QuantNote: Metodologia de Cálculo de Probabilidades Condicionais para Opções

Versão 1.0 | Dezembro 2024

Sumário Executivo

Este documento descreve a metodologia desenvolvida no projeto QuantNote para calcular probabilidades de atingimento de preços-alvo em ativos financeiros. A abordagem combina técnicas de Machine Learning (K-Means clustering) com otimização por Algoritmo Genético para superar as limitações das probabilidades incondicionais tradicionais.

O sistema calcula duas métricas fundamentais:

- **P(fechar)**: Probabilidade de o preço **fechar** em ou acima do alvo no vencimento
- **P(tocar)**: Probabilidade de o preço **tocar** o alvo em algum momento até o vencimento

---

1. Introdução: O Problema das Probabilidades Incondicionais

1.1 Definição de Probabilidade Incondicional

A forma mais simples de calcular a probabilidade de um ativo atingir determinado preço é através da **probabilidade incondicional** (ou frequentista):

```
P(retorno > alvo) = (número de ocorrências históricas onde retorno > alvo)
/ (total de observações)
```

Exemplo: Se em 1000 dias de negociação, o ativo subiu mais de 5% em 7 dias, temos:

- P(retorno > 5% em H dias) = 150/1000 = 15%

1.2 Limitações da Abordagem Incondicional

A probabilidade incondicional apresenta sérias limitações para tomada de decisão:

1.2.1 Ignora o Contexto de Mercado

O mercado não é estacionário. Um ativo em forte tendência de alta tem probabilidades muito diferentes de um ativo em consolidação. A probabilidade incondicional trata todos os dias como equivalentes.

1.2.2 Média de Cenários Heterogêneos

Ao calcular uma média simples, misturamos:

- Períodos de alta volatilidade com baixa volatilidade

- Tendências de alta com tendências de baixa
- Momentos de consolidação com breakouts

1.2.3 Não Captura Regimes de Mercado

Mercados financeiros exibem **regimes** distintos (bull market, bear market, sideways). A probabilidade incondicional é uma média ponderada desses regimes, sendo subótima para qualquer um deles individualmente.

1.3 Motivação para Probabilidades Condicionais

A solução é calcular **P(alvo | regime atual)** - a probabilidade de atingir o alvo **dado** o regime de mercado em que nos encontramos.

```
P(alvo | regime) ≠ P(alvo)
```

Se conseguirmos identificar corretamente o regime atual, podemos obter estimativas de probabilidade mais precisas e acionáveis.

---

2. Métricas de Probabilidade: P(fechar) vs P(tocar)

2.1 P(fechar) - Probabilidade de Fechamento

Definição: Probabilidade de o preço de **fechamento** no final do horizonte H estar em ou acima do alvo.

```
# Cálculo do retorno futuro (close-to-close)
log_return_future = log(close[t+H] / close[t])

# Hit se retorno >= alvo
hit = 1 se log_return_future >= log(1 + alvo) else 0
```

Uso: Precificação de opções europeias (exercício apenas no vencimento).

2.2 P(tocar) - Probabilidade de Toque

Definição: Probabilidade de o preço **tocar** o alvo em **qualquer momento** durante o horizonte H.

```
# Para alvos positivos (calls): máximo do HIGH na janela
max_high = max(high[t+1:t+H+1])
log_return_touch_max = log(max_high / close[t])

# Para alvos negativos (puts): mínimo do LOW na janela
min_low = min(low[t+1:t+H+1])
log_return_touch_min = log(min_low / close[t])
```

Uso: Precificação de opções com barreiras, análise de stop-loss/take-profit.

2.3 Relação Matemática

Por definição:

```
P(tocar) ≥ P(fechar)
```

Isso ocorre porque:

- Se o preço **fechou** acima do alvo, necessariamente ele **tocou** o alvo em algum momento
- Porém, o preço pode **tocar** o alvo e depois recuar, **não fechando** acima

A diferença `P(tocar) - P(fechar)` representa a probabilidade de "toque sem permanência".

