使用文档（A股中线右侧交易分析系统）

本项目用于 **A股日线级别** 的选股分析（v3：低位主升浪模型）：自动拉取日线数据 → 写入数据库 → 计算指标/支撑压力 → **强制过滤**（急跌/高位）→ **评分 v3** → 导出股票池 → 回测/统计信号后表现。
免责声明：仅用于学习研究，不构成任何投资建议，请自行承担风险。

---

1. 目录结构

- `astock_analyzer.py`：命令行入口
- `a_stock_analyzer/`：核心逻辑
- `requirements.txt`：Python 依赖
- `data/`：默认 SQLite 数据库目录
- `output/`：导出 CSV、回测报告输出目录
- `config.ini`：数据库配置（MySQL/SQLite）

---

2. 环境要求与安装

- Python：建议 `3.9+`
- 需要联网（AkShare 拉取行情需要网络）

安装依赖：

```
pip install -r requirements.txt
```

---

3. 数据库选择（强烈推荐 MySQL）

3.1 SQLite（默认）

- 优点：零配置，直接用文件数据库
- 缺点：不适合并发写入；本项目会自动强制 `workers=1`

SQLite 默认路径：`data/stock.db`

3.2 MySQL（推荐，用于多线程加速）

本项目使用 SQLAlchemy + PyMySQL 连接 MySQL，并做了 **按股票粒度的多线程抓取 + 多线程写入**（每只股票一个事务），用于显著提升"几千只股票"的更新速度。

---

4. 配置文件（config.ini）

你说你有本地 MySQL：直接编辑项目根目录的 `config.ini`，把账号密码填进去即可。

有两种写法（二选一）：

4.1 直接写完整连接串（推荐）

```
[database]
db_url = mysql+pymysql://user:password@127.0.0.1:3306/astock?charset=utf8mb4
```

4.2 填写 mysql 段（程序自动拼接）

```
[mysql]
host = 127.0.0.1
port = 3306
user = your_user
password = your_password
database = astock
charset = utf8mb4
```

优先级（从高到低）：

1. 命令行 `--db-url`
2. 环境变量 `ASTOCK_DB_URL`
3. 命令行 `--db`（SQLite 文件路径）
4. `config.ini`

默认会读取项目根目录的 `config.ini`；如果你想指定其它配置文件，使用 `--config your.ini`。

---

5. 本地 MySQL 初始化（一次性）

你本地已有 MySQL 的话，按需执行（示例）：

```
CREATE DATABASE astock DEFAULT CHARACTER SET utf8mb4 COLLATE
utf8mb4_0900_ai_ci;
CREATE USER 'astock'@'localhost' IDENTIFIED BY 'your_password';
GRANT ALL PRIVILEGES ON astock.* TO 'astock'@'localhost';
FLUSH PRIVILEGES;
```

然后把 `config.ini` 填好。

初始化表结构：

```
python astock_analyzer.py init-db
```

MySQL 下会额外创建一个视图：`vw_stock_pool_v3_latest`（取 `stock_scores_v3` 最新交易日的股票池）。

> 说明：如果你不想用 `config.ini`，也可以直接传 `--db-url "...mysql+pymysql://..."`。

---

6. 更新数据 + 计算指标/评分（run）

命令：

```
python astock_analyzer.py run --start-date 20000101 --end-date 20260107 --
workers 16
```

参数说明：

- `--start-date`：首次全量时的起始日期（格式 `YYYYMMDD`）
- `--end-date`：结束日期（格式 `YYYYMMDD`），默认当天
- `--workers`：线程数（MySQL 推荐 8~32；SQLite 会自动强制 1）
- `--limit-stocks`：仅跑前 N 只股票（调试用）

