AI Meets Finance

Machine Learning for Crypto Markets

Day 4 of 5

Prof. Jörg Osterrieder

BSc Seminar: Digital Finance

Spring 2026

PhD Seminar Series: Digital Finance Research

Where We Are

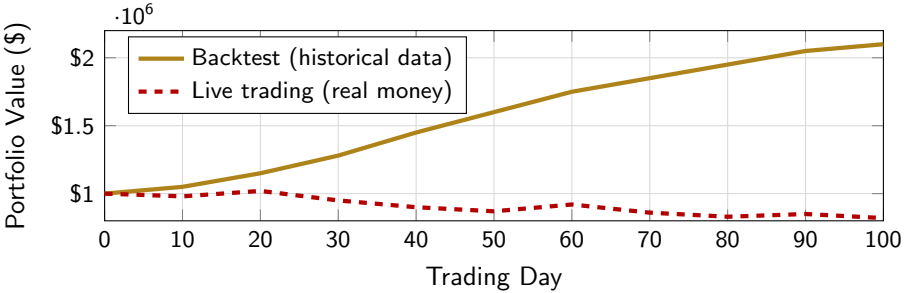
Days 1–3 Recap

- Day 1: What crypto is, exchanges, DeFi basics
- Day 2: AMMs, liquidity, Uniswap math
- Day 3: Consensus, EIP-1559, MEV, stablecoins, CBDCs

Today's Roadmap

1. What machine learning actually is
2. Overfitting: the #1 trap in finance
3. Features that predict price moves
4. Decision trees and random forests
5. Hands-on: predict BTC returns

The Bot That Made $1M in a Day...



Same algorithm. Perfect in backtest. Loses money when deployed live.

. . . And Lost It All the Next

What went wrong?

The algorithm memorized the past instead of learning patterns

- It learned: "When BTC dips 3% on a Tuesday after a Monday rally, buy"
- That pattern existed in 2021–2023 data *by coincidence*
- In 2024, the market behaved differently
- The bot kept applying the old "rules" and lost money

This is called **overfitting** — and it's the single most important concept in applied ML.

What Can ML Do (and What Can't It Do)?

ML CAN

- Find subtle patterns in large datasets
- Classify market regimes (volatile vs. calm)
- Estimate risk and volatility
- Detect anomalies and fraud
- Process news and sentiment at scale

ML CANNOT

- Predict the future with certainty
- "Solve" financial markets
- Replace human judgment on risk
- Work reliably without constant tuning
- Guarantee profits

Today: the basics of ML through crypto examples, with honest emphasis on what goes wrong.

Today: Learning ML by Doing

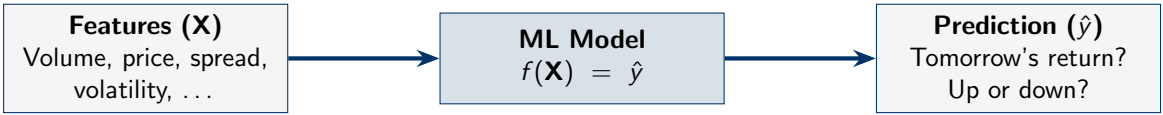
No prior ML experience needed. If you know what a function is and can read a scatter plot, you're ready.

Outline

Supervised Learning: Inputs $\rightarrow$ Outputs

The Basic Idea

Given historical data with known outcomes, learn a function that maps **inputs (features)** to **outputs (labels)**.



The model *learns* the mapping f from data, not from human rules.

Two Types of Predictions

Classification

Predict a category:

- Spam or not spam?
- Will BTC go UP or DOWN tomorrow?
- Is this wallet fraudulent?

Output: a **label** (discrete)

Regression

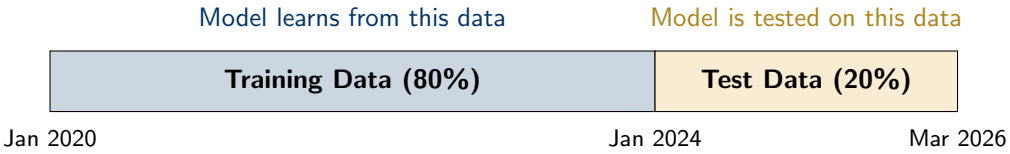
Predict a number:

- What will BTC's return be?
- What's tomorrow's volatility?
- How much gas will this block use?

