Machine Learning Trading Checklist

Signal Pilot Education Hub

🤖 Machine Learning Trading Checklist

Lesson 35: Machine Learning in Trading

This checklist guides you through using ML models (classification, regression, reinforcement learning) to generate trading signals and improve strategy performance.

📋 Phase 1: Problem Definition & Data Prep

Define ML Problem Type

- [ ] **Classification** - Predict direction (up/down/neutral) → Entry signals
- [ ] **Regression** - Predict price/return magnitude → Position sizing
- [ ] **Reinforcement Learning** - Optimize strategy parameters → Adaptive trading

- [ ] **Time series forecasting** - Predict next N bars → Trend prediction

Feature Engineering (Input Variables)

- [ ] **Price-based features** - Returns (1-day, 5-day, 20-day), volatility, ATR
- [ ] **Technical indicators** - RSI, MACD, Bollinger %B, ADX, moving averages
- [ ] **Volume features** - Volume change, volume ratio to avg, volume momentum
- [ ] **Microstructure** - Bid-ask spread, order imbalance, tick direction
- [ ] **Sentiment** - VIX, put/call ratio, news sentiment scores
- [ ] **Temporal features** - Day of week, hour, time since last signal
- [ ] **Lagged features** - Previous bar returns, lagged indicators

Data Preparation

- [ ] **Collect sufficient data** - 3-5 years minimum (more is better for ML)
- [ ] **Handle missing data** - Forward fill or drop (don't backfill = lookahead bias)
- [ ] **Normalize/scale features** - StandardScaler or MinMaxScaler (models need consistent scales)
- [ ] **Create labels** - For classification: +1 (up), -1 (down), 0 (neutral)
- [ ] **Split data** - Train (60%), Validation (20%), Test (20%) - chronological split!

Avoid Lookahead Bias (Critical!)

- [ ] **No future data in features** - Only use data available at prediction time

- [ ] **Check for data leakage** - Price return as feature + label = cheating
- [ ] **Walk-forward splits** - Train on past, test on future (never reverse)
- [ ] **No shuffling time series** - Maintain temporal order always

---

🎯 Phase 2: Model Selection & Training

Choose Model Type

- [ ] **Logistic Regression** - Simple baseline (direction prediction)
- [ ] **Random Forest** - Good for feature importance, non-linear relationships
- [ ] **XGBoost/LightGBM** - State-of-art for tabular data (most popular in trading)
- [ ] **Neural Networks (LSTM/GRU)** - For sequential/time-series patterns
- [ ] **Reinforcement Learning (DQN, PPO)** - For adaptive strategy optimization

Model Training

- [ ] **Train on training set** - Fit model on 60% earliest data
- [ ] **Tune hyperparameters on validation set** - GridSearch or RandomSearch
- [ ] **Avoid overfitting** - Use regularization (L1/L2), early stopping, dropout
- [ ] **Check feature importance** - Which features matter most? Remove noise.

Model Evaluation (On Test Set Only!)

- [ ] **Classification metrics:**
- Accuracy (> 52% for direction = edge)
- Precision/Recall (balance false positives vs. false negatives)
- F1 Score (harmonic mean of precision/recall)
- AUC-ROC (> 0.55 indicates predictive power)
- [ ] **Regression metrics:**
- MAE (Mean Absolute Error)
- RMSE (Root Mean Squared Error)
- R^2 score (how much variance explained)
- [ ] **Trading-specific metrics:**
- Sharpe ratio of strategy using signals
- Max drawdown
- Win rate and average R-multiple

📊 Phase 3: Backtesting ML Strategy

Integrate ML Predictions into Strategy

- [ ] **Define signal threshold** - Probability > 0.6 = long, < 0.4 = short (adjust for precision)
- [ ] **Combine with rule-based filters** - ML + ADX > 25 (trending regime filter)
- [ ] **Position sizing based on confidence** - Higher probability = larger size (Kelly-based)
- [ ] **Risk management overlays** - ML signal + 2% max risk + stop loss rules

Backtest with Realism

- [ ] **Walk-forward testing** - Retrain model every 3-6 months (markets evolve)
- [ ] **Include slippage & commissions** - 0.05-0.1% per trade
- [ ] **Transaction costs for retraining** - Data costs, compute costs
- [ ] **Check regime dependence** - Does model work in all regimes or only trending?