全量/增量逻辑（按每只股票）：

1. 查询 `stock_daily` 里该股票的 `MAX(date)`（最后交易日）
2. 没有记录：从 `--start-date` 开始拉取
3. 有记录：从 `MAX(date)+1` 开始做增量更新
4. 写入 `stock_daily` 后，再基于历史数据计算指标/支撑压力/评分写入：
    - `stock_indicators`
    - `stock_levels`
    - `stock_scores_v3`
    - （可选）`stock_future_perf`：信号后 `ne` 日表现（由 `future-perf` 命令生成）

v3 强制过滤（否决项，先执行）：

- 急跌过滤（最近 20 个交易日内）：单日跌幅 ≥9% / 3 日累计跌幅 ≥15% / 放量下跌（量 > 20 日均量 ×1.8 且收盘 < MA20）
- 高位否决：距离最近 120 日最高价 < 15%

性能建议：

- 首次全量建议先用 `--limit-stocks 50` 试跑，确认流程与数据正常后再全市场跑。
- 如果抓取端（AkShare 数据源）出现限速/失败日志，适当降低 `--workers`。

---

7. 导出股票池（export）

命令：

```
python astock_analyzer.py export --output output/pool.csv --top 200 --min-score
80 --require-tags TREND_UP,AT_SUPPORT --min-resistance-distance 0.10
```

参数说明：

- `--output`：输出 CSV 路径
- `--top`：导出数量上限（按评分降序）
- `--min-score`：最低总评分（例如 80）
- `--require-tags`：必须包含的标签（逗号分隔）
- `--min-resistance-distance`：距离压力位至少多少比例（例如 `0.10` 表示 10%）

标签说明（`status_tags`，JSON 数组）：

- `TREND_UP` / `LOW_BASE` / `PULLBACK` / `AT_SUPPORT` / `SPACE_OK` / `NEAR_RESISTANCE` / `RISK_FILTERED`

---

8. 回测接口（backtest）

功能：随机抽取 `nd` 个交易日，在当日选出 `score>=k` 的股票，观察其后续 `ne` 个交易日的表现，输出：

- `ne` 窗口内的 **最高价/最高收益率**
- `ne` 窗口内的 **最低价/最低收益率**
- `ne` 窗口结束时的 **最终收盘价/最终收益率**

说明：

- 抽样交易日来自 `stock_daily` 的交易日序列（会自动避开末尾 `ne` 天，保证有足够的未来窗口）。
- 如果 `stock_scores` 表里刚好存在该 `score_date` 的预计算评分，则直接使用；否则会基于当日之前的历史数据 **即时计算评分**（`nd` 很大时会比较慢）。

> v3 版本默认使用 `stock_scores_v3` 作为评分表（旧表 `stock_scores` 仅为历史兼容，不再作为主流程输出）。

命令：

```
python astock_analyzer.py backtest --nd 50 --ne 20 --k 80 --seed 42 --workers
16 --out-dir output
```

参数说明：

- `--nd`：随机抽样交易日数量
- `--ne`：向后观察的交易日数量（按该股票后续 `stock_daily` 的 `ne` 条记录）
- `--k`：评分阈值（默认 80，可随时改）
- `--seed`：随机种子（可选；固定后可复现实验）
- `--workers`：回测读取线程数
- `--out-dir`：输出目录

输出文件：

- `output/backtest_YYYYMMDD_HHMMSS.csv`：明细
- `output/backtest_YYYYMMDD_HHMMSS.md`：按抽样日聚合的汇总报告

收益率计算：

- `return = price / start_close - 1`

---

9. 信号后表现（future-perf）

用途：按评分表 `stock_scores_v3` 中的"信号"（`total_score >= k` 且非 `RISK_FILTERED`），统计信号后 `ne` 个交易日的最大涨幅/最大回撤/最终收益，并写入 `stock_future_perf`。

默认 `ne=5,10,20,30`，并输出 `output/future_perf_*.csv` + `output/future_perf_*.md`：

```
python astock_analyzer.py future-perf --ne 5,10,20,30 --min-score 80 --workers
16 --out-dir output
```

可选：指定信号日期（`YYYY-MM-DD`）：

```
python astock_analyzer.py future-perf --signal-date 2026-01-05 --ne 20 --min-
score 80
```

---

10. SQLite 强制使用（可选）

如果你填了 `config.ini`（MySQL），但临时想用 SQLite 测试：

```
python astock_analyzer.py run --db data/stock.db --workers 1
```