Output: a **value** (continuous)

Today's exercise: classification — predict whether BTC's next-day return is positive or negative.

The Golden Rule: Train vs. Test Data



Critical Rule for Finance

Always split by time, not randomly. The model must never see future data during training. Random splits leak future information and create unrealistic results.

Overfitting: THE Most Important Concept

Definition

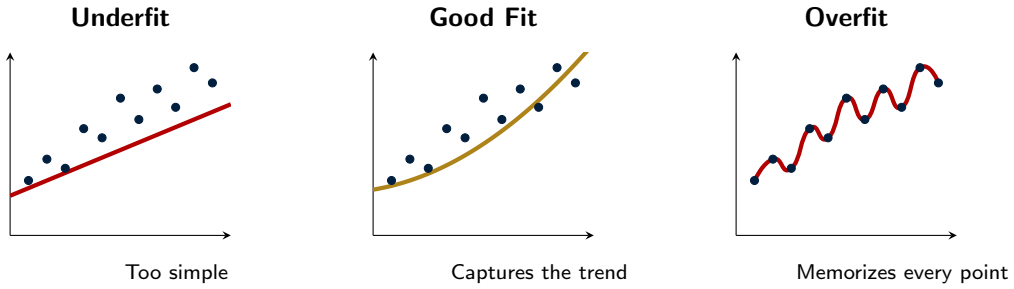
Overfitting occurs when a model learns the noise and random patterns in the training data instead of the true underlying signal.

Symptoms:

- Amazing accuracy on training data (95%+)
- Terrible accuracy on new test data (50% = coin flip)
- The model is essentially *memorizing* rather than *learning*

Analogy: A student who memorizes every answer in a practice exam but can't solve a new problem they haven't seen before. That student didn't learn the material — they learned the answers.

Visual: Underfitting vs. Good Fit vs. Overfitting



The overfit model hits every training point perfectly — but it will fail badly on new data because it learned noise, not signal.

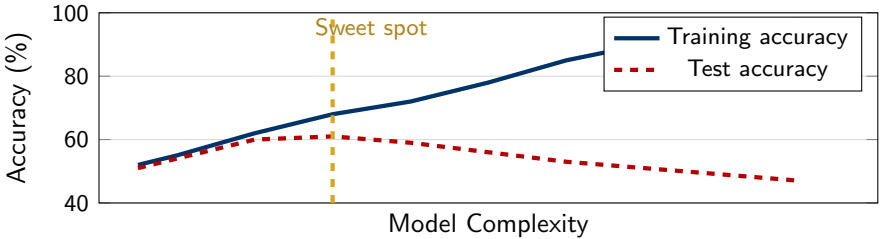
Overfitting in Action: Training Accuracy Is Deceiving

Model Complexity	Training Accuracy	Test Accuracy
Very simple (1 rule)	55%	54%
Simple (5 rules)	62%	60%
Moderate (20 rules)	71%	58%
Complex (100 rules)	88%	52%
Very complex (500 rules)	**98%**	**49%**

The Trap

The 500-rule model looks amazing on training data (98% accurate!) but performs *worse than a coin flip* on new data. More complexity $\neq$ better predictions.

Markets Change: Why Overfitting Is Deadly in Finance



After a point, more complexity *hurts* predictions. The "sweet spot" balances learning real patterns vs. memorizing noise.

Why Overfitting Is ESPECIALLY Dangerous in Finance

1. **Low signal-to-noise ratio**
   Financial data is very noisy. True patterns are subtle and fleeting.
2. **Non-stationarity**
   Markets change over time. Patterns that worked in 2021 may not work in 2025.
3. **Survivorship bias**
   We only see strategies that "worked" in backtests. Failed ones are discarded.
4. **Many researchers, few patterns**
   Thousands of quants test millions of hypotheses. Some will look significant by pure chance.
5. **Adversarial environment**
   Once a pattern is discovered and traded, other participants adapt and the pattern may disappear.

Outline

What Information Predicts Crypto Price Moves?

Features = the input variables we feed to our ML model.