---

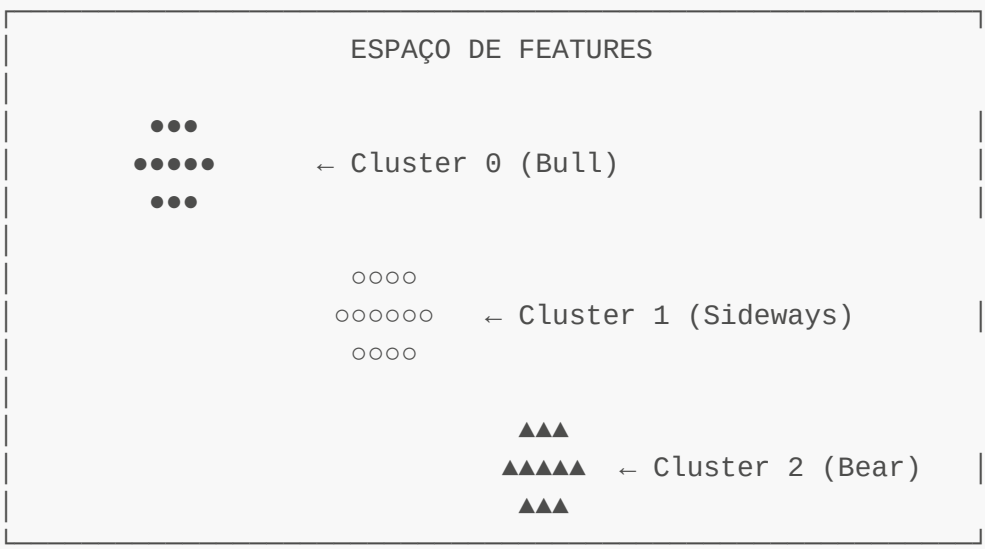
3. Identificação de Regimes com K-Means Clustering

3.1 O que é K-Means?

K-Means é um algoritmo de aprendizado não-supervisionado que agrupa dados em K clusters baseado na similaridade das características (features).

Funcionamento:

1. Inicializa K centróides aleatoriamente
2. Atribui cada ponto ao centróide mais próximo
3. Recalcula os centróides como média dos pontos de cada cluster
4. Repete passos 2-3 até convergência



3.2 Por que K-Means para Regimes de Mercado?

1. **Sem viés de supervisão**: Não precisamos rotular previamente os regimes

2. **Adaptativo**: Clusters se ajustam aos dados específicos de cada ativo
3. **Interpretável**: Podemos analisar os centróides para entender cada regime
4. **Eficiente**: Complexidade O(n × K × i), onde i = iterações

3.3 Features Selecionadas

Após extensivos testes com **Random Forest** e **Árvore de Decisão** (descritos na Seção 5), selecionamos três features principais:

Feature	Descrição	Intuição
Slope	Inclinação da regressão linear do preço	Direção da tendência
MA Distance	Distância entre médias móveis curta e longa	Força da tendência
Volatility	Desvio padrão dos retornos	Regime de volatilidade

```
Regime = f(slope, ma_distance, volatility)
```

