Chen Huang

Deep Learning - Financial Time Series application

Use Deep learning to learn an existing strategy

Warning

- Don’t Try this at home!
- Investment involves risk. Make sure you understand the risk before investing.

A little background about me

- I am working for YiBei Investment and Management LTD
- Started in 2012 with 4 people all with CS background, grow into 12 people with different backgrounds
- YiBei is managing around 60-70 Million Yuan (~10 Millions in USD)
- Published Two funds to public in 2017
- Focus on quantitive trading models on commodity, stocks.
- 2017 started VC.

Agenda

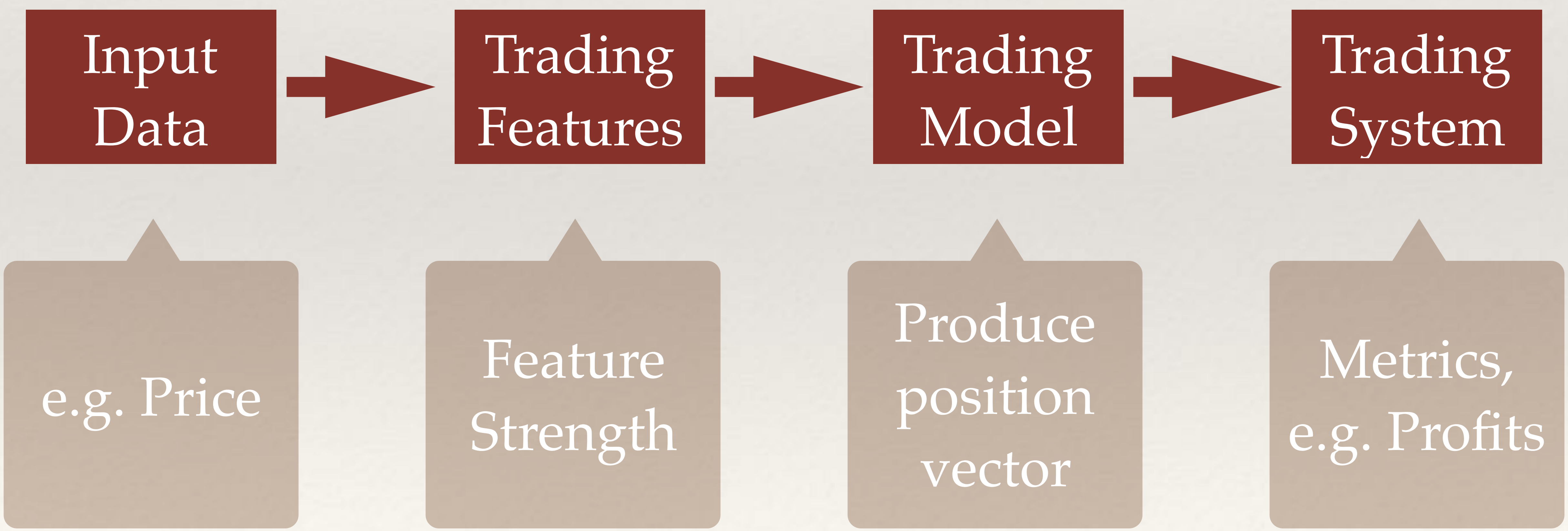
- Introduction on common quantitive trading strategies
  - Background
  - Trading Model development
  - Challenges
- Can deep learning help?
  - LSTM to learn from an existing strategy
  - LRCN network to learn from an existing strategy
- Can deep learning help generate trading strategies?

Terminology

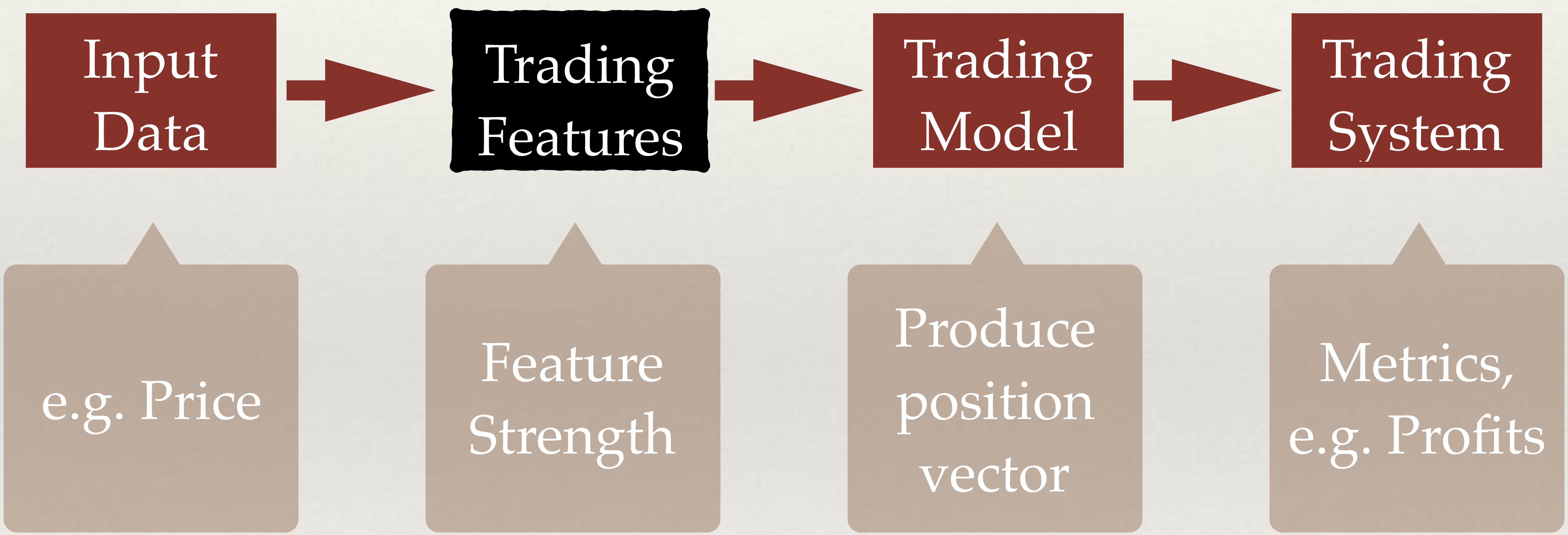
- Future - a financial derivative with leverage option, usually based on some assets, e.g Copper Future
- Price - Price of a single asset. e.g price of 50 bushel Corn
- Feature - a processed input to Model.
- Model - processing features and produces position
- Position - The number of assets holding at any given time
- Actions
  - Long - buy
  - Sell - sell previous purchased asset
  - Short - sell asset by borrowing the asset
  - Cover - buy asset back and return the borrowed asset

Background - Trading Strategy

- Trading Strategy is a program that automates the decision to buy / sell financial assets.