Naive Approach

Just use past prices:

- Yesterday's return
- 5-day moving average
- 30-day trend

Problem: past prices alone are weak predictors (markets are *nearly* efficient [3]).

Better Approach

Use **market microstructure** features:

- Trading volume
- Bid-ask spread
- Liquidity measures
- Volatility
- On-chain data

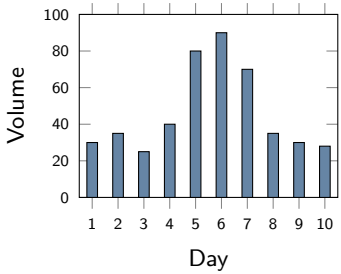
These capture HOW the market is trading.

Feature 1: Trading Volume

What it tells us:

- **High volume + price up** = strong bullish signal (many buyers agree)
- **High volume + price down** = strong bearish signal (many sellers agree)
- **Low volume + price move** = weak signal (few participants, unreliable)

Think of it as "conviction": Volume tells you how many people believe in a price move.

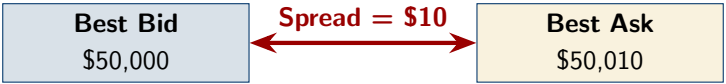


Volume spike on days 5–7 signals something important happened.

Feature 2: Bid-Ask Spread

Definition

Spread = Best ask price − Best bid price



What it tells us:

- **Tight spread** (small): liquid market, easy to trade, low cost
- **Wide spread** (large): illiquid, risky, expensive to trade
- Spreads **widen before crashes** — early warning signal!
- Crypto spreads are much wider than stock spreads

Feature 3: Amihud Illiquidity Ratio

Formula

$$\text{Amihud}_t = \frac{|r_t|}{\text{Volume}_t}$$

where r_t is the return on day t and Volume is in dollars.

Intuition: How much does the price move per dollar of trading?

- **High Amihud** = small trades cause big price swings (illiquid, risky)
- **Low Amihud** = even large trades barely move the price (liquid, safe)

Example

- BTC: $|r_t| = 2\%$, Volume = \$30B $\Rightarrow$ Amihud = 0.00067
- Small altcoin: $|r_t| = 5\%$, Volume = \$1M $\Rightarrow$ Amihud = 50

The altcoin is $\sim$75,000$\times$ more illiquid per dollar traded.

Summary: Our Feature Set for ML

Feature	Formula	What It Captures
Returns (1d, 5d, 20d)	$r_t = \frac{P_t - P_{t-1}}{P_{t-1}}$	Momentum and trend
Volatility (20d)	$\sigma_t = \text{std}(r_{t-19}, \ldots, r_t)$	Risk and uncertainty
Volume ratio	$\frac{\text{Vol}_t}{\text{Vol}_{\text{20d avg}}}$	Unusual activity
Bid-ask spread	$\text{ask}_t - \text{bid}_t$	Liquidity cost
Amihud ratio	$\frac{\lvert r_t \rvert}{\text{Volume}_t}$	Price impact per dollar

These are the **inputs (X)** to our model.
The **output (y)**: will tomorrow's return be positive (1) or negative (0)?

Outline

Decision Trees: A Flowchart for Predictions

Idea

A decision tree makes predictions by asking a series of yes/no questions about the features, splitting the data at each step.

Think of it like 20 Questions:

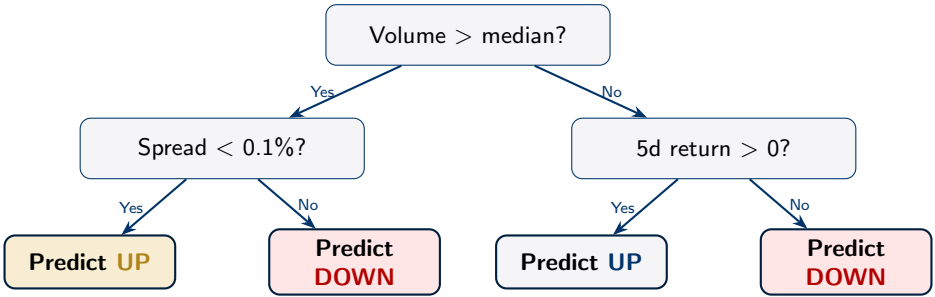
- "Is volume above average?" $\rightarrow$ Yes/No
- "Is the 5-day return positive?" $\rightarrow$ Yes/No
- "Is the spread below 0.1%?" $\rightarrow$ Yes/No
- After enough questions $\Rightarrow$ predict UP or DOWN

Advantages:

- Easy to understand and explain ("show me the rules")
- Handles non-linear patterns naturally
- No need to normalize features

Example: A Simple Crypto Decision Tree

"If volume is high AND spread is tight AND recent trend is up — predict UP."