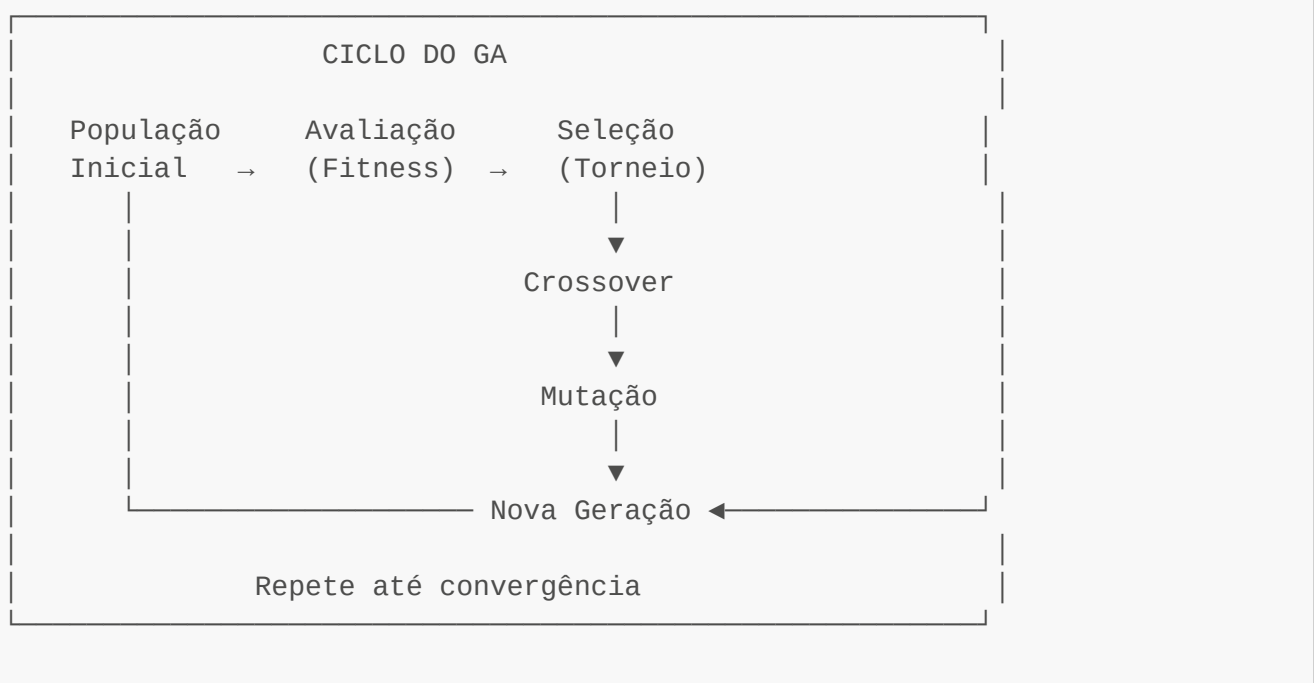
---

4. Otimização por Algoritmo Genético

4.1 O que é Algoritmo Genético?

Algoritmo Genético (GA) é uma metaheurística inspirada na evolução biológica. Mantém uma "população" de soluções candidatas que evoluem ao longo de "gerações" através de:

1. **Seleção**: Soluções mais aptas têm maior chance de reprodução
2. **Crossover**: Combinação de características de dois "pais"
3. **Mutação**: Alterações aleatórias para manter diversidade



4.2 Cromossomo: Representação da Solução

Cada indivíduo (cromossomo) representa um conjunto de parâmetros:

```
@dataclass
class Chromosome:
    # Parâmetros de janela
    window_slope: int          # Janela para cálculo do slope (5-60)
    window_volatility: int     # Janela para volatilidade (5-60)
    window_rolling_return: int # Janela para retorno rolling (5-30)

    # Parâmetros de clustering
    n_clusters: int            # Número de clusters (2-5)

    # Flags de features
    use_volatility: bool       # Incluir volatilidade?
    use_rolling_return: bool   # Incluir retorno rolling?
    use_ma_distance: bool      # Incluir distância de MAs?

    # Parâmetros de MA Distance
    ma_fast_period: int        # Período da MA rápida (5-20)
    ma_slow_period: int        # Período da MA lenta (20-60)
```

4.3 Função Fitness

A função fitness mede a qualidade de um cromossomo. Nosso objetivo é **maximizar a separação entre regimes** (delta_p) enquanto penalizamos overfitting.

```
fitness = delta_p_test - stability_penalty - overfit_penalty +
consistency_bonus
```

Onde:

4.3.1 Delta P (Separação entre Regimes)

```
delta_p = max(P_cluster) - min(P_cluster)
```

Mede quão diferentes são as probabilidades entre o melhor e o pior regime. Quanto maior, melhor a capacidade preditiva.

4.3.2 Stability Penalty

```
stability_penalty = λ × (número de mudanças de regime / total de
observações)
```

Penaliza regimes que mudam muito frequentemente (noise).

4.3.3 Overfitting Penalty

```
overfitting_penalty = μ × max(0, overfitting_ratio - 1.5)
```

Onde `overfitting_ratio = delta_p_train / delta_p_test`. Penaliza quando o modelo performa muito melhor no treino que no teste.

4.3.4 Consistency Bonus

```
consistency_bonus = max(0, 0.1 - std_delta_p_test)
```

Bonifica modelos consistentes ao longo dos folds de validação.

4.4 Walk-Forward Validation

Para evitar overfitting, usamos **validação walk-forward**:

```
┌─────────────────────────────────────────────────────────────┐
│                 WALK-FORWARD VALIDATION                     │
│                                                             │
│ Fold 1: [======TRAIN======][TEST]                           │
│ Fold 2:    [======TRAIN======][TEST]                        │
│ Fold 3:       [======TRAIN======][TEST]                     │
│ Fold 4:          [======TRAIN======][TEST]                  │
│ Fold 5:             [======TRAIN======][TEST]               │
│                                                             │
│ • Treino sempre ANTES do teste (sem look-ahead)             │
│ • K-Means fitado no treino, aplicado no teste               │
│ • Métricas calculadas out-of-sample                         │
└─────────────────────────────────────────────────────────────┘
```

4.5 Early Stopping

O GA implementa **early stopping** para eficiência:

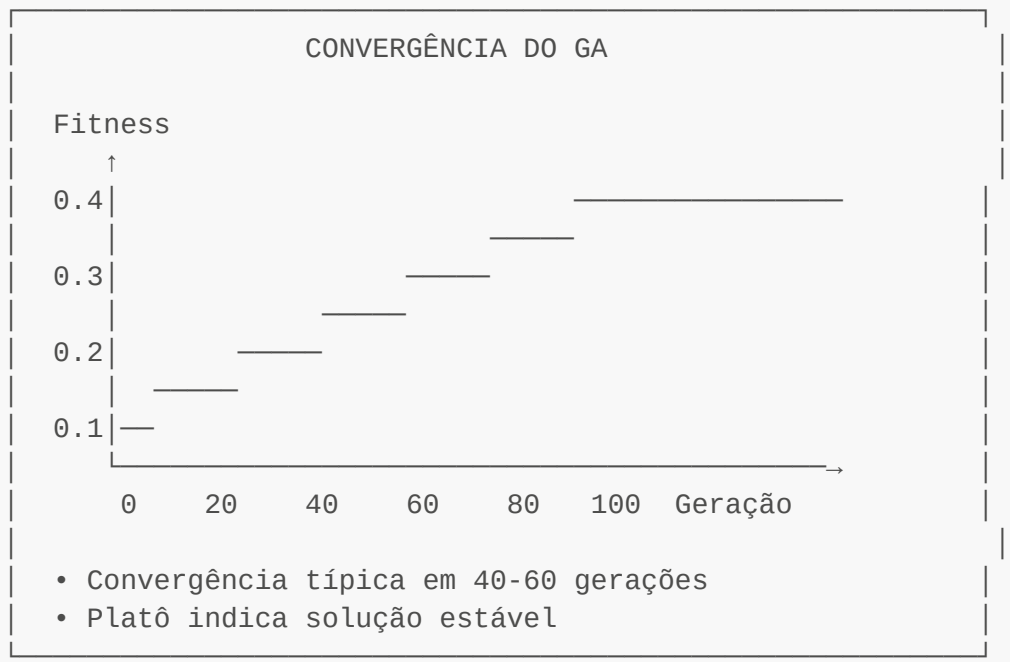
```
early_stopping_patience = 20    # Gerações sem melhoria
early_stopping_threshold = 0.001  # Melhoria mínima

if (fitness_atual - fitness_anterior) < threshold:
    contador += 1
    if contador >= patience:
        PARAR
```

Critérios:

- Parar se não houver melhoria de 0.1% em 20 gerações consecutivas
- Evita desperdício computacional quando o modelo convergiu

4.6 Processo de Convergência



---

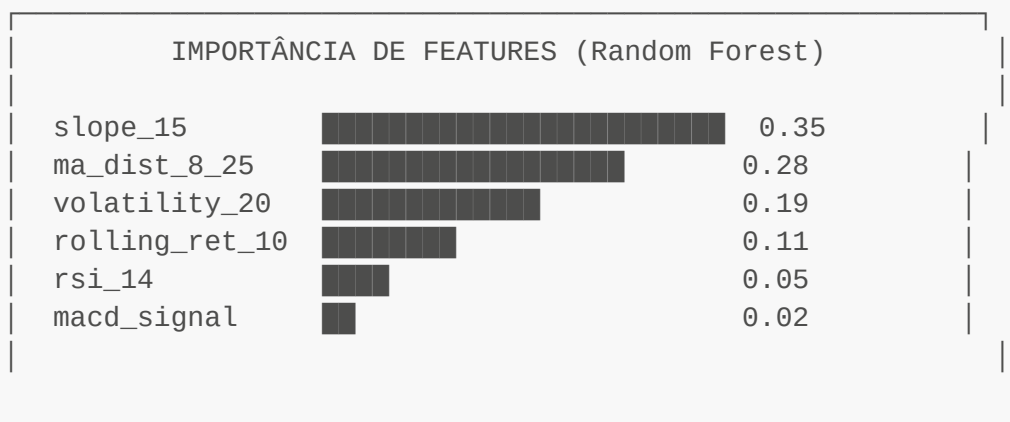
5. Seleção de Features: Random Forest e Árvore de Decisão

5.1 Random Forest

Random Forest é um ensemble de árvores de decisão que:

- Treina múltiplas árvores em subconjuntos aleatórios dos dados (bagging)
- Cada árvore vota na classificação final
- Reduz overfitting pela agregação

Uso no QuantNote: Identificamos a **importância das features** para prever o regime.



```
│  → Top 3 features explicam 82% da variância                  │
└──────────────────────────────────────────────────────────────┘
```

5.2 Árvore de Decisão

Árvore de Decisão cria regras hierárquicas do tipo "se-então":

```
                  [slope > 0.02?]
                  /              \
               Sim                Não
               /                    \
     [volatility > 0.15?]      [ma_dist < -0.01?]
     /            \              /           \
  Bull         Sideways       Bear        Sideways
```

Uso no QuantNote:

- Explicar os clusters gerados pelo K-Means
- Gerar regras interpretáveis para cada regime
- Validar se os clusters fazem sentido econômico

5.3 Decisão Final: Slope + MA Distance + Volatility

Após testes extensivos, concluímos que:

1. **Slope** é sempre a feature mais importante (direção da tendência)
2. **MA Distance** captura a força da tendência de forma robusta
3. **Volatility** diferencia regimes de alta/baixa volatilidade

Features descartadas:

- **RSI**: Redundante com slope para nosso propósito
- **MACD**: Alta correlação com MA Distance
- **Rolling Return**: Capturado indiretamente pelo slope

---

6. Por que Rodar o GA para Cada Alvo e Horizonte?

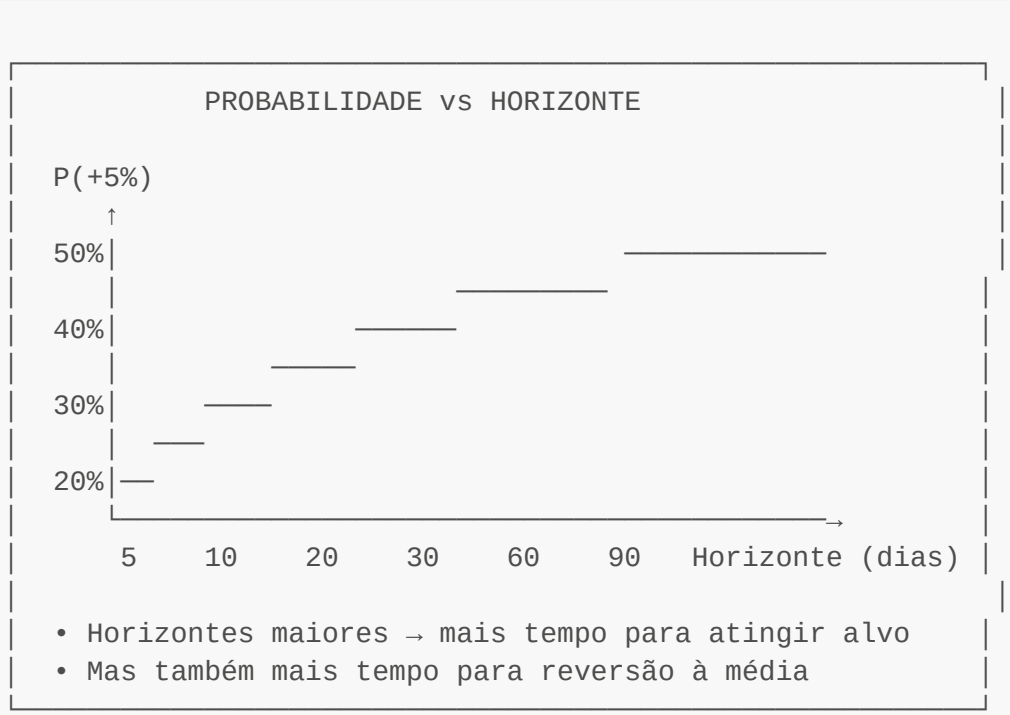
6.1 Sensibilidade ao Alvo

Diferentes alvos requerem diferentes parâmetros ótimos:

Alvo	Slope Ótimo	Volatility Ótima	Intuição
+2%	Janela curta (7-15)	Janela média (15-25)	Movimentos pequenos, reagem rápido
+5%	Janela média (15-30)	Janela média (20-30)	Movimento moderado
+10%	Janela longa (30-50)	Janela longa (30-45)	Tendências estruturais

6.2 Sensibilidade ao Horizonte

O horizonte H afeta fundamentalmente as probabilidades:



6.3 O Viés da Reversão à Média

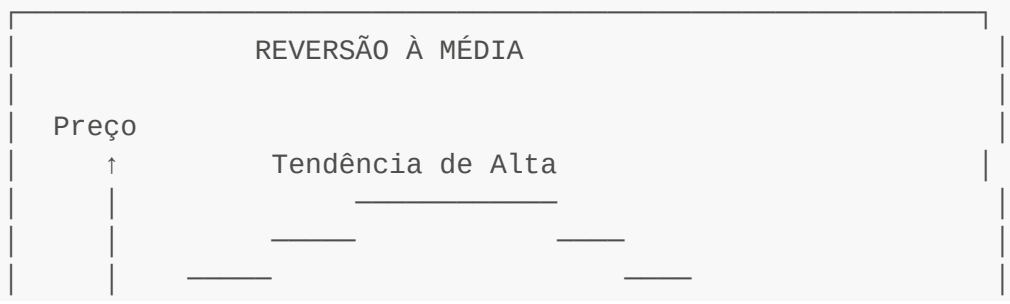
Em tendências fortes, horizontes longos apresentam um fenômeno interessante:

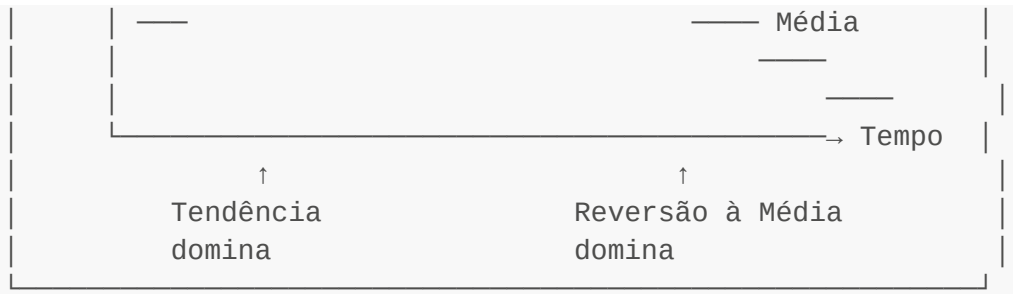
Cenário: Ativo em forte alta (slope muito positivo)

Horizonte	P_condicional(+5%)	P_incondicional(+5%)	Diferença
7 dias	45%	30%	+15%
30 dias	38%	35%	+3%
90 dias	32%	33%	-1%

Explicação:

- Em horizontes curtos, a tendência atual domina
- Em horizontes longos, a **reversão à média** dilui o efeito da tendência
- A probabilidade condicional pode até ser MENOR que a incondicional para horizontes muito longos

