Event-Driven Stock Prediction

By:

Xu Gao,

Scott Edenbaum

Table Of Contents

Overview

- Research
  - Reference Paper
  - Project topic
- Data
  - Web scraping
  - Data Cleaning
  - Database storage
- Model development
- Prediction and trading results
- Conclusions and insights
- Q & A

Preliminary Research

Reference Papers

- Deep Learning for Event-Driven Stock Prediction
- *Xiao Ding, Yue Zhang, Ting Liu, Junwen Duan*
- Built a neural tensor network to transform word embeddings to event embeddings to be used as the input for stock prediction neural network
- Deep Learning in Finance
- *J. B. Heaton , N. G. Polson, J. H. Witte*
- Introduced deep learning techniques used in finance
  - Eg: Shallow Factor Models, Default Probabilities, Event Studies

Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Deep Learning for Event-Driven Stock Prediction

Xiao Ding[†*], Yue Zhang[‡], Ting Liu[†], Junwen Duan[†]
[†]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
{xding, tliu, jwduan}@ir.hit.edu.cn
[‡]Singapore University of Technology and Design
yue_zhang@sutd.edu.sg

Abstract

We propose a deep learning method for event-driven stock market prediction. First, events are extracted from news text, and represented as dense vectors, trained using a novel neural tensor network. Second, a deep convolutional neural network is used to model both short-term and long-term influences of events on stock price movements. Experimental results show that our model can achieve nearly 6% improvements on S&P 500 index prediction and individual stock prediction, respectively, compared to state-of-the-art baseline methods. In addition, market simulation results show that our system is more capable of making profits than previously reported systems trained on S&P 500 stock historical data.

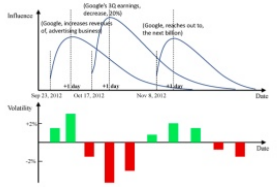


Figure 1: Example news influence of *Google Inc.*

1 Introduction

It has been shown that the financial market is "informationally efficient" [Fama, 1965] — stock prices reflect all known information, and the price movement is in response to news or events. As web information grows, recent work has applied Natural Language Processing (NLP) techniques to explore financial news for predicting market volatility.

Pioneering work mainly uses simple features from news documents, such as bags-of-words, noun phrases, and named entities [Kogan *et al.*, 2009; Schumaker and Chen, 2009]. Although useful, these features do not capture structured relations, which limits their potentials. For example, representing the event "Microsoft sues Barnes & Noble." using term-level features {"Microsoft", "sues", "Barnes", "Noble"} alone, it can be difficult to accurately predict the price movements of *Microsoft Inc.* and *Barnes & Noble Inc.*, respectively, as the unstructured terms cannot differentiate the accuser ("Microsoft") and defendant ("Barnes & Noble").

Recent advances in computing power and NLP technology enables more accurate models of events with structures. Using open information extraction (Open IE) to obtain structured events representations, we find that the actor and object of events can be better captured [Ding *et al.*, 2014]. For example, a structured representation of the event above can be (Actor = *Microsoft*, Action = *sues*, Object = *Barnes & Noble*). They report improvements on stock market prediction using their structured representation instead of words as features.

One disadvantage of structured representations of events is that they lead to increased sparsity, which potentially limits the predictive power. We propose to address this issue by representing structured events using *event embeddings*, which are dense vectors. Embeddings are trained such that similar events, such as (Actor = *Nvidia fourth quarter results*, Action = *miss*, Object = *views*) and (Actor = *Delta profit*, Action = *didn't reach*, Object = *estimates*), have similar vectors, even if they do not share common words. In theory, embeddings are appropriate for achieving good results with a density estimator (e.g. convolutional neural network), which can misbehave in high dimensions [Bengio *et al.*, 2005]. We train event embeddings using a novel neural tensor network (NTN), which can learn the semantic compositionality over event arguments by combining them multiplicatively instead of only implicitly, as with standard neural networks.

For the predictive model, we propose to use deep learning [Bengio, 2009] to capture the influence of news events over a history that is longer than a day. Research shows diminishing effects of reported events on stock market volatility. For example, Xie et al. [2013], Tetlock et al. [2008] and Ding et al. [2014] show that the performance of daily prediction is better than weekly and monthly prediction. As shown in Figure 1, the influences of three actual events for Google

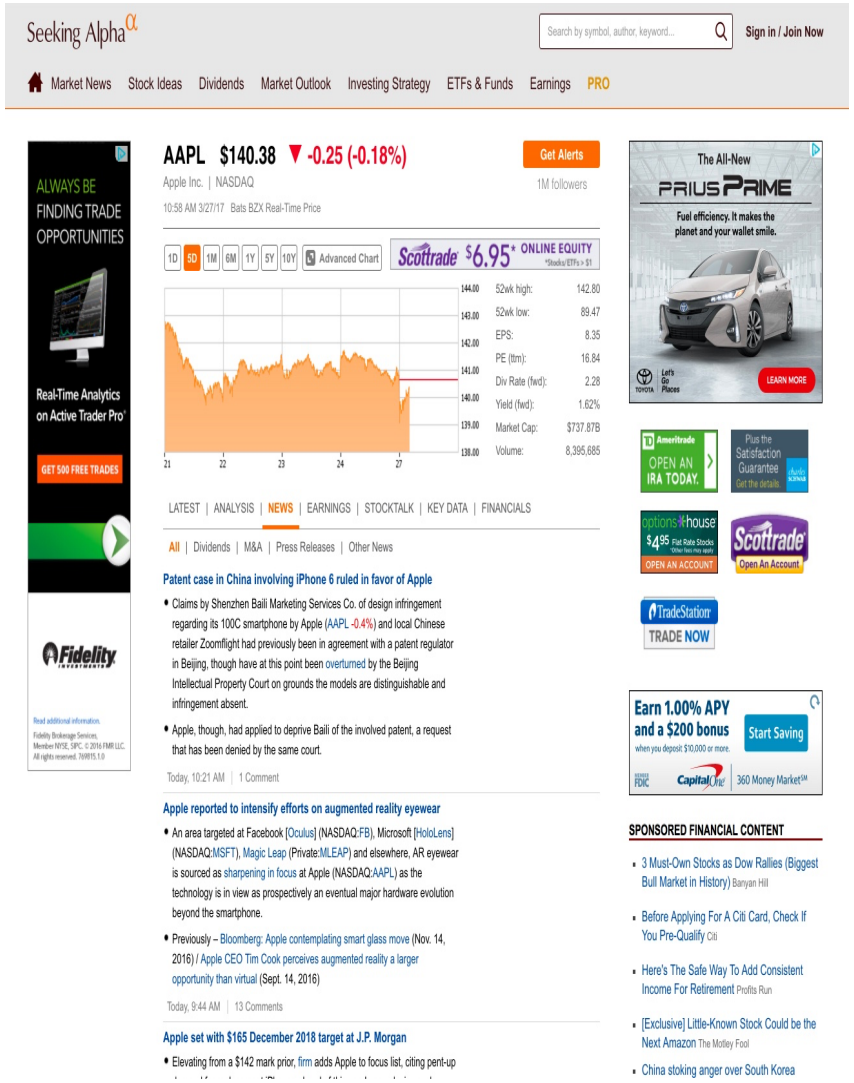
[*]This work was done while the first author was visiting Singapore University of Technology and Design

