I added new API keys

Pub: a33d1218dff1c8f6fee41f687f76f12bc6d1fd62

Secret: 9b21a985a6cd5519f4962a020516fed5e6729c203e140c52d3dd06e4468d8477d0dafefefd82faf8

**collectRaw()**

* Execute every minute for one hour
  + Should have 59-60 datapoints depending on how the time lines up
  + The 60 data points should all be within the same hour, so don’t start collectRaw() at 1:14 or something.
* Calculate the max, min, and close of the 60 datapoints
* The open should be the previous close
* Create an arra y that keeps information as follows when the hour is up:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Time | Open | High | Low | Close |
| 1/13/2013 13:00-14:00 | 125 | 133 | 122 | 124 |

* Clear last 60 data points and start again for the next hour

**trend1()**

* TR1 = max((High-Low),Abs(High-Open),Abs(Low-Open))
* Calculate +DM1 and -DM1

if (Current\_High-Previous\_High)>(Previous\_Low-Current\_Low)

minusDM1 = 0

if (Current\_High-Previous\_High)>0

plusDM1 = Current\_High-Previous\_High

else

plusDM1 = 0

else

plusDM1 = 0

if (Previous\_Low-Current\_Low)>0

minusDM1 = Previous\_Low-Current\_Low

else

minusDM1 = 0

**trend14()**

All of the smoothing techniques in this part should be the same. Because they are smoothed using the previous 14 data, the array will be missing first 13 data points for the smoothed values. All calculations later in the array will be using this data, so the entire rest of the array will be missing the first 13 datapoints.

* TR14
  + First value is =average(Previous 13 TR1 values and current one)
    - This would be a moving average of 14 periods that include the current TR1
  + Subsequent value = (Previous\_TR14\*13/14)+Current\_TR1
* minusDM14
  + First value is =average(Previous 13 minusDM1 values and current one)
    - This would be a moving average of 14 periods that include the current minusDM1
  + Subsequent value = (Previous\_minusDM14\*13/14)+Current\_minusDM1

* plusDM14
  + First value is =average(Previous 13 plusDM1 values and current one)
    - This would be a moving average of 14 periods that include the current plusDM1
  + Subsequent value = (Previous\_plusDM14\*13/14)+Current\_plusDM14

**ADX()**

* plusDI14 = (Current\_plusDM14/Current\_TR14)\*100
* minusDI14 = (Current\_minusDM14/Current\_TR14)\*100
* diffDI14 = abs(Current\_plusDI14-Current\_minusDI14)
* sumDI14 = Current\_plusDI14+Current\_minusDI14
* DX = (diffDI14/sumDI14)\*100
* ADX
  + First value is =average(Previous 13 DX values and current one)
    - This would be a moving average of 14 periods that include the current DX
  + Subsequent value = (Previous\_ADX\*13+Current\_DX)/14
    - Note this is different than the smoothing above

**Decision()**

* If (plusDI14>minusDI14)

Direction = “Up”

else

Direction = “Down”

* Calculate trendExist

if (Current\_ADX > 25)

if (previous\_trendExist = “newTrend” or previous\_trendExist = “currentTrend”)

curent\_trendExist = “currentTrend”

else

current\_trendExist = “newTrend”

else

current\_trendExist = “noTrend”

**Execute()**

* If (current\_trendExist = “newTrend”)

if (Direction = “Up”)

Buy DOGE with BTC

else

Sell DOGE for BTC

else

Do nothing

* For the time being, maybe it would just say whether we bought or sold to see how well this worked instead of actually trading

**arraySize()**

* The array should keep past data of 20 rows just to be safe, but can be cut at top to keep from using so much memory
* For the paper trading session, it should keep 200 rows so that we can see how it held up