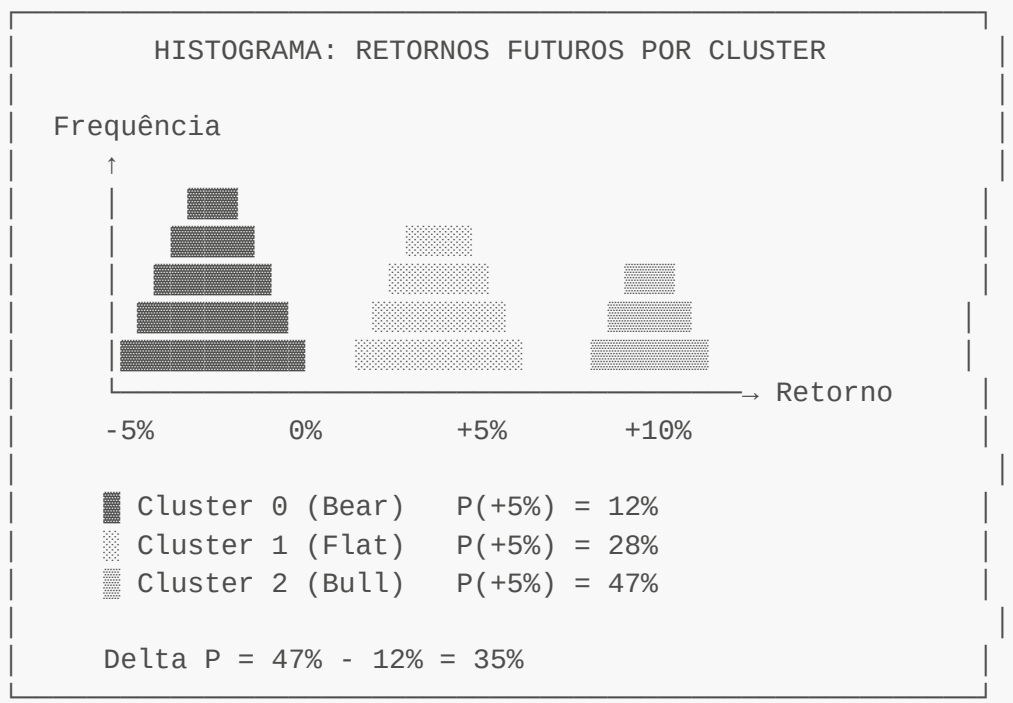




---

7. Resultados e Visualizações

7.1 Distribuição de Retornos por Regime



7.2 Matriz de Probabilidades por Strike

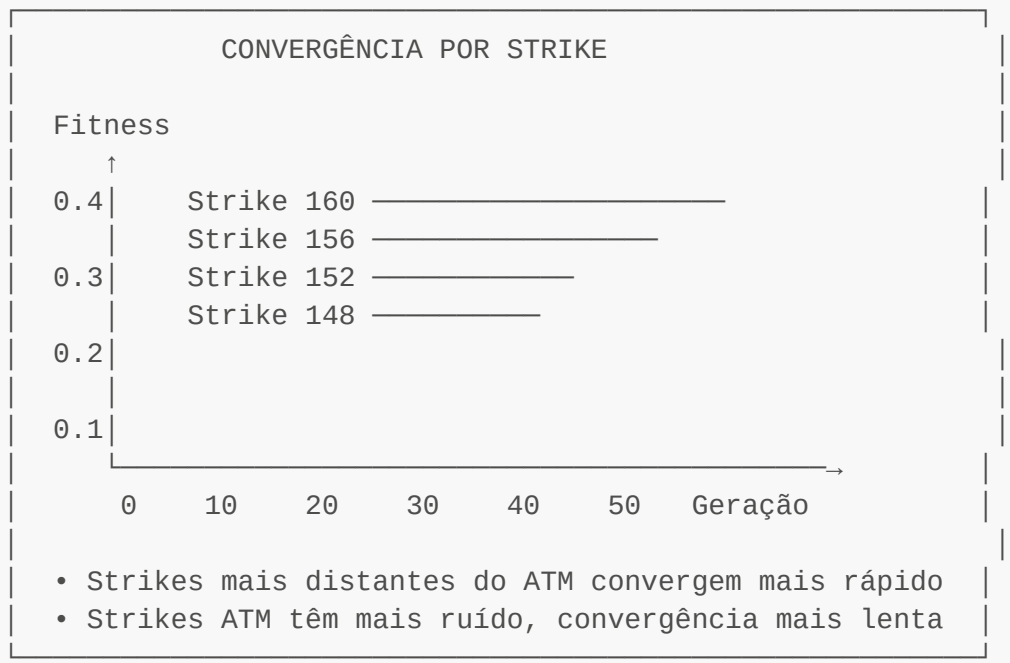
```
================================================================================
=
COMPARAÇÃO: P(fechar) vs P(tocar) - BOVA11.SA
Preço atual: R$ 155.50 | Horizonte: 7 dias
================================================================================
=

  Strike  Target Tipo | P(fech)|R  P(fech) |  P(toc)|R   P(toc) |  Δ(cond)
Δ(base)
--------------------------------------------------------------------------------
-----------
R$   148   -4.8%  PUT |     88.2%    82.3% |     96.5%    94.1% |    +8.3%
+11.8%
```

```
R$   150   -3.5%  PUT │     78.4%    71.5% │     92.1%    88.7% │   +13.7%
+17.2%
R$   152   -2.3%  PUT │     65.3%    58.2% │     84.6%    79.3% │   +19.3%
+21.1%
R$   154   -1.0%  PUT │     52.1%    44.8% │     74.2%    67.4% │   +22.1%
+22.6%
R$   156   +0.3%  ATM │     42.5%    35.1% │     61.2%    52.8% │   +18.7%
+17.7% ← ATM
R$   158   +1.6% CALL │     31.8%    25.3% │     48.5%    41.2% │   +16.7%
+15.9%
R$   160   +2.9% CALL │     22.4%    17.2% │     37.8%    31.5% │   +15.4%
+14.3%
R$   162   +4.2% CALL │     15.1%    11.8% │     28.9%    23.1% │   +13.8%
+11.3%
--------------------------------------------------------------------------------
-----------

Legenda:
  P(fech)|R  = P(fechar | regime) - Probabilidade CONDICIONAL
  P(fech)    = P(fechar) - Probabilidade INCONDICIONAL (base)
  P(toc)|R   = P(tocar | regime) - Probabilidade CONDICIONAL
  P(toc)     = P(tocar) - Probabilidade INCONDICIONAL (base)
  Δ(cond)    = P(toc)|R - P(fech)|R
  Δ(base)    = P(toc) - P(fech) → deve ser sempre POSITIVO
```