Project Topic

- **Event-driven stock prediction**
  - What is it?
    - How can we use it?
- Project Motivation
  - Background in finance
  - Interest in financial modeling
  - Lack of research
  - NLP analysis

Data Sources

- Data Sources
  - News events
    - Seeking Alpha
      - Selenium
      - BeautifulSoup
  - Pricing data
    - Bloomberg terminal

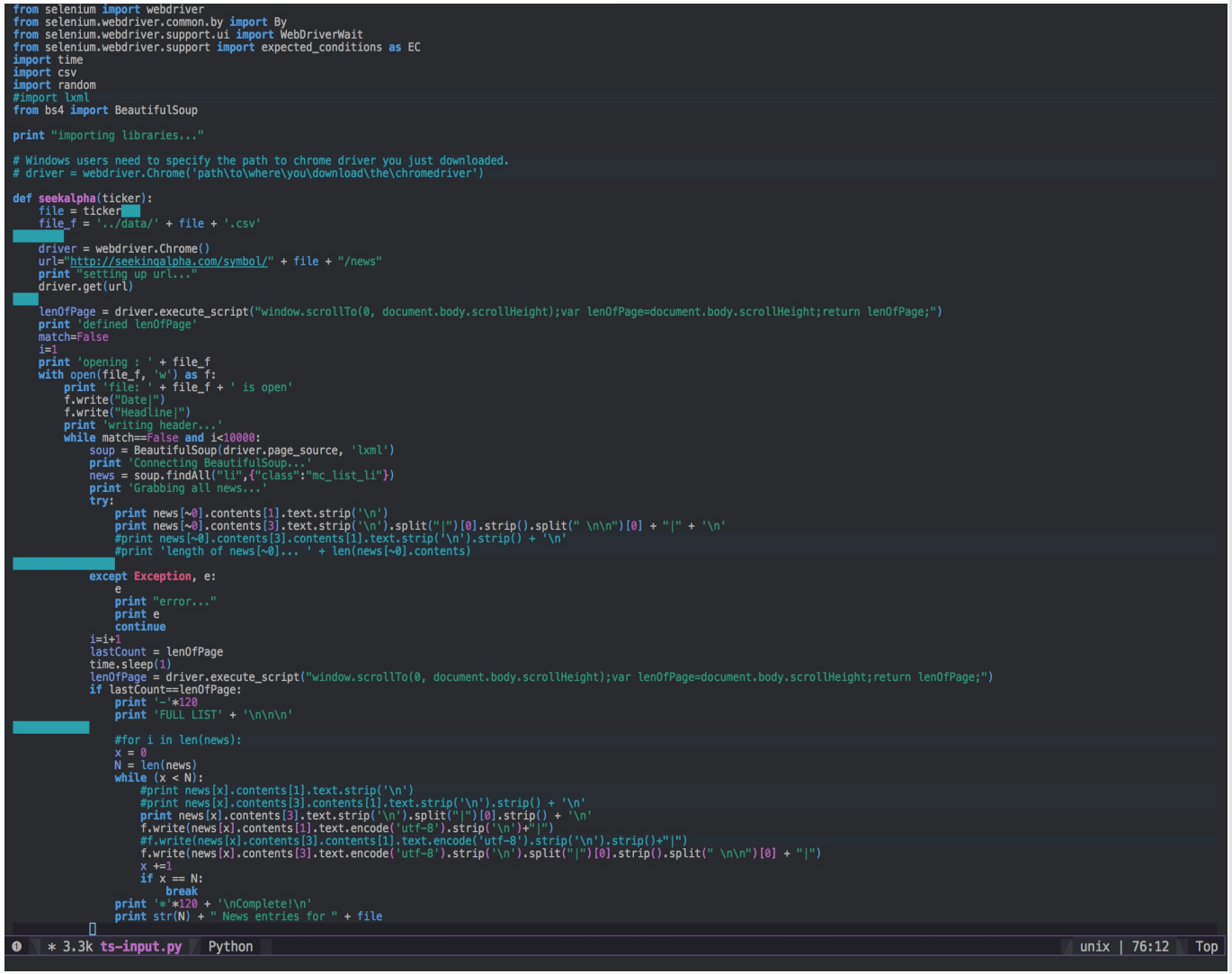


Data hurdles

- Finding news source
  - Content relevance
  - Content consistency
  - Historical data
- Bot & web scraping detection
- Inconsistent structure of news content
- Seemingly random XPath values

Data Scraping

- Selenium
- BeautifulSoup



```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
import csv
import random
#import lxml
from bs4 import BeautifulSoup

print "importing libraries..."

# Windows users need to specify the path to chrome driver you just downloaded.
# driver = webdriver.Chrome('path\to\where\you\download\the\chromedriver')

def seekalpha(ticker):
    file = ticker
    file_f = '../data/' + file + '.csv'

    driver = webdriver.Chrome()
    url="http://seekingalpha.com/symbol/" + file + "/news"
    print "setting up url..."
    driver.get(url)

    lenOfPage = driver.execute_script("window.scrollTo(0, document.body.scrollHeight);var lenOfPage=document.body.scrollHeight;return lenOfPage;")
    print 'defined lenOfPage'
    match=False
    i=1
    print 'opening : ' + file_f
    with open(file_f, 'w') as f:
        print 'file: ' + file_f + ' is open'
        f.write("Date|")
        f.write("Headline|")
        print 'writing header...'
        while match==False and i<10000:
            soup = BeautifulSoup(driver.page_source, 'lxml')
            print 'Connecting BeautifulSoup...'
            news = soup.findAll("li",{"class":"mc_list_li"})
            print 'Grabbing all news...'
            try:
                print news[~0].contents[1].text.strip('\n')
                print news[~0].contents[3].text.strip('\n').split("|")[0].strip().split(" \n\n")[0] + "|" + '\n'
                #print news[~0].contents[3].contents[1].text.strip('\n').strip() + '\n'
                #print 'length of news[~0]... ' + len(news[~0].contents)

            except Exception, e:
                e
                print "error..."
                print e
                continue
            i=i+1
            lastCount = lenOfPage
            time.sleep(1)
            lenOfPage = driver.execute_script("window.scrollTo(0, document.body.scrollHeight);var lenOfPage=document.body.scrollHeight;return lenOfPage;")
            if lastCount==lenOfPage:
                print '-'*120
                print 'FULL LIST' + '\n\n\n'

                #for i in len(news):
                x = 0
                N = len(news)
                while (x < N):
                    #print news[x].contents[1].text.strip('\n')
                    #print news[x].contents[3].contents[1].text.strip('\n').strip() + '\n'
                    print news[x].contents[3].text.strip('\n').split("|")[0].strip() + '\n'
                    f.write(news[x].contents[1].text.encode('utf-8').strip('\n')+"|")
                    #f.write(news[x].contents[3].contents[1].text.encode('utf-8').strip('\n').strip()+"|")
                    f.write(news[x].contents[3].text.encode('utf-8').strip('\n').split("|")[0].strip().split(" \n\n")[0] + "|")
                    x +=1
                    if x == N:
                        break
                print '*'*120 + '\nComplete!\n'
                print str(N) + " News entries for " + file
```

