Artificial Intelligence in Quantitative Finance - Project Report

Short abstract:
An end-to-end research and engineering pipeline that implements LSTM and Transformer deep-learning models to forecast asset returns and drive dynamic portfolio allocation via mean-variance optimization. The repository covers data collection, feature engineering, model training, rolling predictions, portfolio construction, backtesting, statistical evaluation, and reproducible artifacts.

1. Goals & Scope

- Build and evaluate an AI-driven portfolio strategy using LSTM & Transformer to forecast next-day returns.
- Convert forecasts to daily portfolio weights with mean-variance optimization (MVO).
- Backtest and evaluate against benchmarks using financial and statistical metrics.
- Provide reproducible code, notebooks, and visual artifacts.

2. High-level Pipeline

1. Fetch data - src/data_fetch.py
2. Preprocess / Feature engineering - src/preprocessing.py
3. Model training - src/train.py
4. Prediction - src/predict.py
5. Optimization & Backtest - src/optimizer.py and src/backtest.py
6. Evaluation & Figures - src/metrics.py and src/evaluate_and_plot.py

3. Models & Key Hyperparameters

LSTM: 2 layers, 50 hidden units, 60-day lookback, dropout 0.2.
Transformer: 2 encoder blocks, 8 heads, dim=64, FFN=128, dropout 0.1.

4. Portfolio Construction & Backtest

- Forecast returns -> Expected returns
- Rolling covariance estimation

- Max-Sharpe allocation (no shorting)
- Daily rebalancing, transaction cost assumption
- Backtest applied to realized returns

5. Evaluation & Statistical Testing

Metrics: Annualized return, volatility, Sharpe, Max drawdown, Calmar, Sortino, turnover, RMSE, directional accuracy.

Statistical tests: Paired t-test, Jobson & Korkie (Memmel correction).

6. How to Reproduce

1. Create environment: conda env create -f environment.yml
2. Fetch raw data: python src/data_fetch.py --start 2010-01-01 --end 2020-12-31
3. Preprocess: python src/preprocessing.py
4. Train model: python src/train.py --model lstm --epochs 50
5. Predict: python src/predict.py --model-type lstm
6. Evaluate: python src/evaluate_and_plot.py --model-type lstm

7. Expected Outputs

- results/checkpoints/<model>/<timestamp>/
- results/predictions_<model>.csv
- results/metrics_<model>.csv
- results/figures/*.png

8. Limitations & Next Steps

- Data quality issues, risk of overfitting, transaction costs, regime shifts, interpretability limits.
- Future: ensemble models, RL baselines, explainability, rolling retraining.

Closing remarks:

This project is designed for reproducible research and experimentation on AI-augmented portfolio management.