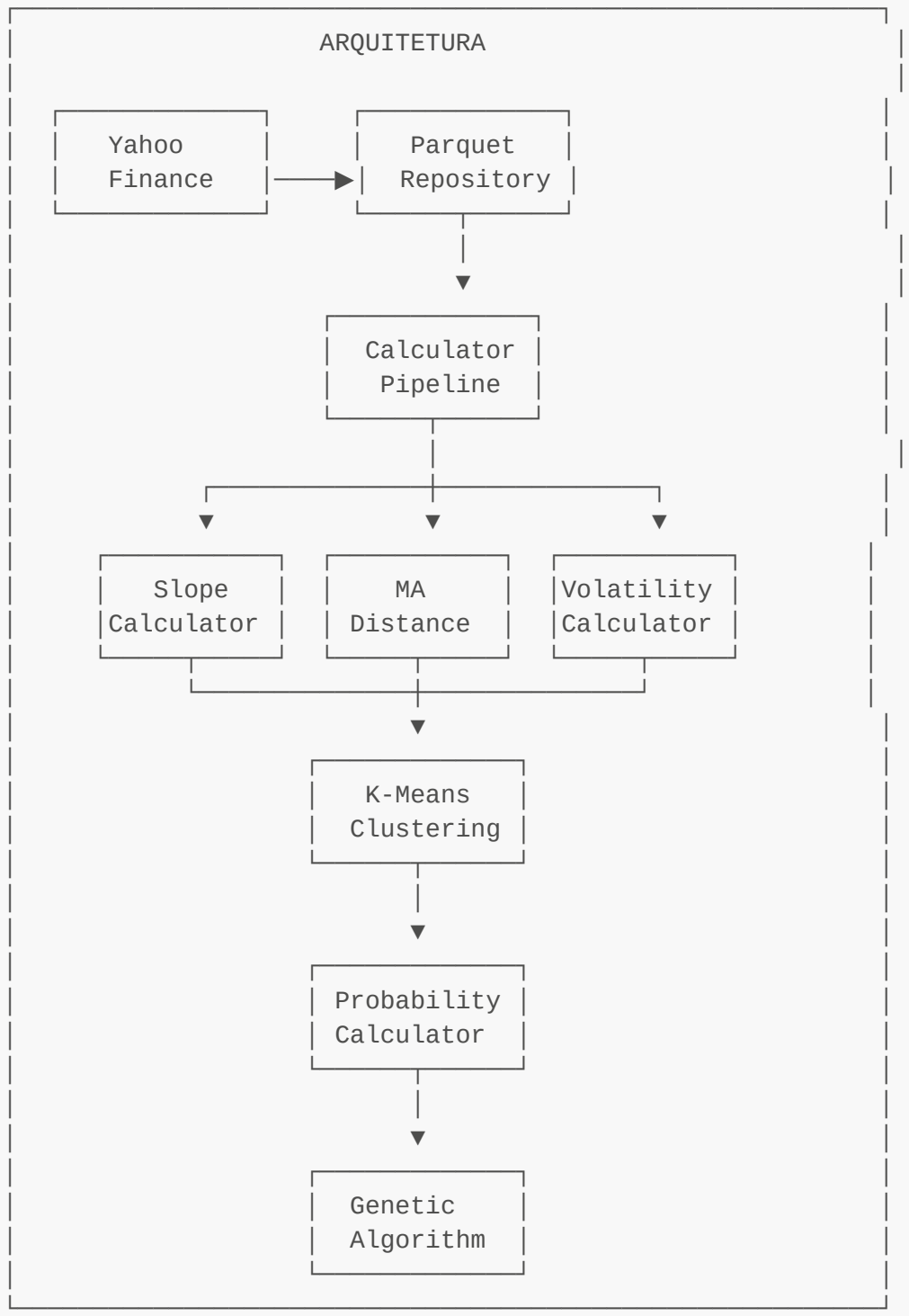
7.3 Convergência do GA por Strike



---

8. Implementação Técnica

8.1 Arquitetura do Sistema



8.2 Strategy Pattern para P(fechar) vs P(tocar)

```python
class IReturnStrategy(Protocol):
    """Interface para estratégias de retorno."""

    @property
    def name(self) -> str: ...

    def should_include_touch_calculator(self) -> bool: ...
```

```
    def get_return_column(self, horizon: int, target_return: float) -> str:
...


class CloseReturnStrategy:
    """P(fechar) - usa retorno close-to-close."""

    def get_return_column(self, horizon: int, target_return: float) -> str:
        return f'log_return_future_{horizon}'


class TouchReturnStrategy:
    """P(tocar) - usa max(high) ou min(low) conforme direção."""

    def get_return_column(self, horizon: int, target_return: float) -> str:
        if target_return >= 0:
            return f'log_return_touch_max_{horizon}'
        else:
            return f'log_return_touch_min_{horizon}'
```

---

9. Conclusões e Trabalhos Futuros

9.1 Conclusões

1. **Probabilidades condicionais superam incondicionais**: Delta P médio de 15-25% entre regimes demonstra o valor da segmentação.
2. **K-Means é eficaz para identificação de regimes**: Clusters interpretáveis e economicamente sensíveis.
3. **GA otimiza eficientemente o espaço de parâmetros**: Convergência típica em 40-60 gerações.
4. **P(tocar) > P(fechar)** é uma relação matemática validada empiricamente.
5. **Horizonte e alvo requerem otimização independente**: Parâmetros ótimos variam significativamente.

9.2 Trabalhos Futuros

- [ ] Incorporar features de sentimento de mercado
- [ ] Testar outros algoritmos de clustering (DBSCAN, GMM)
- [ ] Implementar ensemble de modelos por regime
- [ ] Adicionar backtesting integrado para validação em produção
- [ ] Explorar Deep Learning para feature extraction automática

---

Referências

1. MacQueen, J. (1967). "Some methods for classification and analysis of multivariate observations". Berkeley Symposium on Mathematical Statistics and Probability.
2. Holland, J.H. (1975). "Adaptation in Natural and Artificial Systems". University of Michigan Press.

3. Breiman, L. (2001). "Random Forests". Machine Learning, 45(1), 5-32.

4. Bailey, D.H., et al. (2014). "The Probability of Backtest Overfitting". Journal of Computational Finance.

Documento gerado pelo projeto QuantNote *Versão 1.0 - Dezembro 2024*