Data Cleaning

- Extract from data
  - Headline
  - Content
  - Date & time

```
Date|Headline|Mon, Mar. 13, 10:24 AM|Global miners move higher as China affirms capacity cuts

China's steel and iron ore futures resume their climb overnight, surging a respective 5.9% and 4.3%, after the head of China’s Ministry of Information and Technology reaffirmed the government’s resolve to cut excess capacity in the steel sector. Today's strength also is attributed to seasonal demand, as Chinese construction activity picks up after a slowdown over winter. Among relevant tickers: SLX +2.4%, BHP +2.1%, RIO +3.5%, VALE +4.1%, CLF +4%, X +3.9%, MT +4.2%, AKS +3%, WOR +2.7%, FCX +2.3%, TECK +2.6%, AA +2.4%, CENX +1.7%, ACH +1.2%, KALU +1%. |Mon, Mar. 6, 8:53 AM|Alcoa upgraded to Buy at Goldman Sachs

Alcoa (NYSE:AA) +1.5% premarket as Goldman Sachs upgrades shares to Buy from Neutral and raises its six-month  price target $52 from $34, TheFly.com reports.Goldman says China's capacity cuts are improving  industry dynamics, and the firm expects  AA to generate a 12% free cash yield on average through 2018 amid  higher aluminum and alumina prices.|Mon, Mar. 6, 8:14 AM|China proposes capacity cuts in steel, coal

China says it will cut steel capacity by 50M metric tons and coal output by more than 150M metric tons in 2017, according to a report issued at the weekend's National People's Congress.The expected cuts come as the government also reduces its annual economic growth target to ~6.5% vs. last year's 6.5%-7% range. The government also will target cuts in energy consumption per unit of gross domestic product by 3.4% and in carbon intensity by 4% this year.No mention was made of aluminum cuts.Potentially relevant tickers include: SLX, X, MT, AKS, NUE, STLD, BHP, RIO, AA, CENX, ACH |Thu, Mar. 2, 10:06 AM|Alcoa combines business units, names Reyes to run aluminum business
```

Data Pre-Processing

ReVerb

- Java tool for text processing
  - identifies "similar" words
- Identifies & extracts binary relationships in English sentences
- Designed for use with "Web-scale" information extraction
- Eg: "is" -> "be", "was" -> "be"
  - Conceptually similar to nltk "stemming"
- Each news item/event is a separate input for ReVerb

ReVerb Results

ReVerb Sample Output

- *1 China 's steel and iron ore futures resumetheir climb 0 7 7 8 8 10 0.8225299076805256 China 's steel and iron ore futures resume their climb overnight , surging a respective 5.9 % and 4.3 % , after the head of China's Ministry of Information and Technology reaffirmed the government's resolve to cut excess capacity in the steel sector . NNP POS NN CC NN NN NNS VBP PRP$ NN RB , VBG DT JJ CD NN CC CD NN , IN DT NN IN NNP NNP NNP IN NNP CC NNP VBD DT NN VBD NN TO VB JJ NN IN DT NN NN . B-NP I-NP I-NP I-NP I-NP I-NP I-NP B-VP B-NP I-NP B-ADVP O B-VP B-NP I-NP I-NP I-NP O B-NP I-NP O B-PP B-NP I-NP I-NP I-NP I-NP I-NP I-NP I-NP I-NP I-NP B-VP B-NP I-NP B-VP B-NP B-VP I-VP B-NP I-NP B-PP B-NP I-NP I-NP O china 's steel and iron ore futures resume their*
- Not human readable

