Resumen Profundo para Identificar Patrones de Velas de Rechazo/Retrocesos en Gráficos de Criptomonedas

Basado en el contenido del libro "Price Action Trading Secrets" de Rayner Teo, aquí tienes un análisis profundo con ideas implementables para identificar patrones de velas de rechazo y retrocesos:

Conceptos Fundamentales Extraídos del Libro

1. Esencia de los Patrones de Velas de Rechazo

- **No memorizar patrones**, sino entender QUÉ representan
- **Dos preguntas clave** para cualquier vela:

- ¿Dónde cerró el precio relativo al rango?
- ¿Cuál es el tamaño de la vela comparado con las anteriores?

2. Características de las Velas de Rechazo

- **Sombra larga** en la dirección del rechazo
- **Cuerpo pequeño** relativo a las sombras
- **Cierre cerca del extremo opuesto** al rechazo

Patrones Clave para Implementar

1. Martillo (Hammer) - Rechazo Bajista

```python
# Características detectables:
- Ocurre después de una caída de precio
- Sombra inferior ≥ 2x el tamaño del cuerpo
- Poca o ninguna sombra superior
- Cierre en el 25% superior del rango
```

2. Estrella Fugaz (Shooting Star) - Rechazo Alcista

python

```
# Características detectables:
- Ocurre después de un avance de precio
- Sombra superior ≥ 2x el tamaño del cuerpo
- Poca o ninguna sombra inferior
- Cierre en el 25% inferior del rango
```

3. Patrón de Engulfing (Alcista o Bajista)

python

```
# Características detectables:
- Dos velas consecutivas
- El cuerpo de la segunda vela "envuelve" completamente el cuerpo de la primera
- Cierre en dirección opuesta a la tendencia previa
```

Algoritmos Sencillos para Implementar

1. Detección Básica de Rechazo

python

```
def detectar_rechazo(open, high, low, close):
  rango_total = high - low
  cuerpo = abs(close - open)
  sombra_superior = high - max(open, close)
  sombra_inferior = min(open, close) - low

  # Cálculo de proporciones
  proporcion_sombra_sup = sombra_superior / rango_total
  proporcion_sombra_inf = sombra_inferior / rango_total
  proporcion_cuerpo = cuerpo / rango_total

  # Detección de martillo (rechazo bajista)
  if (proporcion_sombra_inf > 0.6 and
    proporcion_cuerpo < 0.3 and
    close > (low + rango_total * 0.6)):
    return "MARTILLO"

  # Detección de estrella fugaz (rechazo alcista)
```

```
  if (proporcion_sombra_sup > 0.6 and
    proporcion_cuerpo < 0.3 and
    close < (high - rango_total * 0.6)):
    return "ESTRELLA_FUGAZ"

  return "SIN_RECHAZO"
```

2. Detección de Fuerza en el Movimiento

python

```
def fuerza_movimiento(velas_actual, velas_anteriores):
  """
  Compara el tamaño de la vela actual con las anteriores
  para detectar fuerza/convicción
  """
  tamaño_actual = velas_actual['high'] - velas_actual['low']
  tamaños_anteriores = [v['high'] - v['low'] for v in velas_anteriores]
  promedio_anteriores = sum(tamaños_anteriores) / len(tamaños_anteriores)

  if tamaño_actual >= promedio_anteriores * 1.5:
    return "FUERTE"
  elif tamaño_actual <= promedio_anteriores * 0.7:
    return "DEBIL"
  else:
    return "NORMAL"
```

3. Análisis de Contexto de Mercado

python

```
def contexto_mercado(velas, periodo=20):
  """
  Determina si el mercado está en tendencia o rango
  """
  highs = [v['high'] for v in velas]
  lows = [v['low'] for v in velas]

  max_high = max(highs[-periodo:])
  min_low = min(lows[-periodo:])
  rango_total = max_high - min_low

  # Si el rango es pequeño comparado con el precio, probablemente está en consolidación
  precio_promedio = sum([v['close'] for v in velas[-periodo:]]) / periodo
```

```
    proporcion_rango = rango_total / precio_promedio

    if proporcion_rango < 0.02:  # 2% del precio
        return "RANGO"
    else:
        return "TENDENCIA"
```