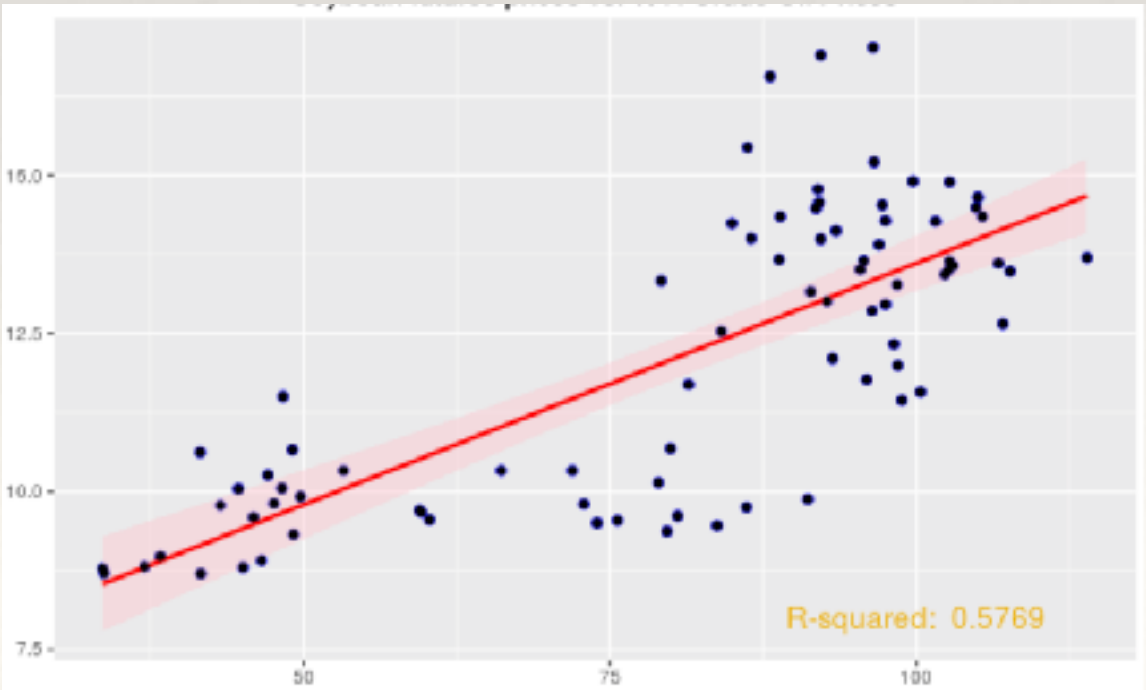


Background - Trading Feature

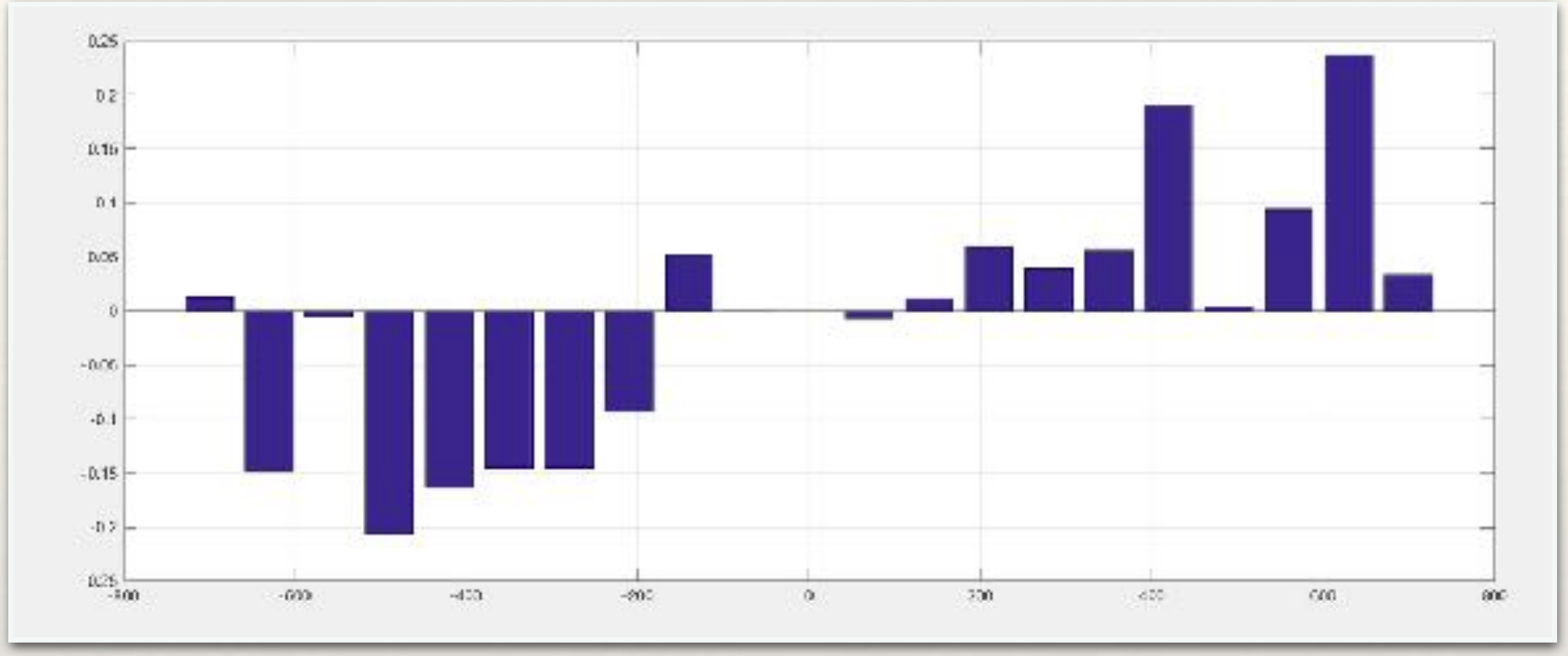


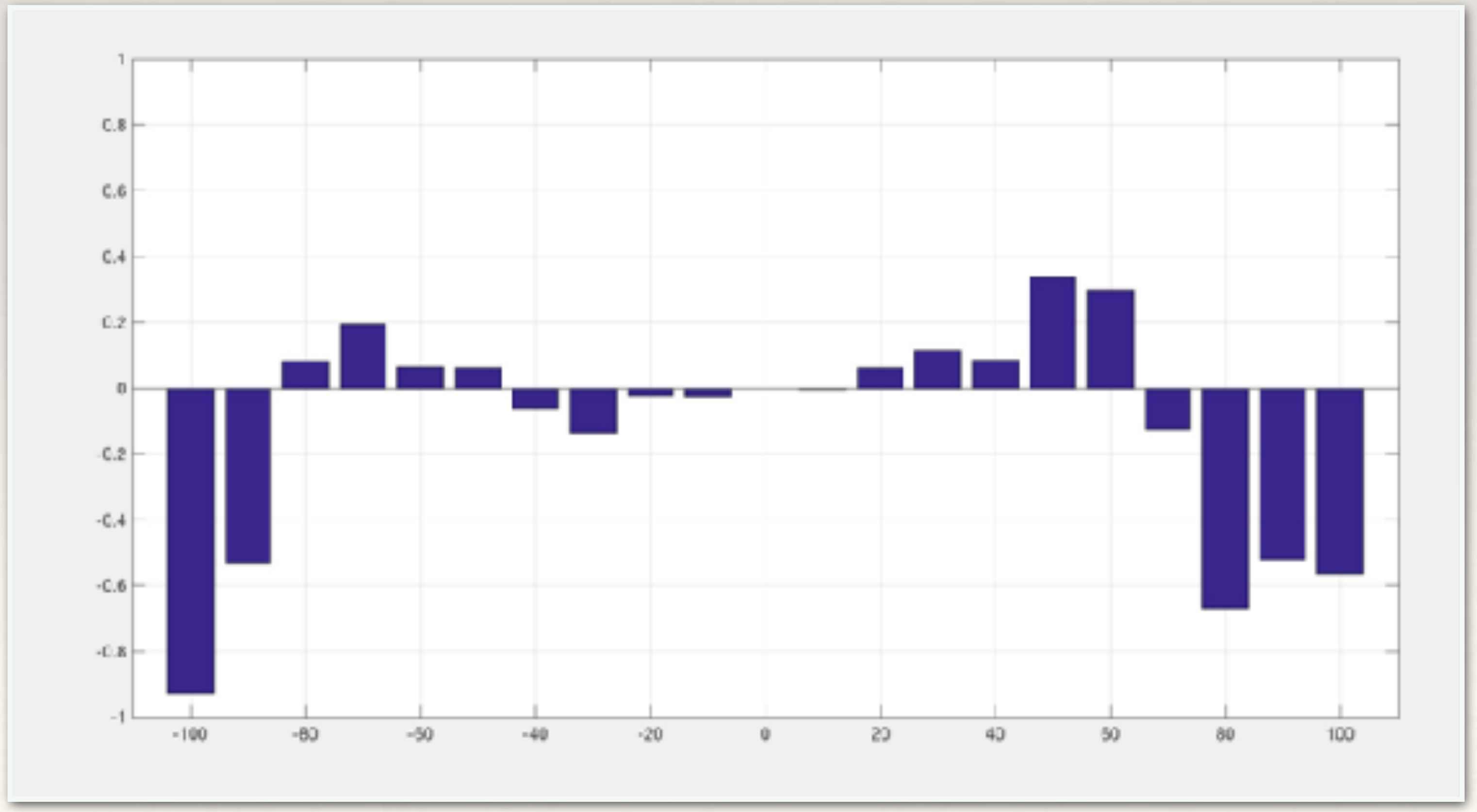
Background - Trading Feature

- Treading Feature provides as input to trading model.
  - e.g. many technical indicators
  - Feature has a strength level
  - A good feature should have a **large absolute** correlation between feature strength and price movement in the near future.