ReVerb Code

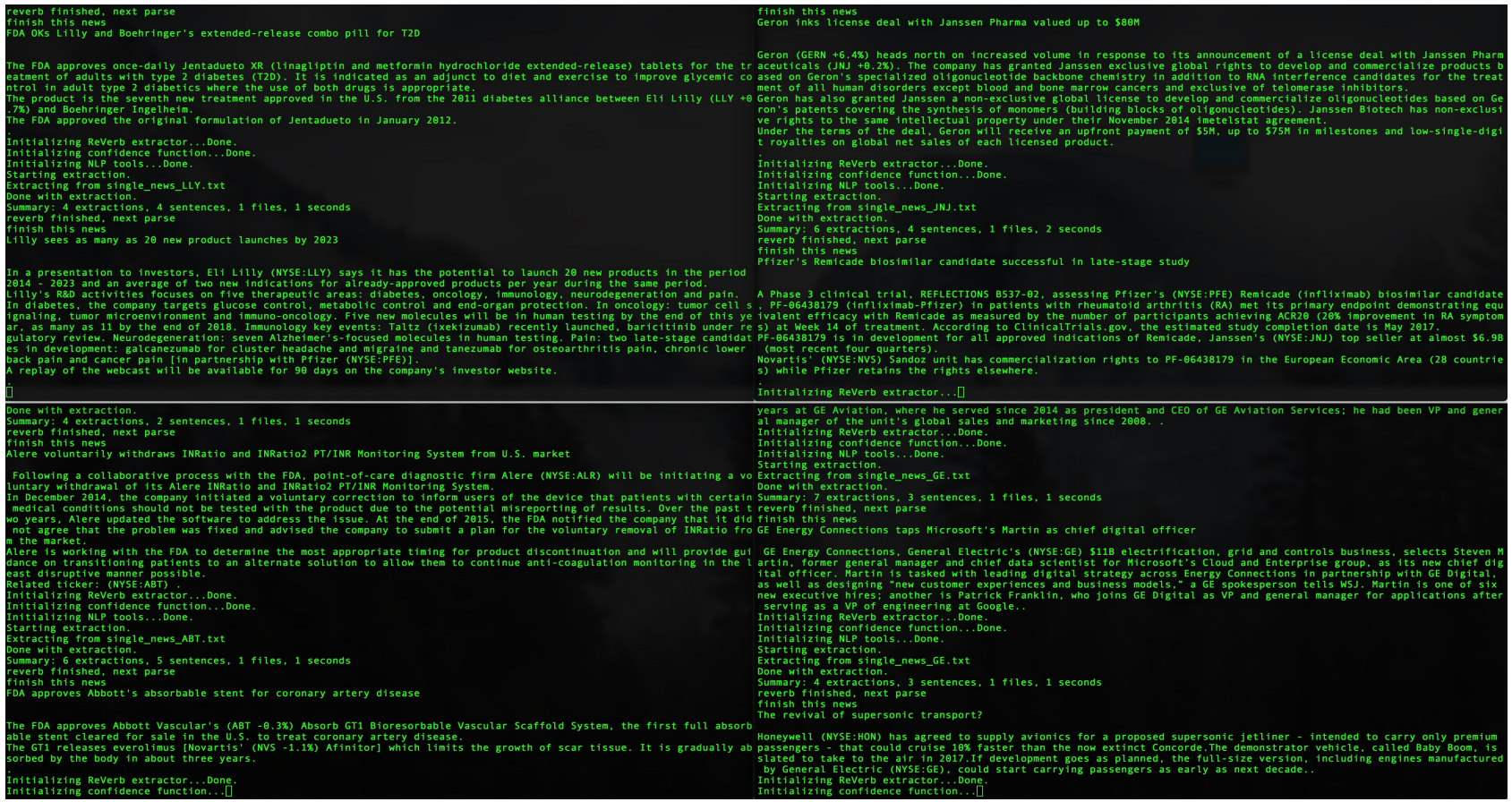
Reverb is slow

- Tweaked code for concurrent data processing

```
from sentiment import parse_1

def reverb():
    ticker, df=parse_1()
    news=df.content
    confidence=[]
    subject=[]
    object_=[]
    verb=[]

    single_news = "single_news_" + ticker + ".txt"
    ticker_output = "output_" + ticker + ".txt"
    for each_news in news:
        news=each_news+'.'
        print news
        f=open(single_news,'w')
        f.write(news)
        f.close()
        javaosarg = "java -Xmx512m -jar reverb-latest.jar " + single_news + " > " + ticker_output
        os.system(javaosarg)
        output=open(ticker_output,'r')
        info=output.read()
        output.close()
        print "reverb finished, next parse"
        confid,subj,obje,verb_=reverb_parse(info)
        confidence.append(confid)
        subject.append(subj)
        object_.append(obje)
        verb.append(verb_)

    osrmarg1 =  "rm " + single_news
    osrmarg2 = "rm " + ticker_output
    os.system(osrmarg1)
    os.system(osrmarg2)
    df['confidence']=confidence
    df['subject']=subject
    df['object']=object_
    df['verb']=verb
    file_='../data/reverb/'+ticker+'reverb.csv'
    df.to_csv(file_)

def reverb_parse(info):
    reverb_list=info.split('\n')
    confi=[]
    sub=[]
    obj=[]
    verb=[]
    if len(reverb_list)==0:
        confidence=0
        subject=''
        verb_=''
        object_=''
    else:
        for i in range(len(reverb_list)-1):
            buf=reverb_list[i].split('\t')
            confi.append(float(buf[11]))
            sub.append(buf[-3])
            verb.append(buf[-2])
            obj.append(buf[-1])
        if len(confi)==0:
            confidence=0
            subject=''
            verb_=''
            object_=''
        else:
            idx=np.argmax(confi)
            confidence=confi[idx]
            subject=sub[idx]
            object_=obj[idx]
            verb_=verb[idx]
            print "finish this news"
    return confidence,subject,object_,verb_
```

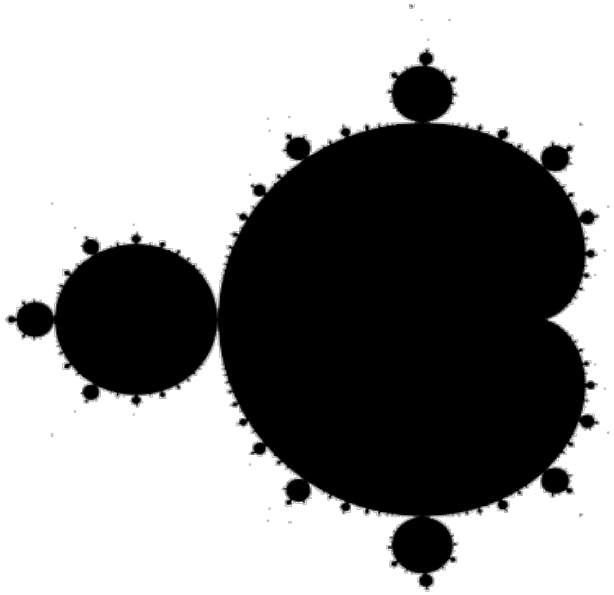
ReVerb

- Processing the data can be quite time consuming
  - Solution – concurrent instances to speed up data processing



Data Transform

- News and text data have many possible transformations to assist in modeling/analysis
  - Sentiment Analysis with TextBlob
  - Word Embedding with Word2Vec
  - Document Embedding with Doc2Vec
  - Event Embedding



TextBlob

gensim

topic modelling for humans

word2vec

Tool for computing continuous distributed representations of words

Data Storage

Stock Pricing Data

- MySQL DB

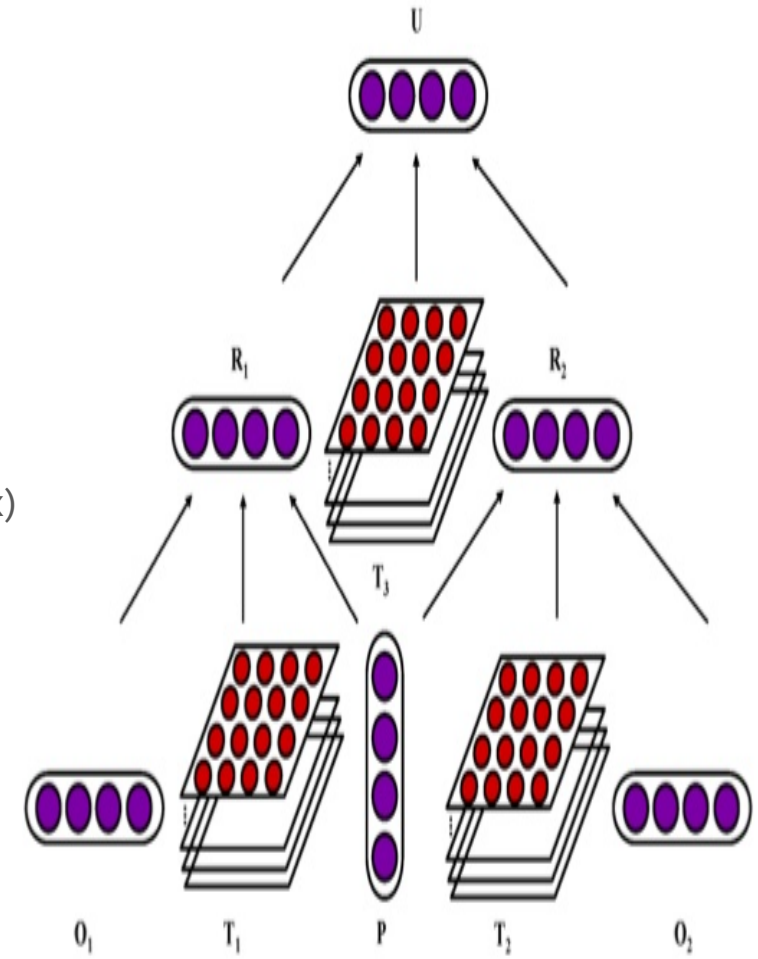
```
mysql> describe common_data;
+---------+---------------+------+-----+---------+-------+
| Field   | Type          | Null | Key | Default | Extra |
+---------+---------------+------+-----+---------+-------+
| SPX     | decimal(19,10) | YES  |     | NULL    |       |
| NYSEVOL | decimal(19,10) | YES  |     | NULL    |       |
| CCMP    | decimal(19,10) | YES  |     | NULL    |       |
| date    | date          | NO   |     | NULL    |       |
+---------+---------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> describe stock_data;
+--------+--------------+------+-----+---------+-------+
| Field  | Type         | Null | Key | Default | Extra |
+--------+--------------+------+-----+---------+-------+
| ticker | varchar(5)   | NO   |     | NULL    |       |
| close  | decimal(19,4) | YES  |     | NULL    |       |
| date   | date         | NO   |     | NULL    |       |
+--------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```python
#######################################################################
# update the db_host, db_user, db_name, db_pass values
# db_get_all only grabs data for stocks included in ../data/Price
# db_get_all outputs a dictionary of dataframes named by the stock ticker (capital)
# eg:
# stocks = db_get_all()
# stocks['AA'] -> dataframe for AA
# stocks['AA']['close'][0] -> 81.2958
# stocks['AA']['date'][0] -> datetime.date(2008, 2, 21)


import MySQLdb as mdb
import pandas as pd
import os
import datetime as dt
from pandas.io import sql


def db_get_ticker(ticker):
    t = ticker
    cols = ['ticker', 'closer', 'date']

    db_host = '216.230.228.88:3306'
    db_user = 'bc8_scottede'#scottdb'
    db_name = 'securities'
    db_pass = '######'
    #con = mdb.connect(host = db_host, user = db_user, passwd = db_pass, db=db_name)
    con = mdb.connect(host = db_host, user = db_user, db=db_name)
    query = ("SELECT * FROM stock_data WHERE ticker = '%s'" % (t))
    df = sql.read_sql(query, con=con)
    print "grabbing stock data for %s" % (t)
    return (df)

def db_get_all():
    stocks = []
    pricedir = '../data/Price/' #directory
    for file_ in set(os.listdir(pricedir)):
        try:
            g = len(file_)
            if g < 10:
                stocks.append(str(file_).split(".")[0])
        except Exception, e:
            print e
            continue
    print "getting all stock data: "
    [stock for stock in stocks]
    stock_data = {stock: db_get_ticker(stock) for stock in stocks}

    return (stock_data)
~
~
1  * 1.5k dbgrab.py  Python                                  unix | 9:55  All
```