Compare to Baseline

- [ ] **ML strategy vs. simple rule-based** - Is ML adding alpha or just complexity?
- [ ] **Sharpe ratio improvement** - ML should improve by 0.2-0.5+ (meaningful edge)
- [ ] **Drawdown comparison** - ML should reduce drawdown, not increase it
- [ ] **Trade frequency** - Fewer, higher-quality trades = better

---

📊 Phase 4: Live Deployment & Monitoring

Pre-Deployment Checks

- [ ] **Paper trade for 3+ months** - Validate live data pipeline works
- [ ] **Monitor prediction drift** - Are live predictions similar to backtest?
- [ ] **Check data consistency** - Live data matches training data format
- [ ] **Set up retraining pipeline** - Automate model retraining (monthly/quarterly)

Model Monitoring (Critical!)

- [ ] **Track prediction accuracy over time** - Is accuracy degrading? (model decay)
- [ ] **Monitor Sharpe ratio rolling 30-day** - Still > 1.5? Or dropping?
- [ ] **Check feature drift** - Are input distributions shifting? (regime change)
- [ ] **Set performance alerts** - If accuracy < 50% for 20 trades, pause system

Retraining Schedule

- [ ] **Retrain monthly** - Incorporate latest market data
- [ ] **Validate on out-of-sample test** - Don't deploy if test performance drops
- [ ] **Version control models** - Keep old models in case new one underperforms
- [ ] **A/B testing** - Run old vs. new model side-by-side (paper trade)

Kill-Switch Rules

- [ ] **Stop trading if:**
- Model accuracy < 48% over 30 trades (worse than random)
- Sharpe ratio < 0.5 for 1 month
- Drawdown > 20%
- Feature distributions shift dramatically (KS test $p < 0.05$)

💡 Pro Tips

ML Trading Mastery

- **Start simple, not complex** - Logistic Regression or Random Forest before deep learning
- **Feature engineering > model complexity** - Good features with simple model beats complex model with bad features
- **Retrain regularly** - Markets evolve, models decay (3-6 month retraining)
- **Combine ML with rules** - Hybrid approach (ML + regime filters) outperforms ML alone

Common Mistakes to Avoid

- ❌ Data leakage (using future information in features)
- ❌ Overfitting (99% train accuracy, 45% test accuracy)
- ❌ Not retraining (model becomes stale after 6-12 months)
- ❌ Using ML without risk management (model is signal, not strategy)
- ❌ Ignoring transaction costs (ML generates lots of trades = death by commissions)

Feature Engineering Tips

- **Lagged returns work well** - 1-day, 5-day, 20-day returns
- **Volatility features** - ATR, Bollinger width, rolling std dev
- **Trend features** - MA slopes, ADX, directional indicators
- **Volume features** - Volume change, volume-weighted returns
- **Avoid collinear features** - Don't include 10 moving averages (redundant)

Model Selection by Problem Type

- **Direction prediction (up/down):** XGBoost, Random Forest, Logistic Regression
- **Return magnitude:** Linear Regression, XGBoost Regressor
- **Time series:** LSTM, GRU (if sequential patterns important)
- **Strategy optimization:** Reinforcement Learning (DQN, PPO)

Realistic Expectations

- **Accuracy 52-58%** = Good (edge present)
- **Accuracy 58-65%** = Excellent (strong edge)
- **Accuracy > 65%** = Suspicious (check for overfitting/leakage)
- **Sharpe improvement 0.2-0.5** = Meaningful
- **Sharpe improvement > 1.0** = Verify no errors

📚 Related Resources

- **Lesson 34:** System Development (framework for building ML-based systems)
- **Lesson 37:** Trading Automation APIs (deploy ML models in production)
- **Lesson 39:** Performance Attribution (analyze ML model contribution)
- **Recommended Tools:** Python (scikit-learn, XGBoost, PyTorch), QuantConnect, Alpaca

---

Version: 1.0
Last Updated: 2025-11-02
Difficulty: Advanced

Remember: ML is a tool, not magic. Garbage in = garbage out. Focus on feature engineering, avoid overfitting, retrain regularly, and always combine with solid risk management.