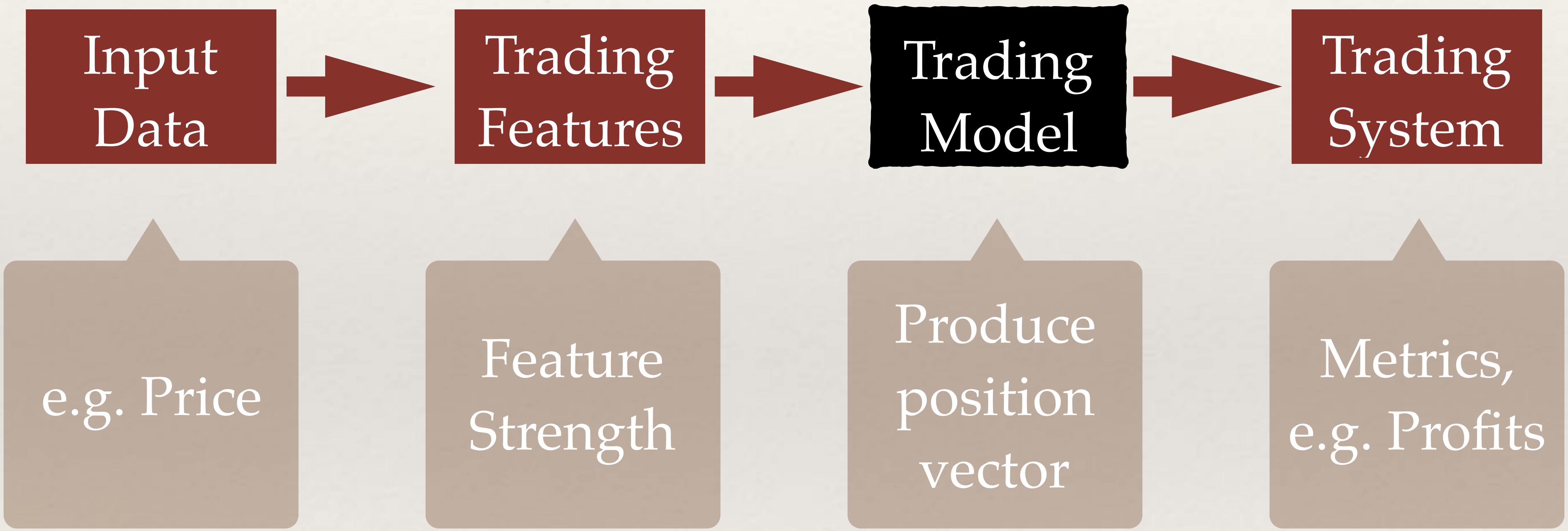


Background - Trading Feature





Background - Trading Model



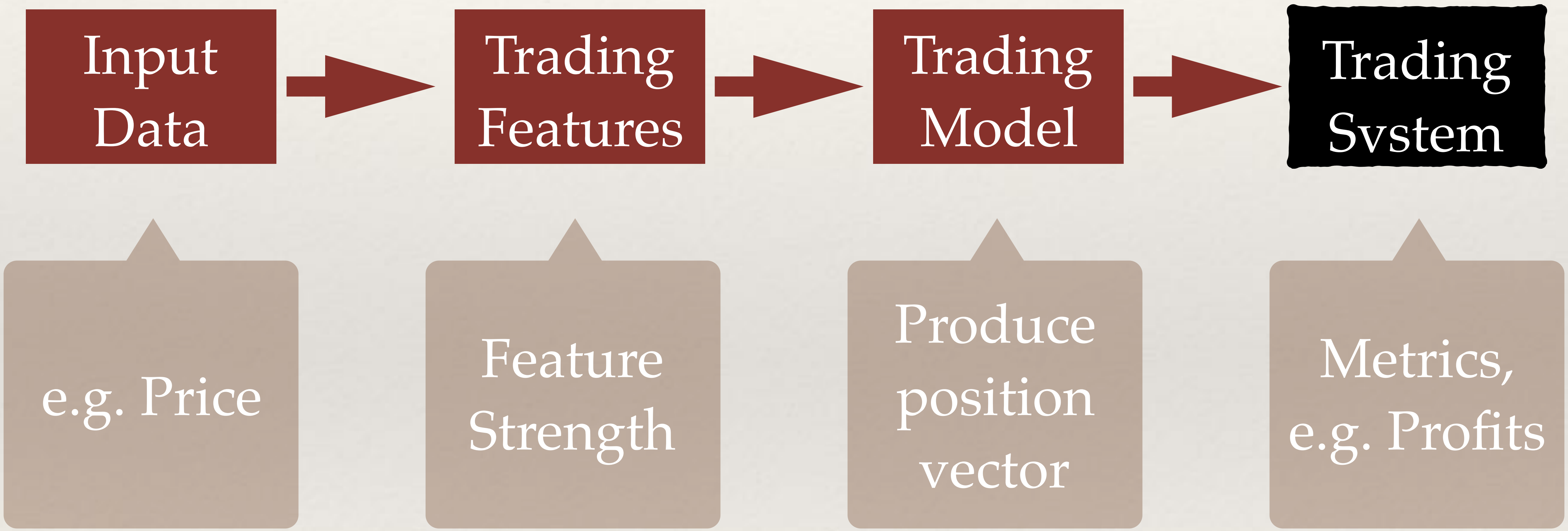
Background - Trading Model

- Trading Model take input from the features and decides what to do with them
  - It output positions, which later translated by the trading system and produces trading actions
  - Trading Model is mostly concerned with the trading logic.