Model Development

Foundational model

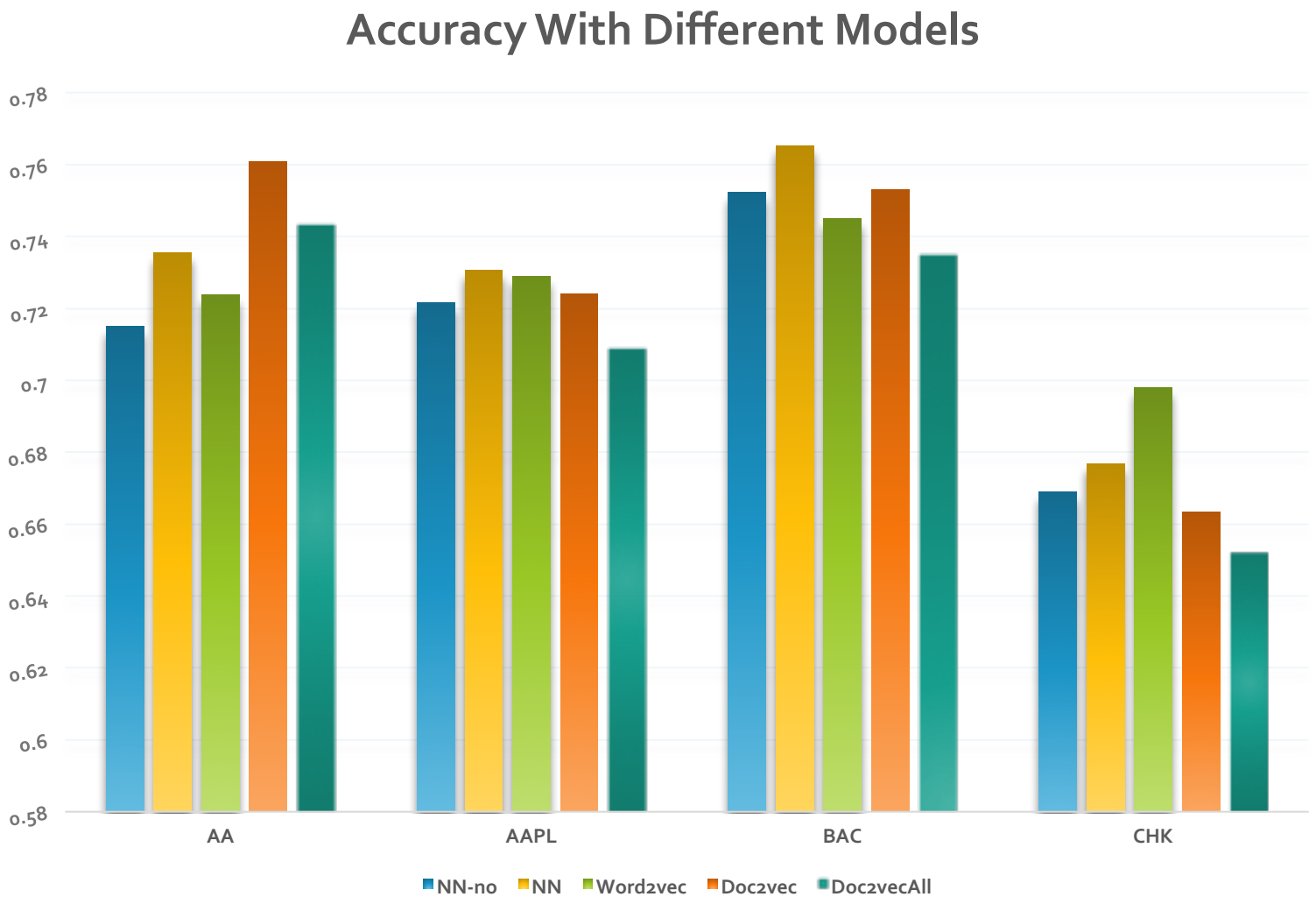
- Neural Network
  - 2 hidden layers
    1. 300 nodes, activation function: tanh(x)
    2. 50 nodes, activation function: tanh(x)
  - Output layer
    - Single node
      - Binary result with Sigmoid activation function