Reading a Decision Tree

How to interpret each node:

1. **Split condition**: what question is asked?
2. **Left branch**: "Yes" (condition is true)
3. **Right branch**: "No" (condition is false)
4. **Leaf nodes**: final predictions

The algorithm learns:

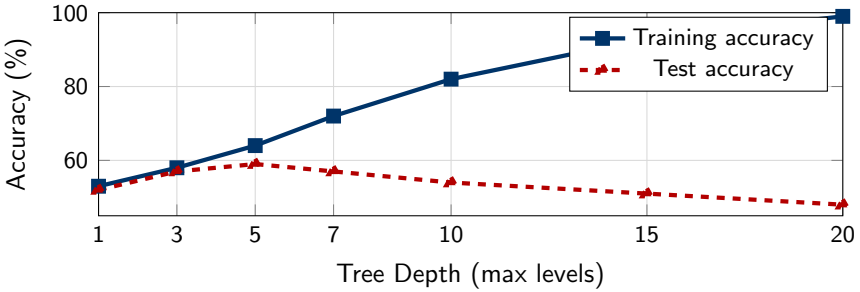
- Which features to split on
- What threshold values to use
- How deep to grow the tree

All from data — no human rules needed.

Key Problem

A single decision tree can grow very deep, memorizing every training example. Result: **overfitting!** A deep tree on financial data might have 99% training accuracy and 48% test accuracy.

Problem: Single Trees Overfit Easily

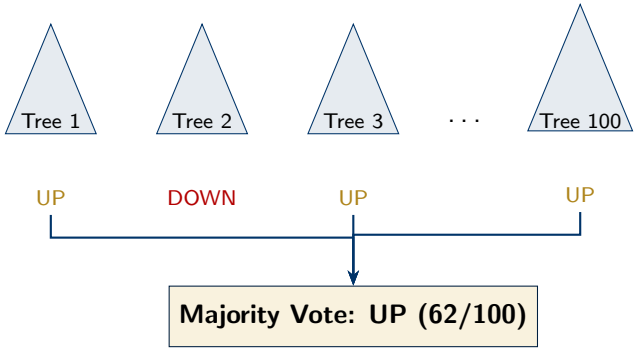


Beyond depth $\sim$5, the tree starts memorizing noise. **Solution:** don't rely on a single tree.

Random Forests: The Wisdom of Crowds

Key Idea

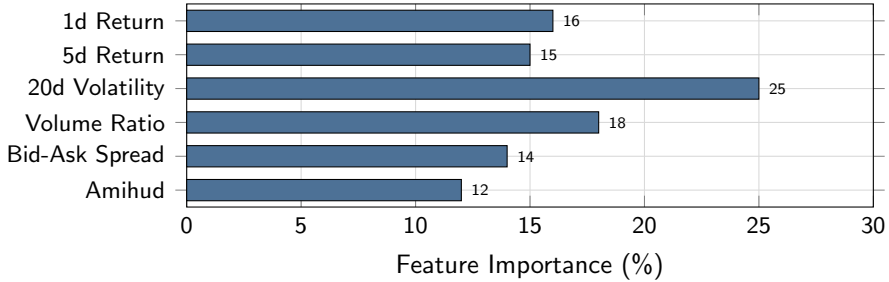
Instead of one tree, grow **100+ trees**, each trained on a **random subset** of the data and features. Take a **majority vote**.



Each tree is weak, but together they're strong. Individual errors cancel out when you

Feature Importance: Which Inputs Matter Most?

Random forests tell us which features contribute most to predictions.