Background - Trading Model

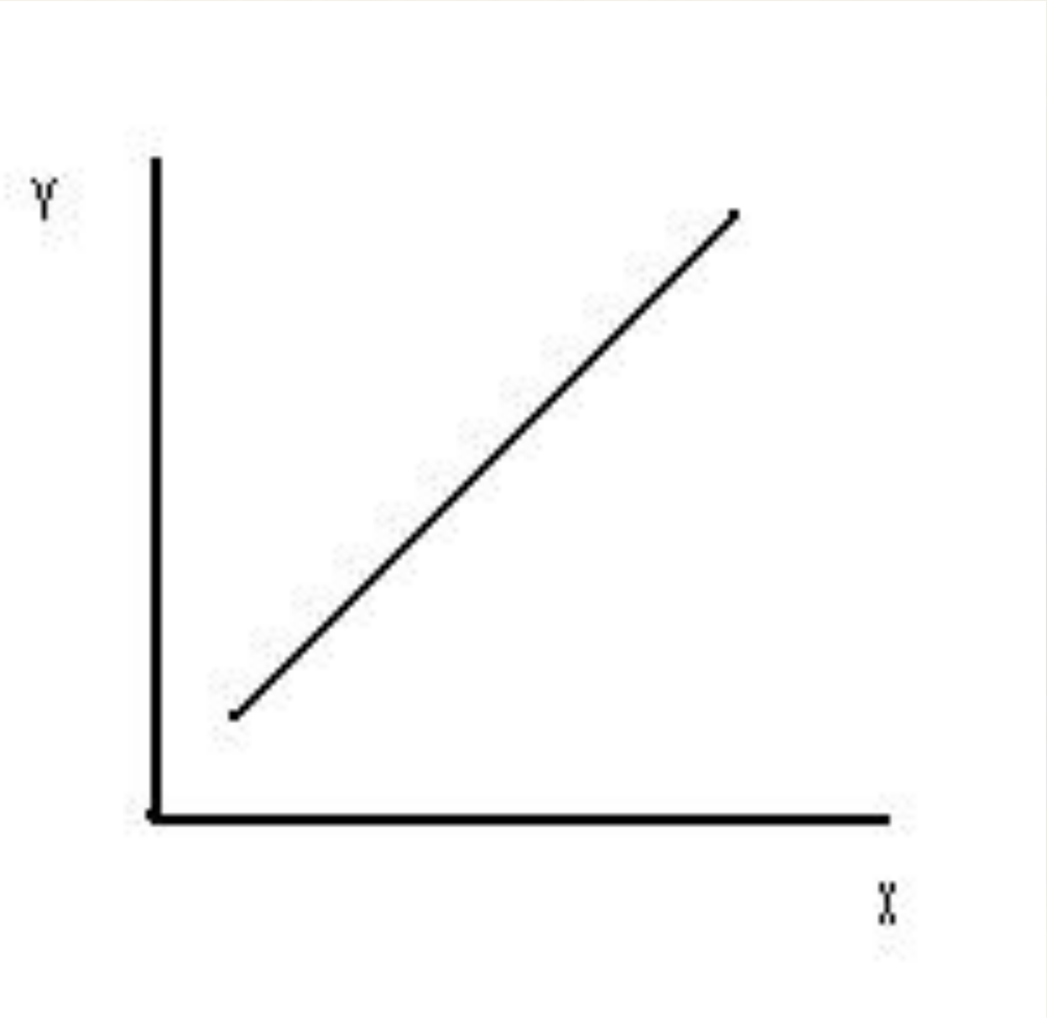
```
bcond1_1 = Close > trendline && Close > shortline;
bcond1_2 = trendline < shortline;
bcond1_3 = abs(shortline - trendline) > myThreshold;
bcond1 = bcond1_1 && bcond1_2 && bcond1_3;
AddColumn( bcond1, "bcond1", format=1);

bcond2 = LinRegSlope(C, myS) > LinRegSlope_coeff * cond3_coeff * lastCoupleDaysATR;
AddColumn( bcond2, "bcond2", format=1);

bcond3 = shortSlope > (LinRegSlope_coeff * lastCoupleDaysATR);
AddColumn( bcond3, "bcond3", format=1);

BSIG = bcond1 && bcond2 && bcond3;
AddColumn( BSIG, "BSIG", format=1);
```

Background - Trading System



Background - Trading System

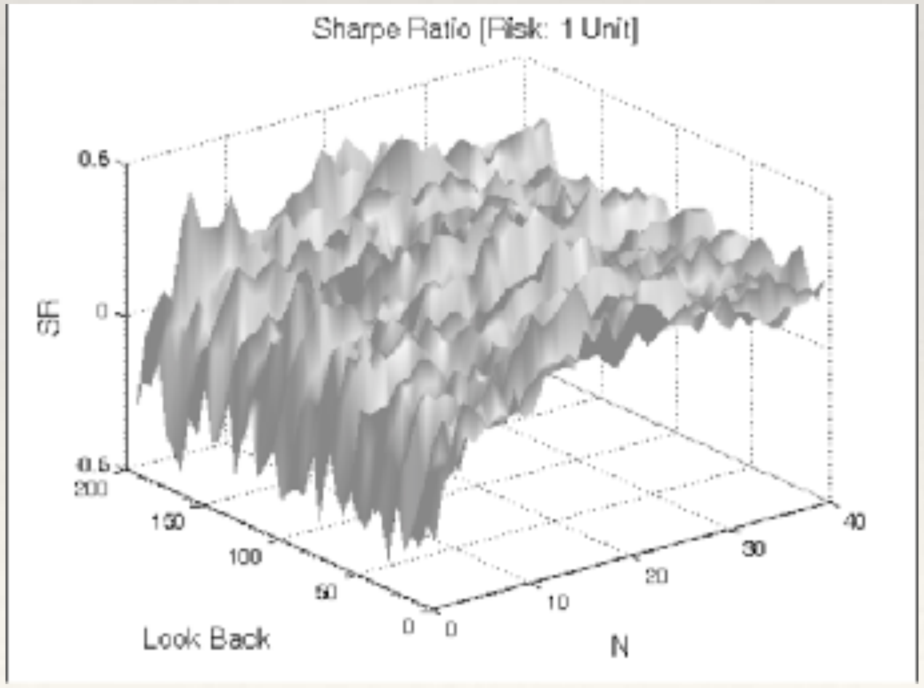
- The main functionality of the trading system is to calculate different metrics, e.g.
  - Return
    - NetProfits
    - Annual Return
  - Risk
    - Max Drawdown (MDD)
    - Standard Error
  - Risk adjusted Return
    - CAR/MDD
    - Profit Factor
  - Frequency(Number of Trades)
  - Overfitting Prevention
    - Consistency (K-ratio)
    - Robustness