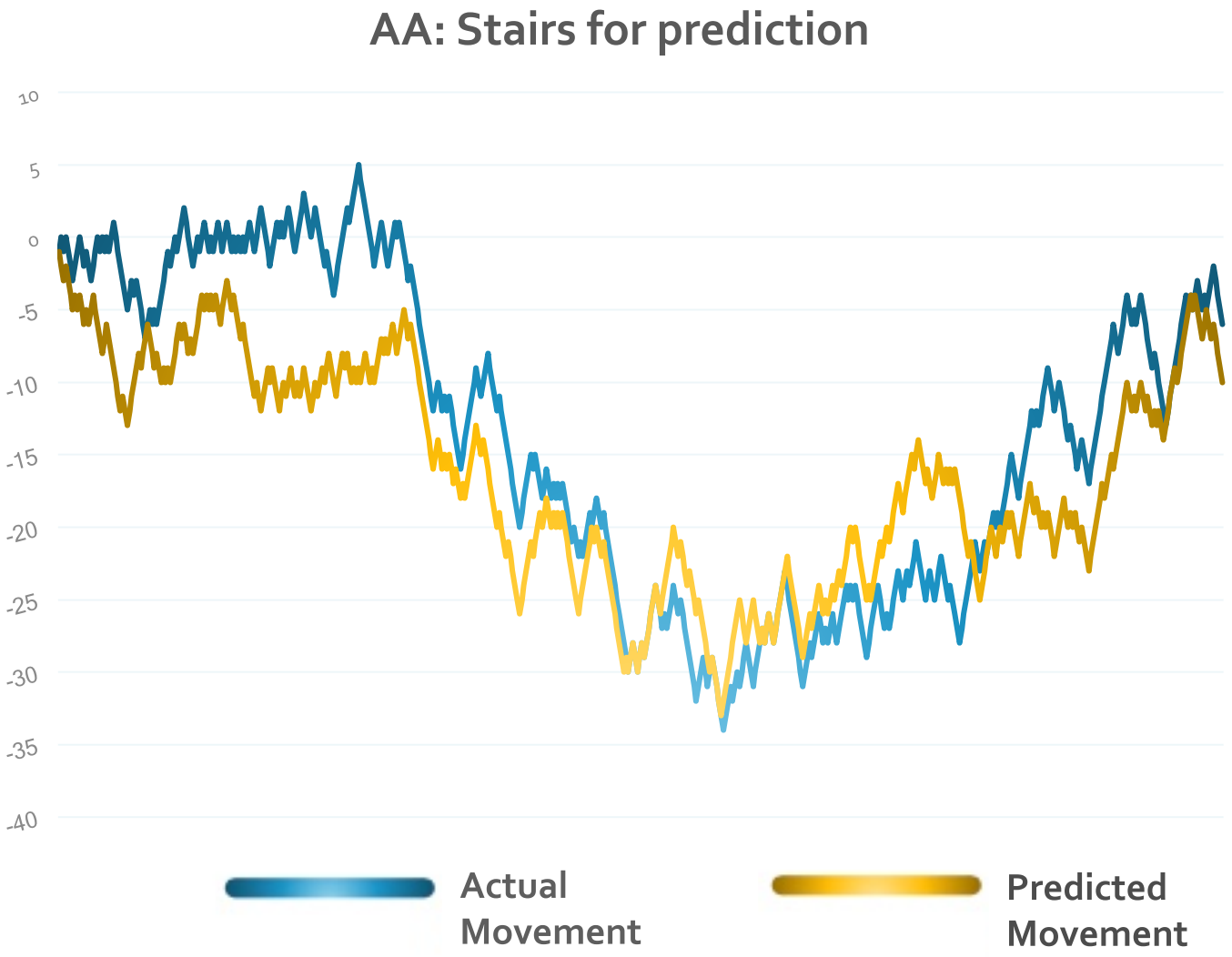
Dataset Development

1. Market Data
   - 5 day window
     - S&P 500, NASDAQ Composite, NYSE Volume
2. Sentiment polarity
   - Generated with TextBlob
3. Word Embeddings
   - Uses output from ReVerb
   - Grabs sentence with highest confidence level
   - Subject, object, and verb 'tuple' forms word embeddings
   - Downsides: - loss of information through selection process
4. Doc2vec
   - Converts paragraph to single vector
   - Use vector as input for Neural Network
   - Generate training dataset for each stock and combines
   - Downsides: - uses all info, does not filter out irrelevant text

Results


Accuracy With Different Models
0.78
0.76
0.74
0.72
0.7
0.68
0.66
0.64
0.62
0.6
0.58
AA
AAPL
BAC
CHK
NN-no
NN
Word2vec
Doc2vec
Doc2vecAll

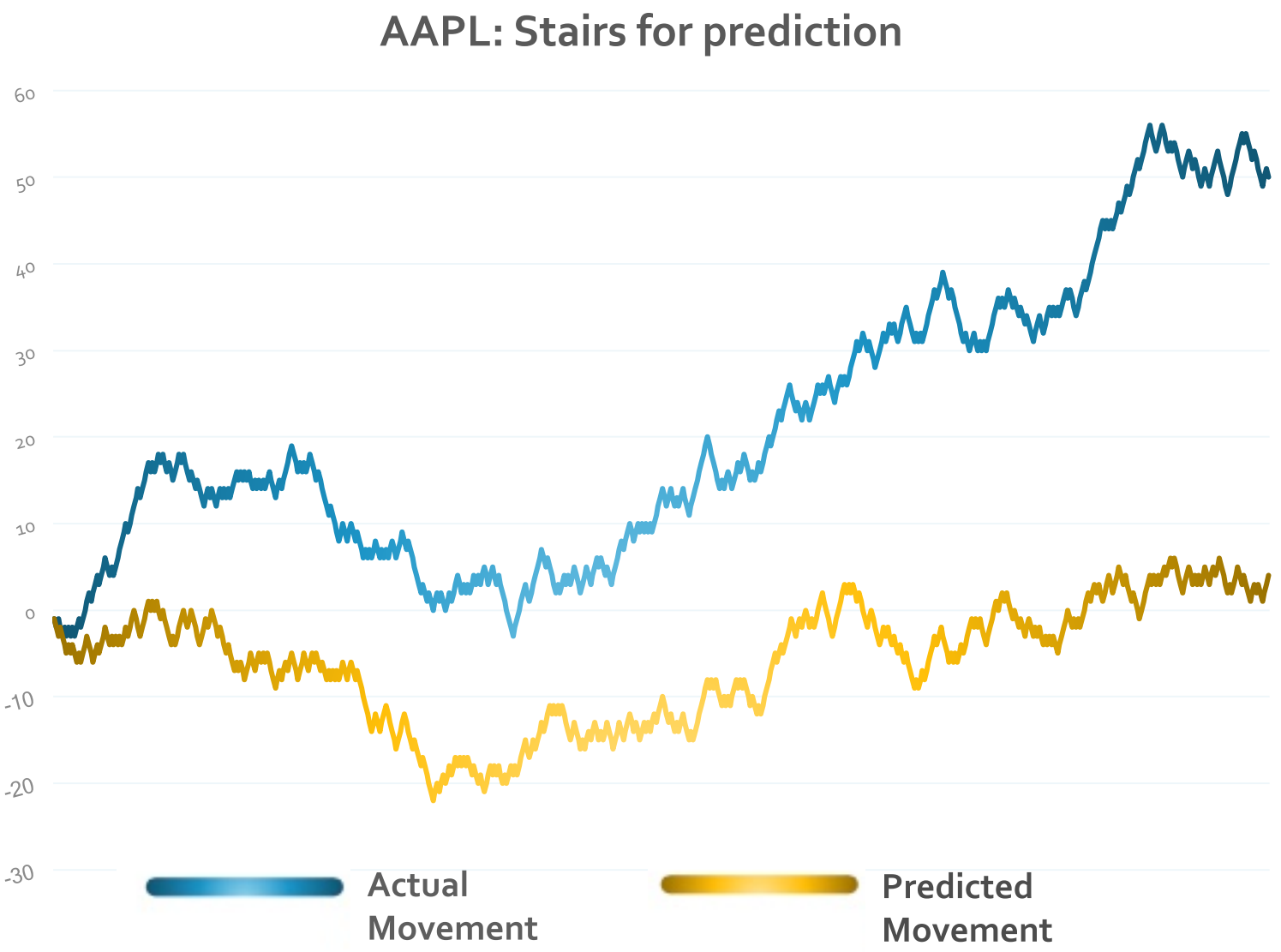
Results: AA


AA: Stairs for prediction
10
5
0
-5
-10
-15
-20
-25
-30
-35
-40
Actual Movement
Predicted Movement