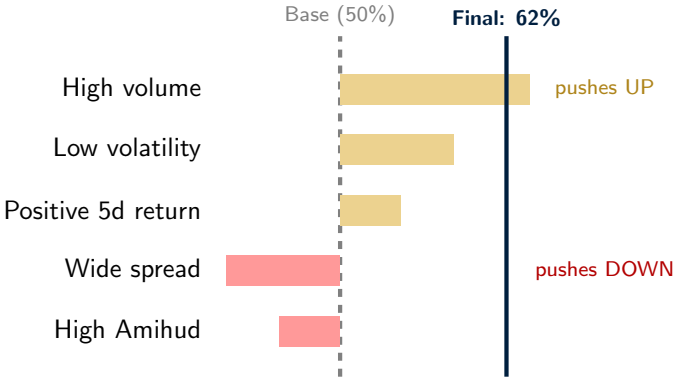


Volatility and volume matter more than past returns. **This is a finding, not something we assumed!**

SHAP Values: Explaining Individual Predictions

What is SHAP?

For each prediction, SHAP tells you **how much each feature pushed the prediction up or down**.



Outline

Preview: Reinforcement Learning & LLMs

Reinforcement Learning (RL)

- Agent learns by **trial and error**
- Takes actions (buy/sell/hold)
- Receives rewards (profit/loss)
- Optimizes long-term cumulative reward
- Like training a dog with treats!

Used for: portfolio management, execution optimization

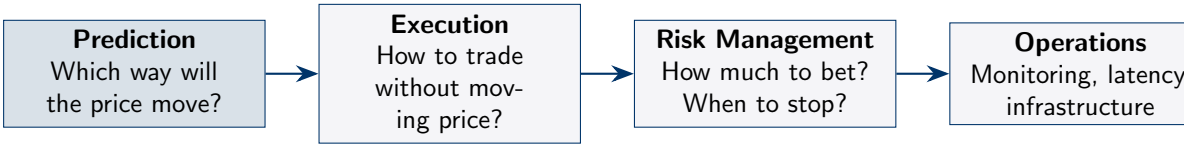
Large Language Models (LLMs)

- Process news, tweets, reports
- Extract **sentiment** at scale
- "Is this article bullish or bearish?"
- Can summarize 1000 articles per second
- Human-like understanding of context

Used for: sentiment analysis, event detection

These are advanced topics — but today's ML foundations apply to all of them.

Key Message: Prediction Is Just the Beginning



Reality Check

Even a "good" prediction model (55% accuracy) is useless without:

- Proper risk management (so one bad trade doesn't wipe you out)
- Efficient execution (so fees and slippage don't eat your edge)
- Continuous monitoring (so you know when the model breaks)

Outline

Hands-On Exercise

Predicting BTC Returns with Random Forests

Python / Jupyter Notebook (scaffolded)

Exercise: Predict BTC Next-Day Returns

Goal: Build a random forest classifier to predict whether BTC's next-day return is positive or negative.

What's provided (pre-loaded in notebook)

- 4 years of daily BTC data (2020–2024)
- Pre-computed features: returns, volume, spread, volatility, Amihud
- Target variable: $y = 1$ if next-day return > 0, else $y = 0$
- All import statements and plotting code

What you code (fill in the blanks)

- Train/test split (time-based, 80/20)
- Fit the random forest model
- Evaluate accuracy and confusion matrix
- Interpret SHAP feature importance

The Dataset: BTC Daily Features

Column	Type	Example Value	Description
`return_1d`	float	-0.023	Yesterday's return
`return_5d`	float	$+0.051$	5-day cumulative return
`return_20d`	float	-0.082	20-day cumulative return
`vol_20d`	float	0.041	20-day realized volatility
`volume_ratio`	float	1.35	Today / 20-day avg volume
`spread`	float	0.0008	Bid-ask spread (%)
`amihud`	float	0.0015	Amihud illiquidity
`target`	int	1	1 = up, 0 = down

1,461 rows (4 years $\times$ 365 days, minus some missing data).

Step 1: Train/Test Split (Your Task)

Fill in the code:

```
# Split point:  80% of the data
split_idx = int(len(df) * 0.80)

# Training data:  first 80%
X_train = df[features].iloc[:split_idx]
y_train = df['target'].iloc[:split_idx]

# Test data:  last 20%
X_test = df[features].iloc[split_idx:]
y_test = df['target'].iloc[split_idx:]
```

Remember

We split by **time**, not randomly! Training = Jan 2020 to Dec 2023. Testing = Jan 2024 to Mar 2026. The model never sees the future.