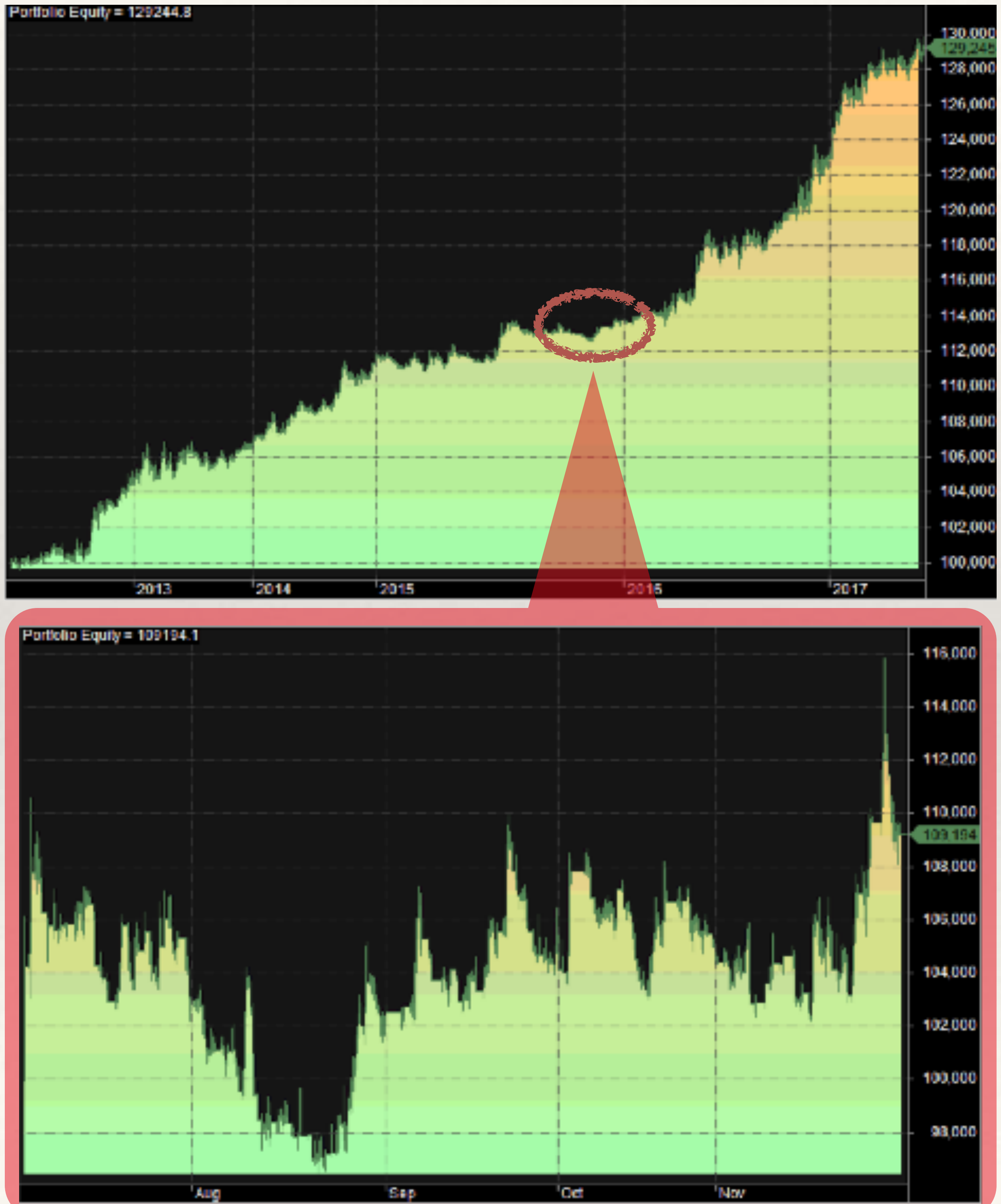
Background - Equity Curve

Equity Curve is the only truth

- Metrics can be deceiving.
- For example, sharpe ratio misses max drawdown.
- Evaluate a strategy involves evaluating multiple metrics are the same time.
- The objective function involves multiple metrics.
- objective function surface is very spiky!



Background - Equity Curve



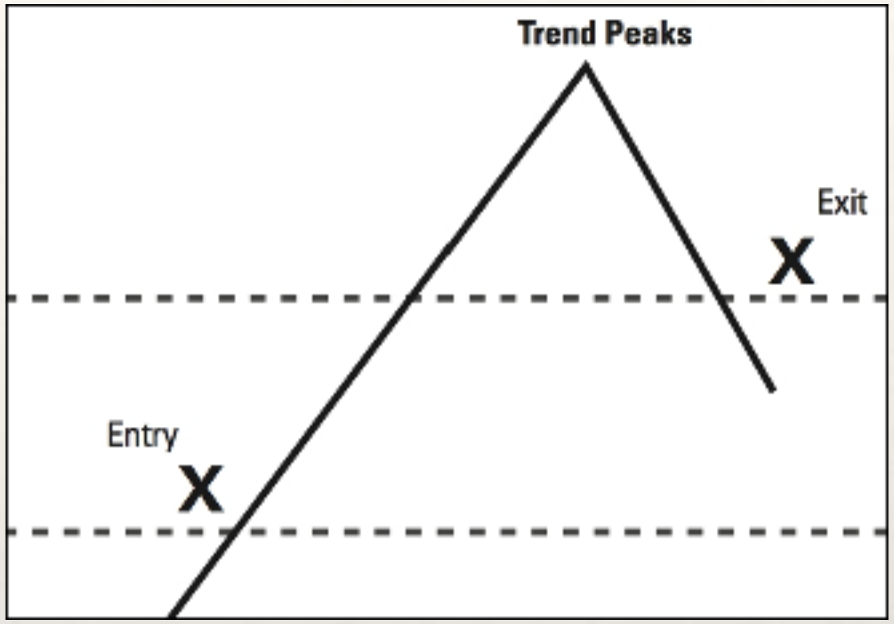
- Trading is not only a technical challenge, but also a **phycological** challenge.
- Under a working strategy, within certain timeframe, the strategy could perform worse.
- How to handle the pressure and take courages bet is beyond trading strategy development

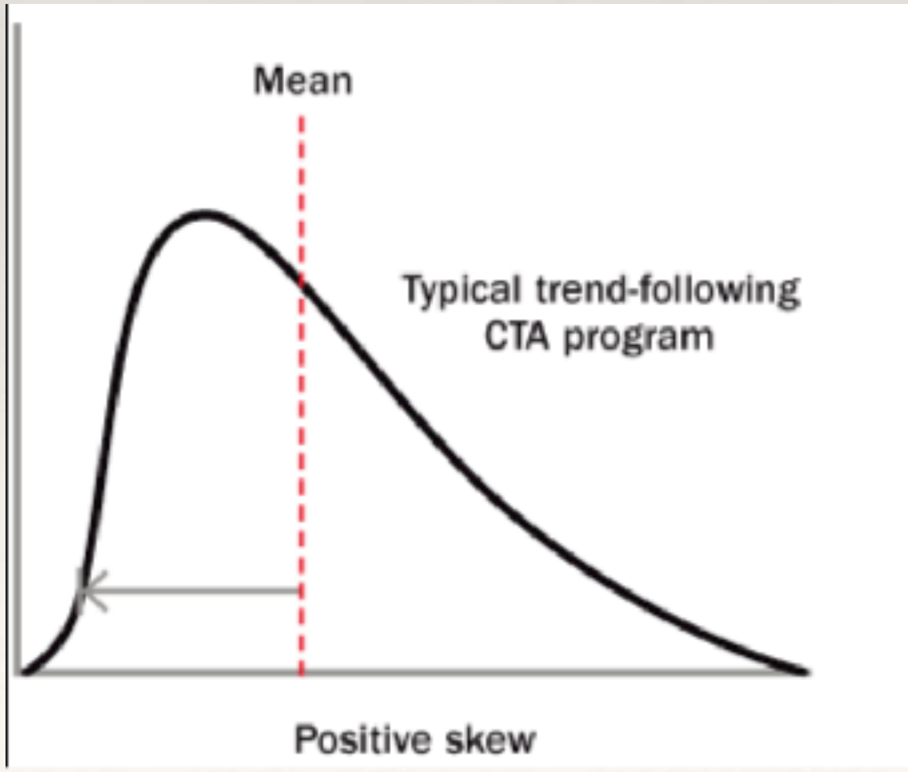
Model Development

- There are many ways to develop a model. Common models based on prices includes
    - Trend Following
    - Mean Reversion
    - Pattern Matching/Statistical Methods

