**ML ideas**

* Timing trades requires anaylizing patterns in a stocks price history as well as societal opinion of the underlying company
* With this, I would like to design a model that uses two types and modalities of data
  + **News/investor blog data** (Text)
  + **Stock price history data** (numeric)
* This means that the overall model will need to be composed of two smaller models that handle each task seperately

**The main model**

* The purpose of the main model is to combine the data from the two encoder models into an action (such as buying or selling stock)
* **There are two main ways to combine the data**
  + **Early Fusion:** combine the outputs from each submodel before feeding them to the decision making model
    - **Easier to implement**
    - Better when the two modalities interact
  + **Late fusion:** each submodel produces a decision (buy sell hold) and then the final decision layer combines the votes from each submodel
* **Making buy/sell hold decision with the model**
  + Use reinforcement learning to make decisions
  + Create an enviornment to simulate stock trading
  + **Actor critic network:** 
    - **Actor:** decides an action (buy/sell/hold)
    - **Critic:** estimates the value of the current state-action pair (Computes the reward function value)

**The text analyzer**

* BERT/DistilBERT are good models for semantic encoding of text
* **Sentiment analysis problem:** classify each news article into categories such as positive, neutral or negative
* **Use LORA to fine tune the model efficiently**

**The Price history analyizer**

* Consider LSTM/GRU models
  + These can model temporal depedencies in the price history
  + Treat this a sequence prediction problem where each time step consists of features such as (open, close, high, low, volume or other techincal indicators)

**Making the trading envorment (Trading gym)**

* We need to turn the stock market simulator into a RL envornment
* **State Space representation**
  + A state is the stock enviornments description at a given time
    - Cash holdings
    - Portfolio information
    - Stock price history (last N days)
    - **Technical indicators:** RSI, MACD
  + These form a state input vector to the RL agent
* **Action space**
  + These define the actions that the agent can take
    - **Buy:** buy X shares of foo
    - **Sell:** sell X shares of foo
    - **Hold:** wait until the next trading period
* **Reward function**
  + **Profit/Loss:** new\_portfolio\_value - old\_portolio\_value
  + **Risk:** regularizer for making risky trades
  + **Risk adjusted return:** Sharpe ratio or maximum drawdown
* **Transition Function**
  + Compute how the current state changes after taking the given action
    - Implement the env.step(Action a) function
    - Takes the action (portfolio update)
    - Calculates the reward and state and returns them
* **Default starting state**
  + The state that each agent spawns in

**Training the model**