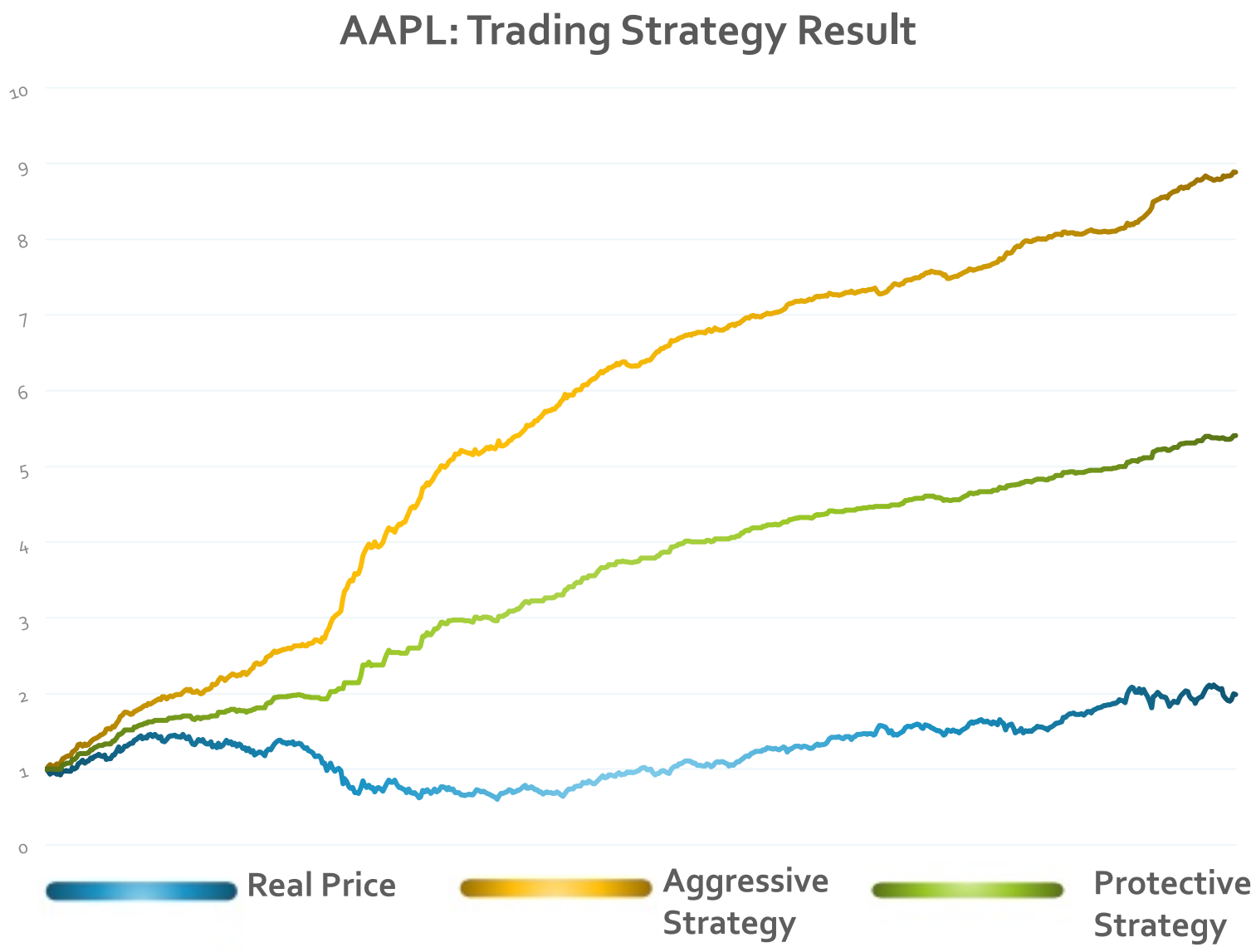
Results: AA


Alcoa: Trading Strategy Results
2.5
2
1.5
1
0.5
0
Real Price
Aggressive Strategy
Protective Strategy