Model Development - Trend Following

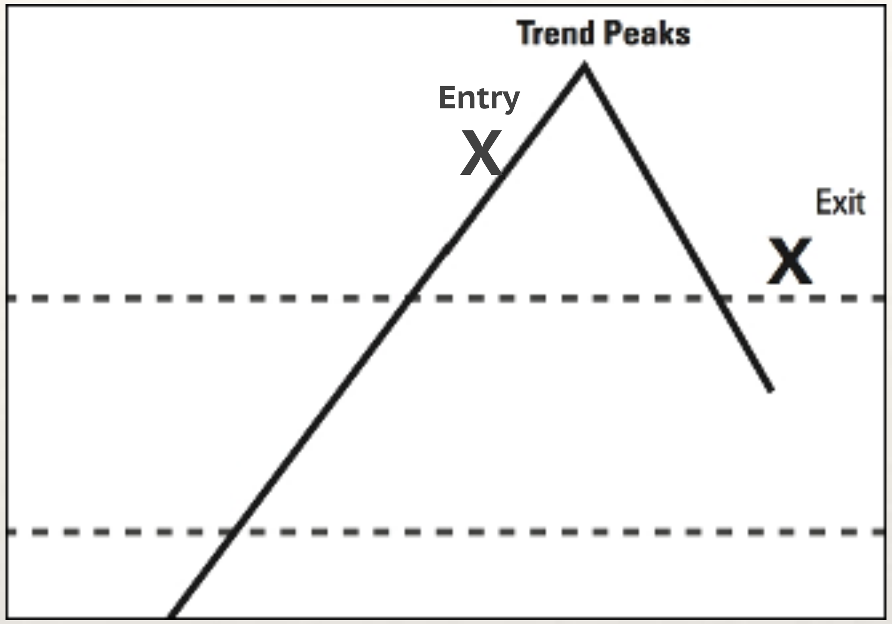
- Trend following is based on the belief that price movement has momentum, the direction of the price moment won't change too soon.
- It's relatively easy to compose a trend following model.
- There are many trend following models because trend can be defined in many ways.





Model Development - Mean Reversion

- Mean reversion is based the assumption that the price will often overshoot and revert back to its mean.
- It's slightly more difficult to compose compared to trend following.
- There are many mean reversion models as it can defined differently.



Model Development

- Trend Following is the completely opposite of the Mean reversion.
- Under market efficient theory, neither of the strategy would work out
- However, does the data show that market is efficient?

Model Development

Model Development

- After a model has been created, an optimization is performed to decide what parameters are the best suitable for certain assets.
- The optimization is based on a objective function, which could be a linear combination of different metrics.
- Then we looked at the top 200 optimization results and hand-pick a couple parameters to trade.

Comparison

Comparison	Trading Strategy	Machine Learning
Model	mathematical model	Neural Network, etc
Optimization	hand-crafted	Gradient Decent, Adam, etc
HyperParameter	Grid Search	Grid Search, Bayesian Optimization, etc

Deep Learning Application in Time Series

- In 2012, I've tried deep neural network as well as reinforcement learning to see if they can be a good model for trading
- The results is not promising. reinforcement approach didn't coverage and deep neural network doesn't produce a results that is tradable after slippage and commission.
- This time I am trying to see if neural network is able to learn trend following strategy.

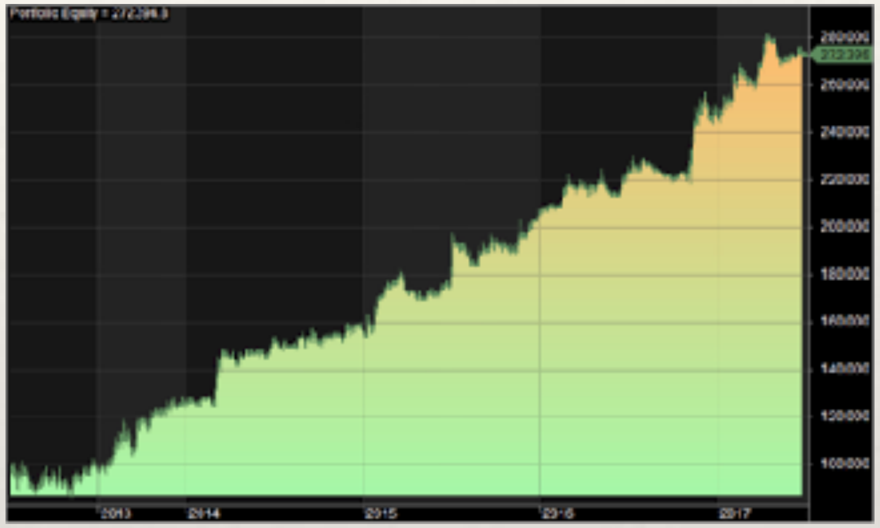
Deep Learning Application in Time Series

- First, we picked one of our current trading trend-following strategy based on moving average and an extra linear regression line.
- Two moving averages and linear regression line decides whether a trend is formed from past data.
- Once the condition met, place the trade in the direction of the short moving average.
- Exit the position when maximum drawdown for this specific position exceeds a certain threshold.