Estrategia de Implementación Completa

1. Pipeline de Detección

python

```
def pipeline_deteccion_patrones(velas, lookback=50):
    resultados = {}

    # 1. Contexto del mercado
    resultados['contexto'] = contexto_mercado(velas)

    # 2. Velas de rechazo individuales
    vela_actual = velas[-1]
    resultados['rechazo'] = detectar_rechazo(
        vela_actual['open'],
        vela_actual['high'],
        vela_actual['low'],
        vela_actual['close']
    )

    # 3. Fuerza del movimiento
    resultados['fuerza'] = fuerza_movimiento(
        vela_actual,
        velas[-10:-1]  # Últimas 10 velas excluyendo la actual
    )

    # 4. Detección de patrones multi-vela
    if len(velas) >= 3:
        resultados['engulfing'] = detectar_engulfing(velas[-2:])
        resultados['tweezer'] = detectar_tweezer(velas[-2:])

    return resultados
```

2. Filtros de Confirmación

python

```python
def filtrar_señales(deteccion, niveles_soporte_resistencia):
  """
  Filtra señales basado en contexto y niveles clave
  """
  señales_validas = []

  # Solo operar rechazos en áreas de valor
  precio_actual = deteccion['precio_actual']

  for nivel in niveles_soporte_resistencia:
    distancia_porcentual = abs(precio_actual - nivel) / precio_actual

    if (deteccion['rechazo'] != "SIN_RECHAZO" and
      distancia_porcentual < 0.02):  # Dentro del 2% del nivel
      señal = {
        'patron': deteccion['rechazo'],
        'nivel': nivel,
        'tipo_nivel': 'SOPORTE' if precio_actual > nivel else 'RESISTENCIA',
        'confianza': calcular_confianza(deteccion)
      }
      señales_validas.append(señal)

  return señales_validas
```

Recomendaciones para tu Proyecto

1. Enfoque por Capas

- **Capa 1**: Detección básica de patrones individuales
- **Capa 2**: Análisis de contexto y múltiples timeframes
- **Capa 3**: Filtrado por áreas de valor (soporte/resistencia)

2. Parámetros Ajustables

- Proporciones de sombras/cuerpo configurables
- Sensibilidad a la distancia de niveles de soporte/resistencia
- Períodos de lookback personalizables

3. Visualización

•Resaltar velas de rechazo detectadas
•Mostrar áreas de soporte/resistencia
•Indicar fuerza del patrón (color/intensidad)

La clave, como enfatiza Rayner Teo, es entender la **historia que cuenta cada vela** más que memorizar patrones específicos. Implementa estos conceptos de manera flexible y permite ajustes basados en el comportamiento específico de las criptomonedas que estés analizando.

==========

Patrones de Velas de Rechazo Clave

1. Martillo (Hammer)
•**Apariencia**: Cuerpo pequeño en la parte superior, sombra inferior larga (al menos el doble del cuerpo) y poca o ninguna sombra superior.
•**Significado**: Rechazo de precios bajos, indica posible reversión alcista después de una caída.
•**Implementación**: Buscar velas con:
•Sombra inferior ≥ 2 * cuerpo
•Cierre en la mitad superior del rango
•Preferiblemente en una tendencia bajista o cerca de soporte.

2. Estrella Fugaz (Shooting Star)
•**Apariencia**: Cuerpo pequeño en la parte inferior, sombra superior larga (al menos el doble del cuerpo) y poca o ninguna sombra inferior.
•**Significado**: Rechazo de precios altos, indica posible reversión bajista después de un avance.
•**Implementación**: Buscar velas con:
•Sombra superior ≥ 2 * cuerpo
•Cierre en la mitad inferior del rango
•Preferiblemente en una tendencia alcista o cerca de resistencia.

