基本面综合评分系统算法详解

本文档旨在详细阐述如何一步步构建股票的“基本面综合评分”，确保与实际执行的 Python 代码逻辑一致。

1. 数据收集与处理

1.1. 数据源与周期

- 数据来源：通过 yfinance 库获取美股市场的财务数据和行情数据。S&P 500 公司列表及行业分类信息从维基百科页面 "List of S&P 500 companies" 抓取。
- 覆盖公司：标普500指数成分股。
- 财务报告类型：
  - 年报 **(Annual, 'A')**：获取公司历年年报数据，用于计算长期增长趋势。代码中会选取最近 FY_YEARS（可配置，默认为2年）的年报进行分析。
  - 季报 **(Quarterly, 'Q')**：获取公司历季季报数据，用于计算近期连续增长情况，通常会考察最近4个季度的数据。
- 主要财务数据字段 **(RAW_COLS)**：
  - 利润表：total_revenue (总营收), eps (每股收益), gross_profit (毛利润), operating_income (营业利润), net_income (净利润), research_development (研发费用), interest_expense (利息费用), ebitda (息税折旧摊销前利润)。
  - 资产负债表：total_assets (总资产), total_current_assets (流动资产合计), total_current_liabilities (流动负债合计), cash_and_eq (现金及现金等价物), minority_interest (少数股东权益), total_debt (总负债，代码中特指长期负债与短期长期负债之和), shares_outstanding (流通股数), total_liabilities (负债合计)。
  - 现金流量表：operating_cash_flow (经营活动现金流), capital_expenditures (资本支出)。
  - 衍生计算：free_cash_flow (自由现金流 = 经营活动现金流 + 资本支出，注意资本支出在原始数据中通常为负值)。
- 市场行情数据：
  - price (最近收盘价)
  - forward_eps (远期每股收益)
- 数据更新模式 **(update_mode)**：
  - incremental (增量更新)：只下载数据库中对应股票最新财报日期之后的新数据。
  - full (全量更新)：删除现有数据库文件，重新下载所有数据。
- **EPS** 填补：若原始 eps 数据缺失，则尝试使用 net_income / shares_outstanding 计算填充。

1.2. 数据预处理与存储

- 列名规范化 **(_norm, first_available)**：由于 yfinance 返回的财务报表列名可能存在细微差异 (如 "Total Revenue" vs "Revenue")，代码采用模糊匹配逻辑 (first_available

函数) 来获取正确的列数据。
- 数据库存储：
  - 原始财务数据存储在 SQLite 数据库 (sp500_finance.db) 的 raw_financials 表中。
  - 计算后的指标存储在 derived_metrics 表。
  - 最终评分存储在 scores 表。

2. 核心财务指标选取与计算

系统从增长(Growth)、质量(Quality)、效率(Efficiency)、安全(Safety)、估值(Valuation)五个维度对公司基本面进行评估。

2.1. 增长维度 **(Growth)**

增长维度结合了年报 (FY - Fiscal Year) 的同比/复合年均增长视角和季报 (Q-Seq - Sequential Quarterly) 的环比增长视角。

- 年报 **(FY)** 增长指标 (基于最近 FY_YEARS 年数据，计算方法 FY_CALC 可配置为 average 或 cagr)：
  - fy_rev_y (年营收增长率)：总营收的年均增长率或复合年增长率。
  - fy_eps_y (年EPS增长率)：每股收益的年均增长率或复合年增长率。
  - fy_fcf_y (年自由现金流增长率)：自由现金流的年均增长率或复合年增长率。
  - fy_margin_delta (年毛利率变动)：毛利率 (gross_profit / total_revenue) 的年均变动值。
  - 增长率计算规则 **(_avg_growth for 'average', _cagr for 'cagr')**:
    - 若数据点少于2个，则增长率为 NaN。
    - **Average Growth (_avg_growth)**: 计算每期之间的增长率 (current / previous) - 1。若 previous <= 0 且 current > 0 (扭亏为盈)，则该期增长率视为 100% (1.0)。排除无穷大和NaN后，取平均值。
    - **CAGR (_cagr)**: (last_value / first_value)^(1 / num_periods) - 1。若 first_value <= 0 且 last_value > 0，则CAGR视为 100% (1.0)。若 first_value <= 0 或 last_value <= 0 (除扭亏为盈情况)，则为 NaN。
- 季报 **(Q-Seq)** 序列增长指标 (基于最近4个季度数据，计算最近3个季度的连续环比增速的均值 - seq_growth 函数)：
  - qseq_rev (季营收环比增长率均值)。
  - qseq_eps (季EPS环比增长率均值)。
  - qseq_fcf (季自由现金流环比增长率均值)。
  - qseq_margin_delta (季毛利率环比变动均值)。
  - 序列增长率计算规则 **(seq_growth)**:
    - 取最近4个季度的数据，计算相邻季度间的环比增长率 (current / previous) - 1，共3组。

