Changes to option_chain_fetcher.py

Summary

Completely refactored `option_chain_fetcher.py` from a trading bot to a data collection tool that fetches SPX option chain data and stores it in a SQLite database.

New Features

1. CLI Parameters

The script now accepts command-line arguments:

- `--max_dte`: Maximum days to expiration (required for data fetching)
- `--min_strike`: Minimum strike price (required for data fetching)
- `--max_strike`: Maximum strike price (required for data fetching)
- `--symbol`: Underlying symbol (optional, default: SPX)
- `--db_path`: Database file path (optional, default: database/option_chain_data.db)
- `--clear_db`: Clear all data from the database and exit (optional flag)

2. Data Collection

- Fetches all option chains for expirations from 0 to max_dte days
- Includes both CALL and PUT options for all strikes in the specified range
- **SPX/SPXW Handling**: When requesting SPX or SPXW, automatically fetches BOTH:
  - SPX (monthly/quarterly expirations)
  - SPXW (weekly expirations)
- **Underlying Price Tracking**: Fetches and stores the underlying symbol price at fetch time
  - For SPX/SPXW options, always fetches SPX price (not SPXW)
  - For other symbols, fetches the actual symbol price
  - Stores bid, ask, and mid prices
- Retrieves bid/ask quotes using the existing MarketData class
- Fetches all Greeks (delta, gamma, theta, vega, rho) for each option

3. SQLite Database

- Stores all data in a SQLite database with proper indexing
- Default location: `database/option_chain_data.db`
- `fetch_timestamp` field records when each dataset was collected (ISO format)
- Unique constraint on (fetch_timestamp, symbol) prevents duplicates
- Indexed by fetch_timestamp and expiration_date for fast queries

4. Database Schema

Table: `option_chain_data`

- fetch_timestamp (TEXT) - ISO timestamp when data was fetched

- symbol (TEXT) - Option symbol
- expiration_date (TEXT) - Expiration date
- strike_price (REAL) - Strike price
- option_type (TEXT) - 'CALL' or 'PUT'
- bid_price, ask_price, mid_price (REAL) - Quote data
- delta, gamma, theta, vega, rho (REAL) - Greeks
- days_to_expiration (INTEGER) - DTE

Table: `underlying_prices`

- fetch_timestamp (TEXT) - ISO timestamp when data was fetched
- symbol (TEXT) - Underlying symbol (e.g., SPX)
- price (REAL) - Mid price of underlying
- bid_price (REAL) - Bid price of underlying
- ask_price (REAL) - Ask price of underlying

Note: For SPX/SPXW options, the underlying price is always recorded as SPX (not SPXW), as both option types reference the same underlying index.

Usage Examples

```
# Fetch 0-7 DTE options with strikes 5800-6000
python option_chain_fetcher.py --max_dte 7 --min_strike 5800 --max_strike
6000

# Clear all data from the database
python option_chain_fetcher.py --clear_db

# Clear a specific database
python option_chain_fetcher.py --clear_db --db_path my_data.db
```

Additional Files Created

1. export_to_json.py

Python script to export database to JSON files:

- One JSON file per expiration date
- Data organized by strike with CALL/PUT sub-records
- Extensive CLI filtering options:
  - `--start_date`, `--end_date`: Filter by expiration date range
  - `--min_strike`, `--max_strike`: Filter by strike range
  - `--min_delta`, `--max_delta`: Filter by absolute delta
  - `--expiration_date`: Export single expiration
  - `--fetch_timestamp`: Export specific fetch
- Clean JSON structure ideal for analysis and visualization

Example:

```
# Export Oct 6-10, strikes 5800-6000, deltas 0.25-0.45
python export_to_json.py \
    --start_date 2025-10-06 \
    --end_date 2025-10-10 \
    --min_strike 5800 \
    --max_strike 6000 \
    --min_delta 0.25 \
    --max_delta 0.45
```

2. JSON_EXPORT_GUIDE.md

Complete documentation for the JSON export script:

- JSON structure explanation
- All CLI parameters
- Usage examples for various scenarios
- Code examples in Python and JavaScript
- Data analysis examples

3. OPTION_CHAIN_USAGE.md

Comprehensive usage guide including:

- Parameter descriptions
- Multiple usage examples
- Database schema documentation
- Query examples

4. query_option_data.py

Helper script to query the database with multiple modes:

- `list`: Show all available fetch timestamps
- `summary`: Display statistics for a fetch
- `delta`: Query options by delta range
- `expiration`: Query options for a specific expiration date

Example:

```
# Show summary of latest fetch
python query_option_data.py --action summary

# Find CALLs with delta 0.25-0.35
python query_option_data.py --action delta --option_type CALL --min_delta
0.25 --max_delta 0.35
```

Technical Details

Logging

- Logs to `option_chain_fetcher.log` and console
- INFO level for normal operations
- WARNING for missing data
- ERROR with stack traces for failures

Error Handling

- Gracefully handles missing quotes or Greeks
- Continues fetching even if some symbols fail
- Logs warnings for incomplete data

Performance

- Fetches quotes and Greeks separately with retry logic
- Maximum 50 attempts per data type with 5-second intervals
- Stores data in single transaction for consistency

SPX Symbol Handling

- Automatically detects SPX/SPXW symbols (case-insensitive)
- Fetches both symbol types in a single run
- Error handling for each symbol independently (if one fails, the other continues)
- Combined results in single database with all expirations

Folder Structure

The project now organizes data files in dedicated directories:

```
spx_option_chain_fetcher/
├── database/              # SQLite database files (gitignored)
│   └── option_chain_data.db
├── json_exports/          # Exported JSON files (gitignored)
│   └── spx_options_*.json
├── *.log                  # Log files (gitignored)
└── ...
```

Note: Both `database/` and `json_exports/` directories are automatically created when needed and are excluded from git.

Dependencies

All required dependencies are already in `requirements.txt`:

- tastytrade
- python-dotenv
- sqlite3 (built-in to Python)

Backward Compatibility

This is a complete refactor. The old trading bot functionality has been removed. If you need the old functionality, retrieve it from git history.