3. Patrón de Engulfing Alcista (Bullish Engulfing)
•**Apariencia**: Dos velas. La primera es bajista (cierra por debajo de la apertura) y la segunda es alcista (cierra por encima de la apertura) y su cuerpo cubre completamente el cuerpo de la primera vela.
•**Significado**: Los compradores superan a los vendedores, reversión alcista.
•**Implementación**:
•Primera vela: cuerpos bajista (cierre < apertura)
•Segunda vela: cuerpo alcista (cierre > apertura) y el cuerpo de la segunda vela debe envolver completamente el cuerpo de la primera (sin considerar las sombras).

4. Patrón de Engulfing Bajista (Bearish Engulfing)

•**Apariencia**: Dos velas. La primera es alcista (cierra por encima de la apertura) y la segunda es bajista (cierra por debajo de la apertura) y su cuerpo cubre completamente el cuerpo de la primera vela.
•**Significado**: Los vendedores superan a los compradores, reversión bajista.
•**Implementación**:
•Primera vela: cuerpo alcista (cierre > apertura)
•Segunda vela: cuerpo bajista (cierre < apertura) y el cuerpo de la segunda vela debe envolver completamente el cuerpo de la primera.

5. Tweezer Bottom (Fondo de Pinzas)

•**Apariencia**: Dos velas con mínimos iguales o muy similares. La primera vela es bajista y la segunda es alcista (o muestra rechazo).
•**Significado**: Rechazo de precios bajos, posible reversión alcista.
•**Implementación**:
•Dos velas consecutivas con mínimos muy cercanos (dentro de un umbral).
•La primera vela es bajista y la segunda muestra rechazo (puede ser un martillo, una vela alcista, etc.).

6. Tweezer Top (Top de Pinzas)

•**Apariencia**: Dos velas con máximos iguales o muy similares. La primera vela es alcista y la segunda es bajista (o muestra rechazo).
•**Significado**: Rechazo de precios altos, posible reversión bajista.
•**Implementación**:
•Dos velas consecutivas con máximos muy cercanos (dentro de un umbral).
•La primera vela es alcista y la segunda muestra rechazo (puede ser una estrella fugaz, una vela bajista, etc.).

Ideas para la Implementación en Python

1. Estructura de Datos para Velas

Cada vela debe tener: timestamp, open, high, low, close.

2. Funciones de Detección de Patrones Individuales

python

```python
def is_hammer(open, high, low, close):
    body = abs(close - open)
    lower_shadow = min(open, close) - low
    upper_shadow = high - max(open, close)
    # Condiciones para martillo
    if lower_shadow >= 2 * body and upper_shadow <= 0.1 * body:
        return True
    return False
```

```python
def is_shooting_star(open, high, low, close):
    body = abs(close - open)
    lower_shadow = min(open, close) - low
    upper_shadow = high - max(open, close)
    # Condiciones para estrella fugaz
    if upper_shadow >= 2 * body and lower_shadow <= 0.1 * body:
        return True
    return False

def is_bullish_engulfing(prev_open, prev_close, curr_open, curr_close):
    # Primera vela bajista, segunda vela alcista y cubre el cuerpo de la primera
    if prev_close < prev_open and curr_close > curr_open:
        if curr_open < prev_close and curr_close > prev_open:
            return True
    return False

def is_bearish_engulfing(prev_open, prev_close, curr_open, curr_close):
    # Primera vela alcista, segunda vela bajista y cubre el cuerpo de la primera
    if prev_close > prev_open and curr_close < curr_open:
        if curr_open > prev_close and curr_close < prev_open:
            return True
    return False

def is_tweezer_bottom(prev_low, curr_low, prev_close, curr_close, threshold=0.001):
    # Los mínimos son muy cercanos y la primera vela es bajista, la segunda alcista o con rechazo
    if abs(prev_low - curr_low) <= threshold and prev_close < prev_open and curr_close > curr_open:
        return True
    return False

def is_tweezer_top(prev_high, curr_high, prev_close, curr_close, threshold=0.001):
    # Los máximos son muy cercanos y la primera vela es alcista, la segunda bajista o con rechazo
    if abs(prev_high - curr_high) <= threshold and prev_close > prev_open and curr_close < curr_open:
        return True
    return False
```