        - 若 previous <= 0 且 current > 0 (扭亏为盈)，则该期环比增长率视为 100% (1.0)。
        - 排除无穷大和NaN后，取这3个有效环比增长率的平均值。若有效值不足，则为 NaN。

2.2. 盈利质量维度 **(Quality)**

基于最新一期报告期（优先季报，若无则年报）的数据计算：

- roic (投入资本回报率)：
    - NOPAT (税后营业利润) = operating_income * (1 - 0.21) (税率近似为21%，代码中为0.79，即1-0.21)。
    - Invested Capital (投入资本) = total_debt + minority_interest + (total_assets - total_liabilities) - cash_and_eq。
    - ROIC = NOPAT / Invested Capital。若 Invested Capital <= 0，则为 NaN。
- roe (净资产收益率)：
    - Equity (净资产) = total_assets - total_liabilities。
    - ROE = net_income / Equity。若 Equity <= 0，则为 NaN。

2.3. 运营效率维度 **(Efficiency)**

基于最新一期报告期数据计算：

- ocf_ratio (经营现金流率)：operating_cash_flow / total_revenue。若 total_revenue 为0或NaN，则为 NaN。
- asset_turnover (资产周转率)：total_revenue / total_assets。若 total_assets 为0或NaN，则为 NaN。

2.4. 财务安全维度 **(Safety)**

基于最新一期报告期数据计算：

- net_debt_ebitda (净负债/EBITDA)：
    - Net Debt (净负债) = total_debt - cash_and_eq。
    - Net Debt/EBITDA = Net Debt / ebitda。若 ebitda 为0或NaN，则为 NaN。
- interest_coverage (利息覆盖倍数)：operating_income / abs(interest_expense)。若 interest_expense 为0或NaN，则为 NaN。
- current_ratio (流动比率)：total_current_assets / total_current_liabilities。若 total_current_liabilities 为0或NaN，则为 NaN。

2.5. 估值维度 **(Valuation)**

基于最新市场行情和最新一期报告期数据计算：

- peg (市盈率相对盈利增长比率)：(price / forward_eps) / qseq_eps。要求 price,

forward_eps, qseq_eps 均有效且 qseq_eps > 0。
- fcf_yield (自由现金流收益率)：free_cash_flow / (price * shares_outstanding) (即 公司自由现金流 / 公司市值)。要求 price 和 shares_outstanding 有效。

3. 指标量化与评分

将原始财务指标转化为0-100分的标准化评分。

3.1. 数据预处理

- 缺失值填充：对于计算出的各原始指标 (fy_rev_y, roic 等)，如果存在缺失值 (NaN)，则用该指标在所属行业 **(industry)** 内的中位数进行填充。如果整个行业都缺失该值，则填充为0。
- 极端值处理 **(Winsorization)**：对填充后的每个指标，进行温和截尾处理。将超出特定分位数范围的值拉回边界。
  - 下界：winsor_min (可配置，默认为0.05，即5%分位数)。
  - 上界：winsor_max (可配置，默认为0.95，即95%分位数)。
  - 公式：s.clip(s.quantile(WINSOR_MIN), s.quantile(WINSOR_MAX))

3.2. 分位排名与初步评分

- 排名范围 **(percentile_scope)**：
  - industry (默认)：在公司所属的 GICS Sub-Industry 内进行百分位排名。
  - all：在所有 S&P 500 公司中进行百分位排名。
  - 小行业回退机制：如果选择按行业排名，但该行业的公司数量小于 min_industry_size (可配置，默认为5家)，则自动回退到全市场排名，以保证分数的稳定性。
- 计算百分位得分：
  - 对于"越高越好"的指标 (如 ROIC, 营收增长率)：percentile_rank * 100。
  - 对于"越低越好"的指标 (如 Net Debt/EBITDA, PEG)：(1 - percentile_rank) * 100。
  - HIGH_IS_GOOD 字典定义了每个指标的优劣方向。

