Estrategia de Trading con Pares, Reversiones, Cobertura y Uso del Índice

Respuesta a la Pregunta sobre el Uso del Índice

Después de tu pregunta sobre si el índice fue utilizado, se detectó que el índice no fue inicialmente incorporado en el análisis. A continuación, se proporciona una mejora en la estrategia para incluir el índice, mejorando el rendimiento potencial y la cobertura de riesgo.

Explicación del Enfoque Mejorado

Se ha modificado la estrategia para utilizar el índice como una referencia del mercado para validar las señales de trading entre las dos acciones. Se utilizó un enfoque basado en medias móviles para detectar la tendencia del mercado con el índice:

- Si el índice está en una tendencia alcista, se toman señales más agresivas.
- Si el índice está en una tendencia bajista, se filtran las señales de compra para reducir el riesgo.

Código Modificado para Incluir el Índice

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt



# === 1. Cargar los datos ===

df = pd.read_csv('datos_trading.csv', parse_dates=['Fecha'], index_col='Fecha')



# === 2. Calcular el spread entre las dos acciones ===

df['Spread'] = df['Accion1'] - df['Accion2']
```

Estrategia de Trading con Pares, Reversiones, Cobertura y Uso del Índice

```
# === 3. Calcular el Z-Score del spread ===

def calculate_zscore(spread, window=30):

    mean = spread.rolling(window=window).mean()

    std = spread.rolling(window=window).std()

    return (spread - mean) / std


df['Z-Score'] = calculate_zscore(df['Spread'])


# === 4. Análisis del Índice ===

df['Indice_SMA_50'] = df['Indice'].rolling(window=50).mean()

df['Indice_SMA_200'] = df['Indice'].rolling(window=200).mean()

df['Market_Trend'] = np.where(df['Indice_SMA_50'] > df['Indice_SMA_200'], 1, -1)


# === 5. Señales de Trading ===

df['Signal'] = np.where((df['Z-Score'] > 2) & (df['Market_Trend'] == 1), -1,

                            np.where((df['Z-Score'] < -2) & (df['Market_Trend'] == 1), 1,

0))


# === 6. Cobertura con Divisas ===

df['Hedge'] = np.where(df['Signal'] == 1, -df['ParDivisas'].pct_change().shift(-1),

                                                          np.where(df['Signal']   ==   -1,

df['ParDivisas'].pct_change().shift(-1), 0))


# === 7. Evaluar el rendimiento ===

df['Strategy_Return']     =     df['Signal']     *     (df['Accion1'].pct_change()     -
```

```
df['Accion2'].pct_change())

df['Hedged_Return'] = df['Strategy_Return'] - df['Hedge']

df['Cumulative_Return'] = (1 + df['Hedged_Return']).cumprod()



# === 8. Visualizar el rendimiento ===

plt.plot(df['Cumulative_Return'], label='Rendimiento Acumulado')

plt.legend()

plt.show()
```

Posibles Mejoras Futuras

La estrategia presentada es sólida, pero siempre existen áreas para mejorar:

1. **Optimización de Parámetros**: Ajustar las ventanas de cálculo para obtener mejores señales.

2. **Indicadores Técnicos Adicionales**: Incorporar RSI, MACD o Bandas de Bollinger.

3. **Gestión Dinámica del Tamaño de la Posición**: Usar ATR para ajustar el tamaño de la posición según la volatilidad.

4. **Uso de Machine Learning**: Implementar modelos para mejorar la precisión de las señales.

5. **Cobertura Mejorada**: Usar futuros o derivados para una cobertura más eficiente.

6. **Backtesting y Evaluación Avanzada**: Evaluar con técnicas de Monte Carlo y métricas adicionales como el Sortino Ratio.