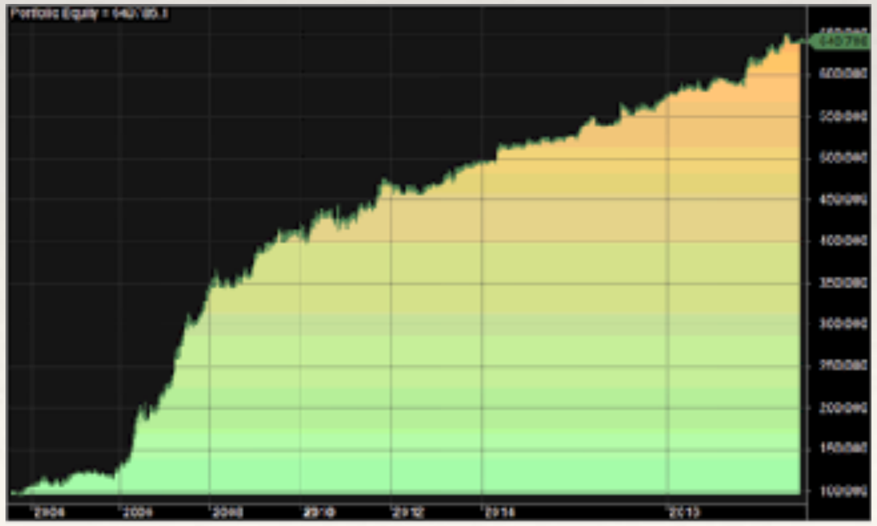
Deep Learning - Target function

```python
def strategy(self):

    """
    策略的逻辑
    """

    recentATR = ATR(self.C, self.H, self.L, 100, False)
    threshold = self.optimize("threshold_multiplier") * recentATR
    linreg_slope_coeff = self.optimize("linreg_slope_coeff")
    linreg_lookback = int(self.optimize("linreg_lookback"))
    long_period = int(self.optimize("longPeriod"))

    short_period = int(self.optimize("short_ratio") * long_period)
    short_line = MA(self.C, short_period)
    long_line = MA(self.C, self.optimize("longPeriod"))

    close_slope = LinRegSlope(self.C, short_period)
    short_slope = LinRegSlope(short_line, linreg_lookback)

    # Long logic
    bcond1_1 = (self.C > long_line) & (self.C > short_line)
    bcond1_2 = long_line < short_line
    bcond1_3 = abs(short_line - long_line) > threshold
    bcond1 = bcond1_1 & bcond1_2 & bcond1_3
    bcond2 = LinRegSlope(self.C, short_period) > linreg_slope_coeff * self.optimize("cond3_coeff") * recentATR
    bcond3 = short_slope > linreg_slope_coeff * recentATR
    BSIG = bcond1 & bcond2 & bcond3

    # Short logic
    scond1_1 = (self.C < long_line) & (self.C < short_line)
    scond1_2 = long_line > short_line
    scond1_3 = abs(short_line - long_line) > threshold
    scond1 = scond1_1 & scond1_2 & scond1_3
    scond2 = LinRegSlope(self.C, short_period) < (-1) * linreg_slope_coeff * self.optimize("cond3_coeff") * recentATR
    scond3 = short_slope < (-1) * linreg_slope_coeff * recentATR
    SSIG = scond1 & scond2 & scond3

    self.BUY = BSIG
    self.SHORT = SSIG

    sigs = MoveStop(self.C, self.BUY, self.SHORT, self.SELL | self.COVER, 100)
    return sigs.values
```

This is the target function.



Equity Curve from 2012



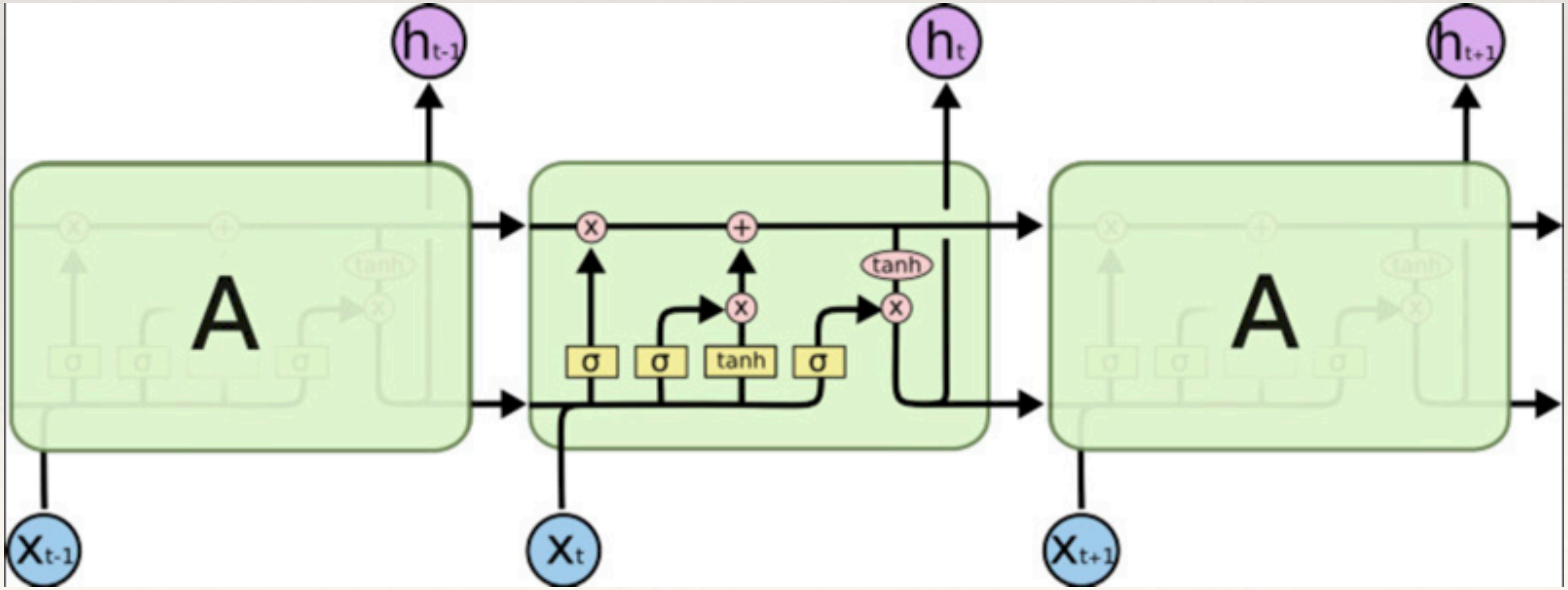
Equity Curve from 2004

LSTM - Financial Time Series

- **input:** Copper minute prices in the format of OHLC (Open, High, Low, Close) of Shanghai Future Exchange from 2012 to July 2017, total of 500310 records.
- Training data is compose the first 35000 records, and testing data is the reset 15000 records. The last 310 records are ignored due to batch size. 20% Training data is further split into validation set without shuffle.
- **output** the positions as a vector.

LSTM - Time Series

- One of the structure that comes in mind for time series would be LSTM.
- It is capable to learn from past experience to predict time series.



Results - Learning from a Strategy

Target: Strategy	Accuracy	Metric	Optimizer	Config	Layers
LSTM (Regression)	30.03%	0.3737 (MSE)	SGD	LR: 1e-8 Decay: 1e-9 Momentum: 0.9	LSTM-128 LSTM-128 LSTM-32 Dense-32 Dense-32 Dense-1
LSTM (Classifier)	28.25%	1.0989 (CrossEntropy)	SGD	LR: 1e-8 Decay: 1e-9 Momentum: 0.9	LSTM-128 LSTM-128 LSTM-32 Dense-32 Dense-32 Dense-3

Learning - Trading Strategy

- It seems that LSTM is not able to learn the strategy function.
- What part of the strategy can't be learned?
- The strategy is composed of features, entry logic and exit logic.
- Each part is tested to see if they can be learned.