3.3. 维度评分

- 增长维度评分 **(growth_score)**：这是一个两步加权过程：
  1. 子项融合：首先，对增长维度下的四个子项（营收、EPS、自由现金流、毛利率）分别计算其融合得分。每个子项的融合得分是其年报(FY)得分和季报(Q-Seq)得分的加权平均：
     - c_rev_score = (FY_W * fy_rev_y_score + QSEQ_W * qseq_rev_score) / (FY_W + QSEQ_W)
     - c_eps_score = (FY_W * fy_eps_y_score + QSEQ_W * qseq_eps_score) / (FY_W + QSEQ_W)
     - c_fcf_score = (FY_W * fy_fcf_y_score + QSEQ_W * qseq_fcf_score) / (FY_W + QSEQ_W)

        - c_margin_score = (FY_W * fy_margin_delta_score + QSEQ_W * qseq_margin_delta_score) / (FY_W + QSEQ_W)
        - 其中 FY_W 和 QSEQ_W 是年报视角和季报视角的权重，来自配置文件 combo_weights (默认 FY: 0.4, Q-Seq: 0.6)。
    2. 整体增长分：然后，将这四个融合后的子项得分再次进行加权平均，得到最终的 growth_score：
        - growth_score = G_W["rev"]*c_rev_score + G_W["eps"]*c_eps_score + G_W["fcf"]*c_fcf_score + G_W["margin"]*c_margin_score
        - 其中 G_W 是各子项在增长维度内部的权重 (默认 rev:0.5, eps:0.3, fcf:0.1, margin:0.1)。
- 其他维度评分 **(Quality, Efficiency, Safety, Valuation)**：
    - quality_score: roic_score 和 roe_score 的简单平均值。
    - efficiency_score: ocf_ratio_score 和 asset_turnover_score 的简单平均值。
    - safety_score: net_debt_ebitda_score, interest_coverage_score, 和 current_ratio_score 的简单平均值。
    - valuation_score: peg_score 和 fcf_yield_score 的简单平均值。

3.4. 综合总评分 **(total_score)**

将五个维度的得分按照预设权重 (weights 配置节) 进行加权平均，得到最终的 total_score：
total_score = W_growth * growth_score + W_quality * quality_score + W_efficiency * efficiency_score + W_safety * safety_score + W_valuation * valuation_score
(默认权重: Growth:0.45, Quality:0.20, Efficiency:0.10, Safety:0.15, Valuation:0.10。注意：配置文件 config_finance.ini 中的权重可能与此默认值不同，代码会优先使用配置文件中的值。)
4. 星级评定

根据 total_score 将公司划分为不同的星级 (rating_thresholds 配置节)：

- ★★★★★ (五星): total_score >= five_star (默认 85)
- ★★★★ (四星): total_score >= four_star (默认 70)
- ★★★ (三星): total_score >= three_star (默认 55)
- ★★ (二星): total_score >= two_star (默认 40)
- ★ (一星): total_score < two_star

(注意：星级评定逻辑在提供的 *Python* 代码中未直接实现输出，但配置文件中有相关阈值定义，暗示了其设计意图。)

5. 输出结果

- **Excel** 文件：将包含原始指标、各级评分和最终总分的详细表格导出到 Excel 文件。文件名可配置，默认为 high_growth_scoring_YYYYMMDD.xlsx。表格会按 total_score 降序排列。
- 数据库：所有原始数据、计算的中间指标和最终评分都保存在 SQLite 数据库中，方便

后续审计或二次研究。

6. 可配置参数总结 (config_finance.ini)

- [data]: start_date, end_date, update_mode
- [database]: db_name
- [weights]: growth, quality, efficiency, safety, valuation (各维度总分权重)
- [metric_parameters]:
  - winsor_min, winsor_max (Winsorization边界)
  - percentile_scope (排名范围：industry/all)
  - fy_years (年报回溯年数)
  - fy_calc (年报增长计算方式：average/cagr)
  - min_industry_size (行业内排名所需最小公司数)
- [rating_thresholds]: five_star, four_star, three_star, two_star (星级划分阈值)
- [combo_weights]: fy, qseq (增长维度下，年报视角与季报视角的融合权重)
- [export]: excel_file_name (导出Excel文件名)

此文档详细描述了基于 Python 代码的评分系统，涵盖了从数据获取、指标计算、量化评分到结果输出的完整流程。所有关键计算逻辑和可配置参数均已说明。