Step 2: Train and Evaluate the Random Forest

Code (provided):

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(
n_estimators=100, max_depth=5, random_state=42)
model.fit(X_train, y_train)
accuracy = model.score(X_test, y_test)
print(f"Test accuracy:  {accuracy:.1%}")
```

Confusion Matrix

		Predicted	
		Down	Up
Actual	Down	68	72
	Up	63	89

How to read it:

- 89 correct UP predictions
- 68 correct DOWN predictions
- $72 + 63 = 135$ mistakes
- Total accuracy: $\sim$54%

What Does 52–54% Accuracy Mean?

Sounds Terrible?

- A coin flip gives 50%
- 54% is "barely better"
- In most contexts, this would fail

Actually Not Bad!

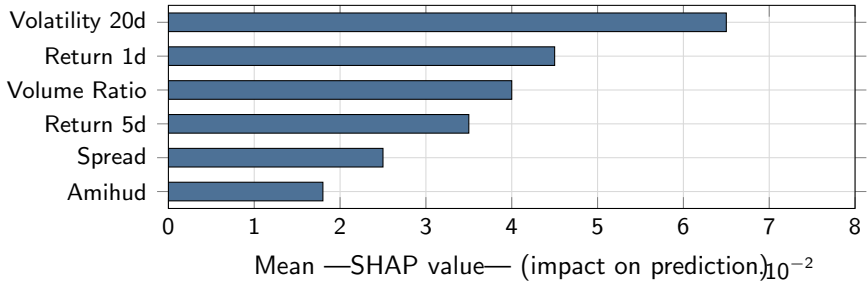
- In finance, 54% is **significant**
- Over 1000 trades, a 4% edge compounds
- Top hedge funds operate at 51–55%
- The key: combine with risk management

Quick Calculation

If you bet \$100 per trade with 54% accuracy and 1:1 risk-reward:
Expected profit per trade $= 0.54 \times 100 - 0.46 \times 100 = \$8$
Over 250 trading days: $250 \times \$8 = \$2{,}000$ expected profit.
But: this ignores fees, slippage, and model degradation.

Step 3: Interpret the SHAP Plot

Your task: Run the SHAP analysis and interpret the results.



Questions for students:

1. Why is volatility the most important feature?
2. Does high volatility predict UP or DOWN?
3. Why might spread and Amihud matter less?

Discussion: AI in Finance

1. **Can AI replace human traders?**
   - What advantages do humans still have?
   - What about rare events (pandemics, wars) with no training data?
2. **Ethical concerns with algorithmic trading**
   - Should high-frequency bots be allowed in crypto?
   - Who is responsible when an AI causes a flash crash?
   - Does AI trading make markets fairer or less fair?
3. **If you had 54% accuracy, would you trade real money?**
   - How much would you risk?
   - How would you know when the model stopped working?

Summary & Day 5 Preview

Today's Key Takeaways

- ML learns patterns from data, not human rules
- **Overfitting** is the biggest risk in financial ML
- Always split train/test **by time**
- Random forests: many weak trees $\rightarrow$ one strong model
- SHAP explains *why* the model predicts what it does
- 52–55% accuracy can be valuable in finance

Reading: [2], Ch. 8–9; [1]

Day 5 Preview: Risk & Regulation

- Value at Risk: measuring the worst case
- Portfolio theory: diversification works (mostly)
- EU MiCA, US regulation, global landscape
- The future: tokenization, AI agents, ZK proofs
- Course wrap-up and group projects

References I

[1] Bank for International Settlements. *Cryptocurrencies and Decentralised Finance*. BIS Annual Economic Report, Chapter III. Bank for International Settlements, 2024.

[2] Marco Di Maggio. *Blockchain, Crypto and DeFi: Bridging Finance and Technology*. Wiley, 2024.

[3] Igor Makarov and Antoinette Schoar. "Trading and Arbitrage in Cryptocurrency Markets". In: *Journal of Financial Economics* 135.2 (2020), pp. 293–319.