=====================================================================

from Cryptsy import Cryptsy

# Pub: a33d1218dff1c8f6fee41f687f76f12bc6d1fd62

# Secret: 9b21a985a6cd5519f4962a020516fed5e6729c203e140c52d3dd06e4468d8477d0dafefefd82faf8

#TODO: call this at the new hour

def collectRaw():

datapoints = []

temp = 0

while len(datapoints) <= 60:

temp = collectRawHelper()

datapoints.append(temp)

threading.Timer(60, collectRawHelper).start()

#exit loop when all 60 are a gathered

maxPoint = max(datapoints)

minPoint = min(datapoints)

closePoint = datapoints[-1]

openPoint = datapoints[0] # first element of hour; NOT close of previous hour

return openPoint, maxPoint, minPoint, closePoint

#returns the last publically excecuted trade

def collectRawHelper():

#TODO get last publically excecuted trade

#takes in high/low/openPoint/prev\_high/prev\_low

#returns plusDM1 and Minus DM1

def trend1(openPoint,high,low,prev\_high,prev\_low):

tr1 = max(high - low,abs(high - openPoint),Abs(low - openPoint))

# Calculate +DM1 and -DM1

if (high - prev\_high)>(prev\_low - low):

minusDM1 = 0

if (high-prev\_high)>0:

plusDM1 = high - prev\_high

else:

plusDM1 = 0

else:

plusDM1 = 0

if (prev\_low - low)>0:

minusDM1 = prev\_low - low

else:

minusDM1 = 0

return tr1, plusDM1, minusDM1

#takes in prev\_tr14/prev\_minusDM14/prev\_plusDM14/tr1, plusDM1, minusDM1

#returns plusDM14 and minus DM14

def trend14(tr1, plusDM1, minusDM1, prev\_tr14,prev\_plusDM14,prev\_minusDM14):

tr14 = (prev\_tr14\*13/14)+tr1

minusDM14 = (prev\_minusDM14\*13/14)+minusDM1

plusDM14 = (prev\_plusDM14\*13/14)+plusDM1

return tr14, plusDM14, minusDM14

def ADX(tr14, plusDM14, minusDM14, prev\_ADX):

plusDI14 = plusDM14/tr14\*100

minusDI14 = minusDM14/tr14\*100

diffDI14 = abs(plusDI14-minusDI14)

sumDI14 = plusDI14+minusDI14

DX = (diffDI14/sumDI14)\*100

ADX = (prev\_ADX\*13+DX)/14

return plusDI14, minusDI14, ADX

def decision(plusDI14,minusDI14, ADX, previous\_trendExist):

if (plusDI14>minusDI14):

direction = "Up"

else:

direction = "Down"

if (ADX > 25):

if (previous\_trendExist == "newTrend" or previous\_trendExist == "currentTrend"):

curent\_trendExist = "currentTrend"

else:

current\_trendExist = "newTrend"

else:

current\_trendExist = "noTrend"

return direction, current\_trendExist

def execute(direction, current\_trendExist):

if (current\_trendExist == "newTrend"):

if (direction == "Up"):

# Buy DOGE with BTC

else:

# Sell DOGE for BTC

else:

# Do nothing

#start at new hour

def start():

openPoint, high, low, closePoint = collectRaw()

#write to csv

prev\_high, prev\_low = #read from csv

tr1, plusDM1, minusDM1 = trend1(openPoint, high, low, prev\_high, prev\_low)

#write to csv

prev\_tr14,prev\_minusDM14,prev\_plusDM14 = #read from csv

tr14, plusDM14, minusDM14 = trend14(tr1, plusDM1, minusDM1, prev\_tr14,prev\_plusDM14,prev\_minusDM14);

#write to csv

prev\_ADX = #read from csv

plusDI14, minusDI14, ADX = ADX(tr14, plusDM14, minusDM14, prev\_ADX)

#write to csv

previous\_trendExist = #read from csv

direction, current\_trendExist = decision(plusDI14,minusDI14, ADX, previous\_trendExist)

#write to csv

execute(direction, current\_trendExist)

start()