Oscillator signals. By considering signals from different timeframes, you

can gain a broader perspective on the market and potentially improve the accuracy of entry and exit decisions.
2. Filter out false signals: Combine the Stochastic Oscillator with additional filters or confirmation indicators to reduce the impact of false signals. For example, you could incorporate a trend-following indicator to confirm the prevailing trend before taking trades based on Stochastic Oscillator readings.
3. Adjust oversold/overbought thresholds: Experiment with different overbought and oversold threshold levels to find the values that work best for the specific market and timeframe. This can help fine-tune the strategy and improve its performance.

Some additional improvements that can be considered for both the Keltner Channel strategy and the Stochastic Oscillator strategy:

1. Incorporate additional indicators: To enhance the reliability of the strategies, consider integrating other technical indicators that complement the Keltner Channel and Stochastic Oscillator. For example, trend-following indicators like moving averages or trend confirmation indicators like the MACD (Moving Average Convergence Divergence) can provide additional insights and improve decision-making.
2. Implement a trailing stop-loss: Introduce a trailing stop-loss mechanism to protect profits and allow for potential further upside in winning trades. This feature can help capture larger price moves and mitigate the risk of significant reversals.
3. Fine-tune parameter selection: Continuously refine and optimize the parameters of the Keltner Channel and Stochastic Oscillator to suit different market conditions. Regularly reevaluate and test various parameter combinations to ensure they remain effective and adaptive to changing market dynamics.
4. Introduce risk management rules: Implement strict risk management rules, such as position sizing based on risk tolerance and setting predefined stop-loss levels. Incorporate these rules to manage potential losses and protect capital.
5. Consider different timeframes: Explore the performance of the strategies on multiple timeframes to capture opportunities across different trading horizons. This can provide a more comprehensive understanding of the strategy's effectiveness and adaptability.
6. Combine with fundamental analysis: Supplement the technical indicators with fundamental analysis to gain a broader perspective on the underlying market factors. Incorporate relevant fundamental data, news releases, or economic events into the decision-making process to make more informed trading decisions.
7. Ongoing evaluation and optimization: Regularly review the performance of the strategies, assess the impact of implemented improvements, and make necessary adjustments. Market conditions change over time, so continuous monitoring and refinement are essential for maintaining the strategies' effectiveness.

To conclude, we believe that our collaborative efforts, effective communication, and equal participation were vital in successfully completing the assignment. The report and well-documented Python code demonstrate our understanding of the trading strategies, technical indicators, and their impact on performance. We are confident that our findings and proposed improvements will contribute to the development of effective trading strategies using technical indicators.