from QuantConnect.Data.Custom.SmartInsider import \*

class SmartInsiderAlphaModel:

def \_\_init\_\_(self):

self.altDataSymbols = {}

def Update(self, algorithm, data):

insights = []

## Company buyback intentions and transaction data are provided

## by Smart Insider. A "buy-back" is when a company repurchases

## its own stock. It reduces the number of shares available to

## other investors, which in theory should reduce the supply of

## shares and increase the stock price.

## Smart Insider has two data sets available to use in your algorithm.

## The Intentions data set is an announcement that establishes the intention

## to buy-back shares of the company. When the buy-back occurs this triggers

## a Transaction event with details about the execution of the buyback.

## Intention events always come before the Transaction event.

# Fetch all transactions and intentions

intentions = data.Get(SmartInsiderIntention)

transactions = data.Get(SmartInsiderTransaction)

# Iterate over transactions and parse information

for intention in intentions.Values:

## Generate Insights!

# Skipping magnitude, confidence and source model and assigning 25% to weighting.

insight = Insight.Price(intention.Symbol.Underlying, timedelta(days=5), InsightDirection.Up, None, None,

None, 0.25)

insights.append(insight)

# Iterate over transactions and parse information

for transaction in transactions.Values:

## Generate Insights!

# Skipping magnitude, confidence and source model and assigning 25% to weighting.

if transaction.VolumePercentage != None and transaction.VolumePercentage > 5:

insight = Insight.Price(transaction.Symbol.Underlying, timedelta(days=5), InsightDirection.Down, None,

None, None, 0.25)

insights.append(insight)

return insights

def OnSecuritiesChanged(self, algorithm, changes):

## Add SmartInsider Transaction and Intention data for each new equity

for security in changes.AddedSecurities:

if security.Type == SecurityType.Equity:

transaction = algorithm.AddData(SmartInsiderTransaction, security.Symbol).Symbol

intention = algorithm.AddData(SmartInsiderIntention, security.Symbol).Symbol

self.altDataSymbols[security.Symbol] = (transaction, intention)

## Remove SmartInsider Transaction and Intention data for each new equity

for security in changes.RemovedSecurities:

if security.Type == SecurityType.Equity:

algorithm.Liquidate(security.Symbol)

transaction, intention = self.altDataSymbols.pop(security.Symbol, (None, None))

algorithm.RemoveSecurity(transaction) if transaction is not None else None

algorithm.RemoveSecurity(intention) if transaction is not None else None