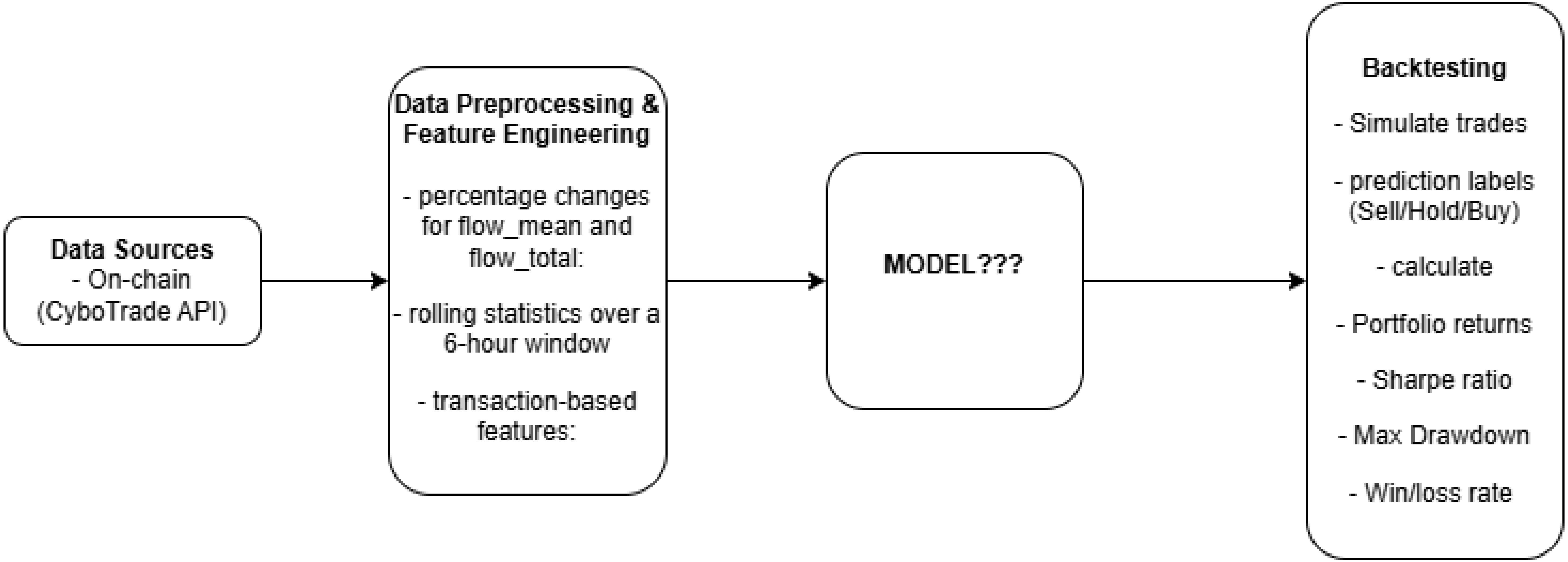
Domain 2

Convolutional Neural Networks for Alpha Strategy Development in Volatile Crypto Markets

Team Arctic Rhinos 1



Data Sources
- On-chain
(CyboTrade API)

Data Preprocessing & Feature Engineering

- percentage changes for flow_mean and flow_total:

- rolling statistics over a 6-hour window

- transaction-based features:

MODEL???

Backtesting

- Simulate trades

- prediction labels (Sell/Hold/Buy)

- calculate

- Portfolio returns

- Sharpe ratio

- Max Drawdown

- Win/loss rate

HMM

Linear Regression / Logistic Regression

K-Means

Transformers

CNN

HMM

HMMs are not the best at modeling complex non-linear patterns, where CNNs, LSTMs, and Transformers perform better.

Linear Regression / Logistic Regression

Assumes fixed relationships between variables, limits ability to adapt to dynamic market conditions that deep learning models handle more flexibly.

K-Means Algo

K-Means assumes fixed clusters and struggles with dynamic patterns, unlike HMMs or LSTMs

Transformers

Require large datasets to perform optimally, making them less effective in data-scarce environments compared to simpler models like Random Forests or Logistic Regression.

CNN

- Learn and extract meaningful patterns from price and indicator sequences without us having to define any state-based assumptions, unlike HMM
- Main goal was to build a system that outputs clear Buy/Hold/Sell predictions that we could be plug into a backtesting engine.
- CNNs are inherently built to handle multi-dimensional inputs, while other model might struggle to scale with higher dimensional feature spaces.
- CNNs can be retrained regularly on new data, enabling them to adapt to market regimes and conditions.

CNN Model Building

- Define a CNN architecture
- Model Compilation
- Data Preparation
- Model Training
- Model Evaluation
- Confusion Matrix

Some Model Details

- Conv1D layer (64 filters, kernel size 3, ReLU activation) with batch normalization and max pooling.
- Conv1D layer (128 filters, kernel size 3, ReLU activation) with batch normalization and max pooling.
- Flatten layer to convert 2D outputs into 1D.
- Dense layer (64 units, ReLU activation) with dropout for regularization.
- Dense layer with 3 output units (one for each class: Sell, Hold, Buy) and softmax activation.

- Model Compilation:
  - Use Adam optimizer with a learning rate of 0.001.
  - Use categorical crossentropy loss for multi-class classification.
  - Track accuracy as the evaluation metric.

Model Demo