Features Learning

```python
def strategy(self):

    """
    策略的逻辑
    """

    recentATR = ATR(self.C, self.H, self.L, 100, False)
    threshold = self.optimize("threshold_multiplier") * recentATR
    linreg_slope_coeff = self.optimize("linreg_slope_coeff")
    linreg_lookback = int(self.optimize("linreg_lookback"))
    long_period = int(self.optimize("longPeriod"))

    short_period = int(self.optimize("short_ratio") * long_period)
    short_line = MA(self.C, short_period)
    long_line = MA(self.C, self.optimize("longPeriod"))

    close_slope = LinRegSlope(self.C, short_period)
    short_slope = LinRegSlope(short_line, linreg_lookback)

    # Long logic
    bcond1_1 = (self.C > long_line) & (self.C > short_line)
    bcond1_2 = long_line < short_line
    bcond1_3 = abs(short_line - long_line) > threshold
    bcond1 = bcond1_1 & bcond1_2 & bcond1_3
    bcond2 = LinRegSlope(self.C, short_period) > linreg_slope_coeff * self.optimize("cond3_coeff") * recentATR
    bcond3 = short_slope > linreg_slope_coeff * recentATR
    BSIG = bcond1 & bcond2 & bcond3

    # Short logic
    scond1_1 = (self.C < long_line) & (self.C < short_line)
    scond1_2 = long_line > short_line
    scond1_3 = abs(short_line - long_line) > threshold
    scond1 = scond1_1 & scond1_2 & scond1_3
    scond2 = LinRegSlope(self.C, short_period) < (-1) * linreg_slope_coeff * self.optimize("cond3_coeff") * recentATR
    scond3 = short_slope < (-1) * linreg_slope_coeff * recentATR
    SSIG = scond1 & scond2 & scond3

    self.BUY = BSIG
    self.SHORT = SSIG

    sigs = MoveStop(self.C, self.BUY, self.SHORT, self.SELL | self.COVER, 100)
    return sigs.values
```

- Features calculation is highlighted to the left
- We picked Moving Average since it's the most commonly used technical indicator and also a basis for most other indicators

$$
\begin{aligned}
\bar{p}_{\text{SM}} &= \frac{p_M + p_{M-1} + \cdots + p_{M-(n-1)}}{n} \\
&= \frac{1}{n}\sum_{i=0}^{n-1} p_{M-i}
\end{aligned}
$$

Results - Learning Features

Target: SMA 20	Metric (MSE)	Optimizer	Config	Layers
LSTM (Regression)	3.9949 (Doesn’t Converge)	SGD Adam AdaDelta		
FullyConnected (Regression)	6.9339e-04 (MSE)	SGD	LR: 1e-7 Decay: 1e-8 Momentum: 0.9	Dense128 Dense64 Dense32 Dense1

Entry Logic Learning

```python
def strategy(self):

    """
    策略的逻辑
    """

    recentATR = ATR(self.C, self.H, self.L, 100, False)
    threshold = self.optimize("threshold_multiplier") * recentATR
    linreg_slope_coeff = self.optimize("linreg_slope_coeff")
    linreg_lookback = int(self.optimize("linreg_lookback"))
    long_period = int(self.optimize("longPeriod"))

    short_period = int(self.optimize("short_ratio") * long_period)
    short_line = MA(self.C, short_period)
    long_line = MA(self.C, self.optimize("longPeriod"))

    close_slope = LinRegSlope(self.C, short_period)
    short_slope = LinRegSlope(short_line, linreg_lookback)

    # Long logic
    bcond1_1 = (self.C > long_line) & (self.C > short_line)
    bcond1_2 = long_line < short_line
    bcond1_3 = abs(short_line - long_line) > threshold
    bcond1 = bcond1_1 & bcond1_2 & bcond1_3
    bcond2 = LinRegSlope(self.C, short_period) > linreg_slope_coeff * self.optimize("cond3_coeff") * recentATR
    bcond3 = short_slope > linreg_slope_coeff * recentATR
    BSIG = bcond1 & bcond2 & bcond3

    # Short logic
    scond1_1 = (self.C < long_line) & (self.C < short_line)
    scond1_2 = long_line > short_line
    scond1_3 = abs(short_line - long_line) > threshold
    scond1 = scond1_1 & scond1_2 & scond1_3
    scond2 = LinRegSlope(self.C, short_period) < (-1) * linreg_slope_coeff * self.optimize("cond3_coeff") * recentATR
    scond3 = short_slope < (-1) * linreg_slope_coeff * recentATR
    SSIG = scond1 & scond2 & scond3

    self.BUY = BSIG
    self.SHORT = SSIG

    sigs = MoveStop(self.C, self.BUY, self.SHORT, self.SELL | self.COVER, 100)
    return sigs.values
```

- Entry logic contains both Long and Short direction
- The basic logics are the same but opposite between long and short.
- The operators used in the entry logics includes, less operator, and operator, multiplication,

Results - Learning Entry Logic

Target: Entry Logic	Metric (Accuracy)	Optimizer	Config	Layers
LSTM (Regression)	N/A	N/A	N/A	N/A
FullyConnected (Regression)	96.057%	Adam	LR: 1e-9	Dense-128 Dense-64 Dense-32 Dense-3

Exit Logic Learning

```
def strategy(self):

    """
    策略的逻辑
    """

    recentATR = ATR(self.C, self.H, self.L, 100, False)
    threshold = self.optimize("threshold_multiplier") * recentATR
    linreg_slope_coeff = self.optimize("linreg_slope_coeff")
    linreg_lookback = int(self.optimize("linreg_lookback"))
    long_period = int(self.optimize("longPeriod"))

    short_period = int(self.optimize("short_ratio") * long_period)
    short_line = MA(self.C, short_period)
    long_line = MA(self.C, self.optimize("longPeriod"))

    close_slope = LinRegSlope(self.C, short_period)
    short_slope = LinRegSlope(short_line, linreg_lookback)

    # Long logic
    bcond1_1 = (self.C > long_line) & (self.C > short_line)
    bcond1_2 = long_line < short_line
    bcond1_3 = abs(short_line - long_line) > threshold
    bcond1 = bcond1_1 & bcond1_2 & bcond1_3
    bcond2 = LinRegSlope(self.C, short_period) > linreg_slope_coeff * self.optimize("cond3_coeff") * recentATR
    bcond3 = short_slope > linreg_slope_coeff * recentATR
    BSIG = bcond1 & bcond2 & bcond3

    # Short logic
    scond1_1 = (self.C < long_line) & (self.C < short_line)
    scond1_2 = long_line > short_line
    scond1_3 = abs(short_line - long_line) > threshold
    scond1 = scond1_1 & scond1_2 & scond1_3
    scond2 = LinRegSlope(self.C, short_period) < (-1) * linreg_slope_coeff * self.optimize("cond3_coeff") * recentATR
    scond3 = short_slope < (-1) * linreg_slope_coeff * recentATR
    SSIG = scond1 & scond2 & scond3

    self.BUY = BSIG
    self.SHORT = SSIG

    sigs = MoveStop(self.C, self.BUY, self.SHORT, self.SELL | self.COVER, 100)
    return sigs.values
```

- Trailing stop is one of the common exit strategy.
- Position is exited when the maximum drawdown exceed a certain threshold
- The position can be open for a long time if maximum drawdown never exceeded the threshold

Results - Learning Exit Logic

Target: Exit Logic	Metric (Accuracy)	Optimizer	Config	Layers
LSTM (Classifier)	79.21%	Adam	lr:1e-9	6 LSTM Layers + 2 Dense layer
FullyConnected (Classifier)	77.68%	Adam	lr: 1e-9	Dense-128 Dense-128 Dense-128 Dense-3
LSTM + Dense (Classifier)	79.21%	Adam	lr:1e-9	6 LSTM Layers + 2 Dense layer

The End

- Q&A