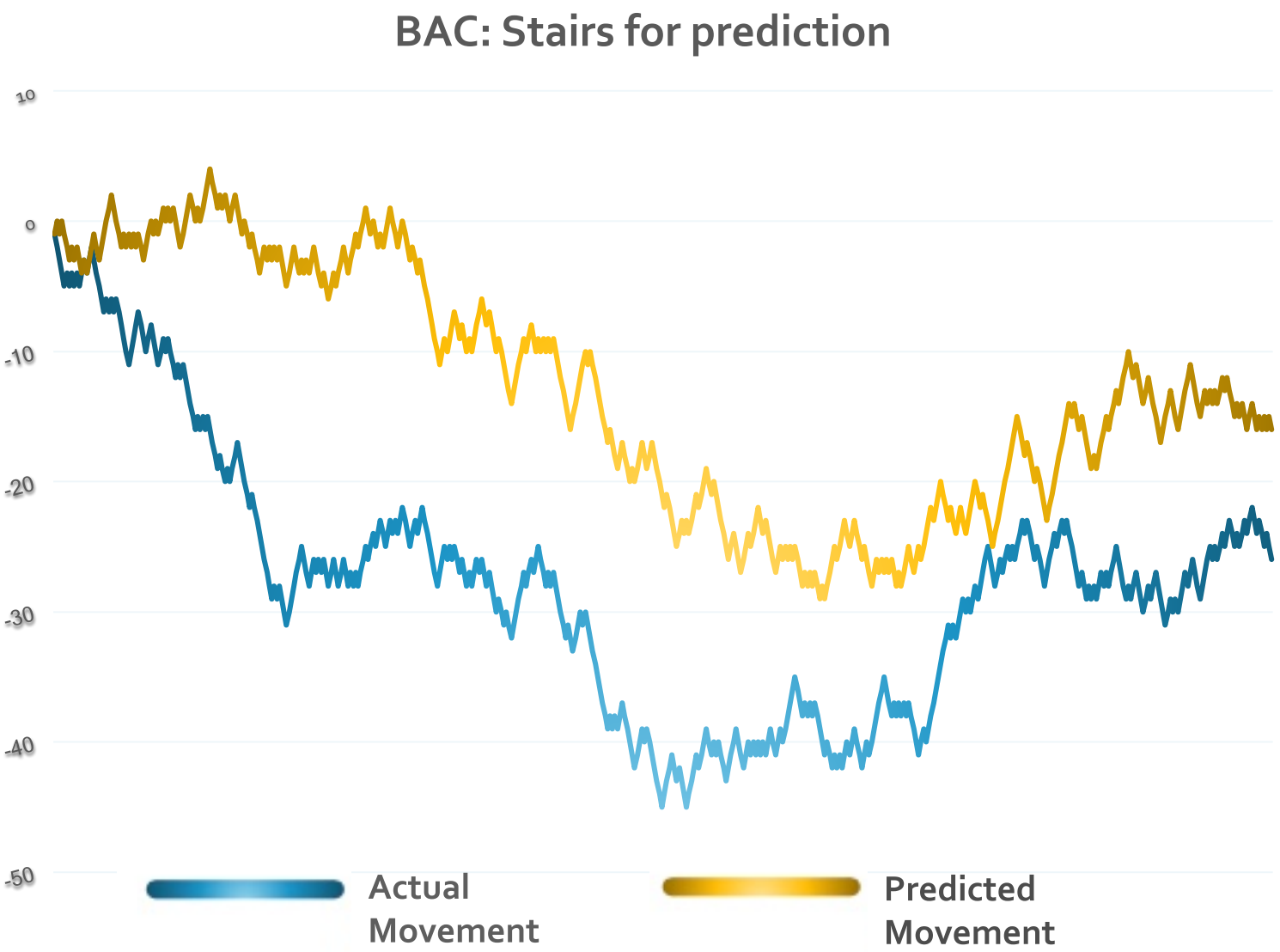
Results: AAPL



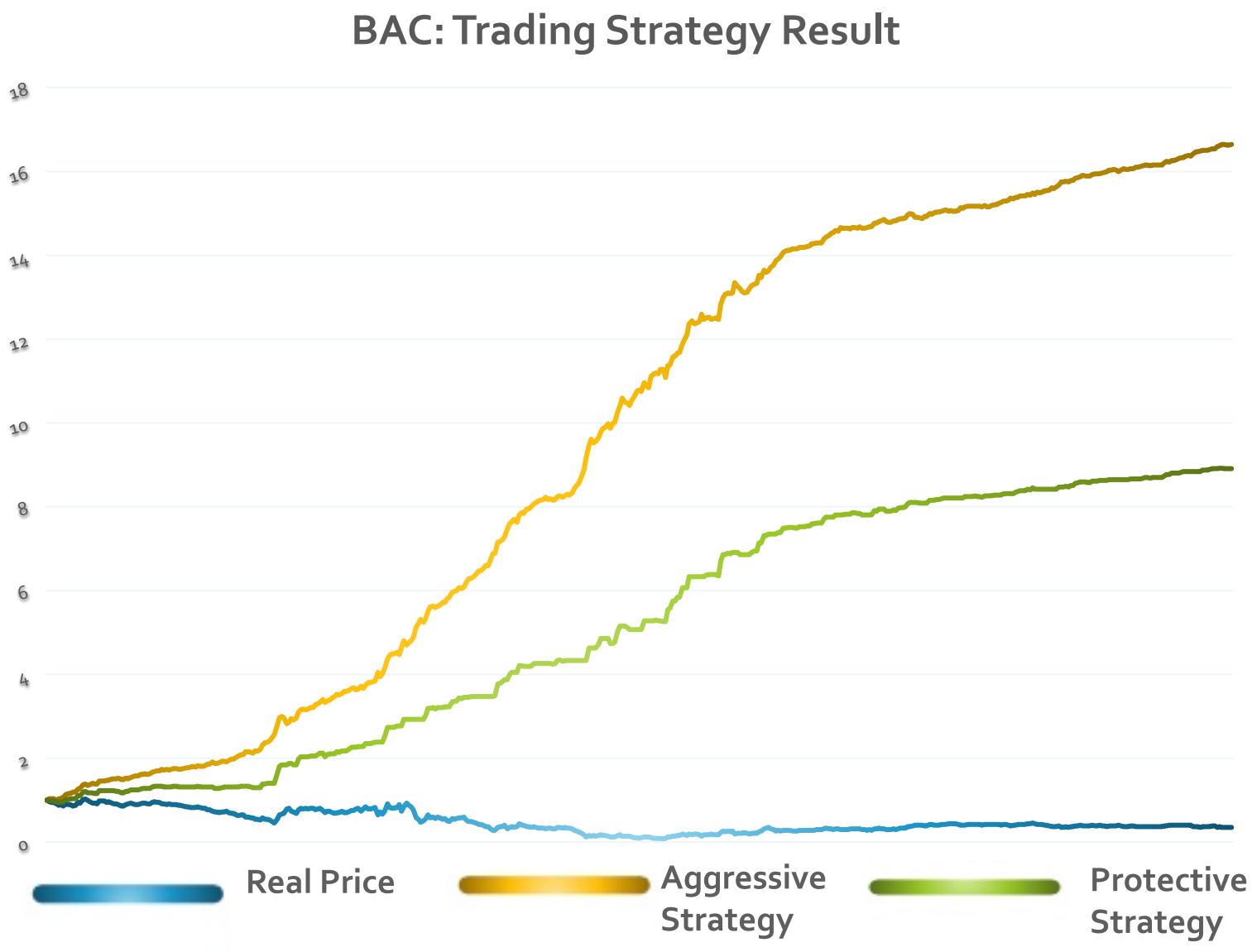
Results: AAPL



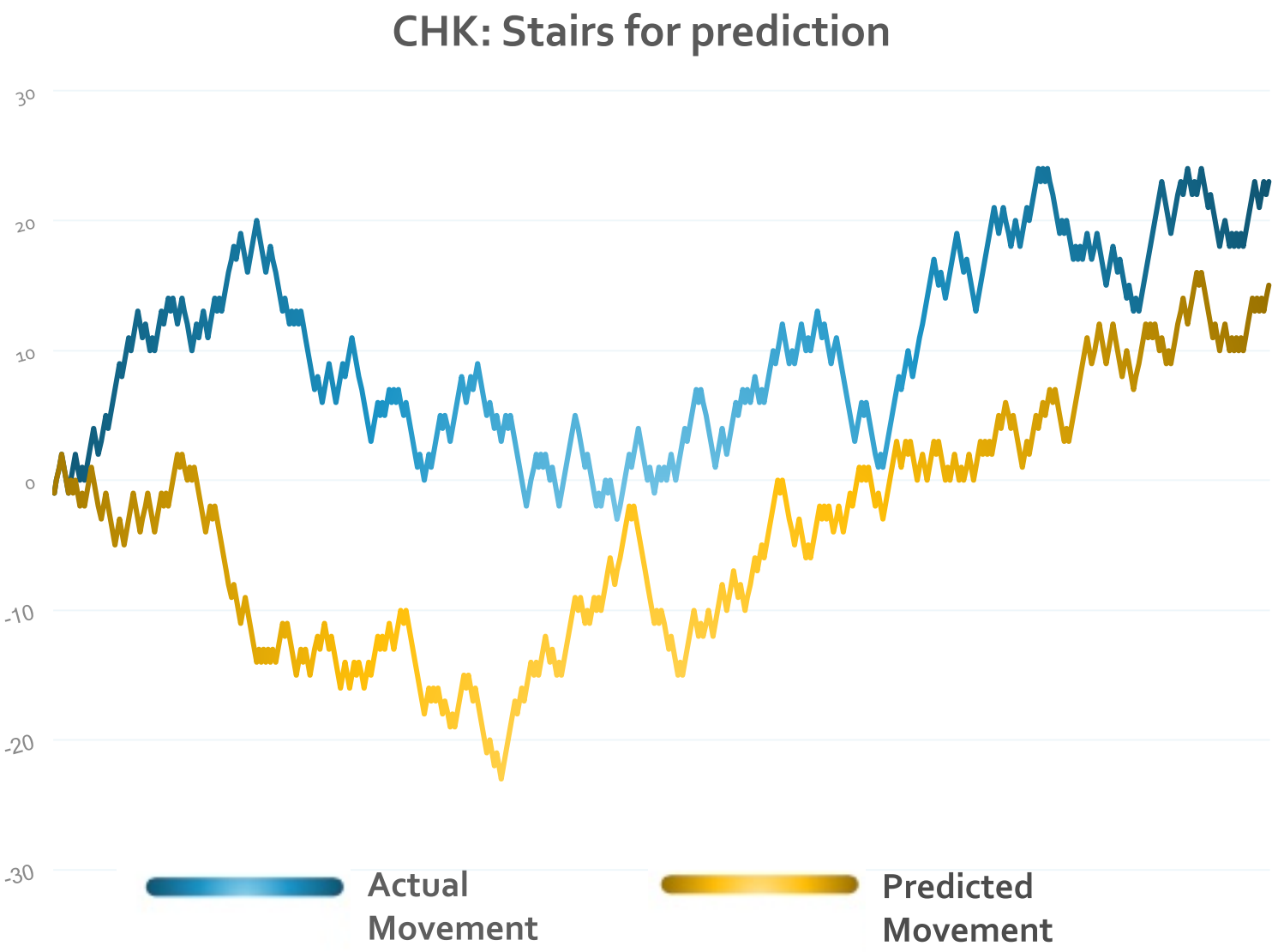
Results: BAC


BAC: Stairs for prediction
10
0
-10
-20
-30
-40
-50
Actual Movement
Predicted Movement

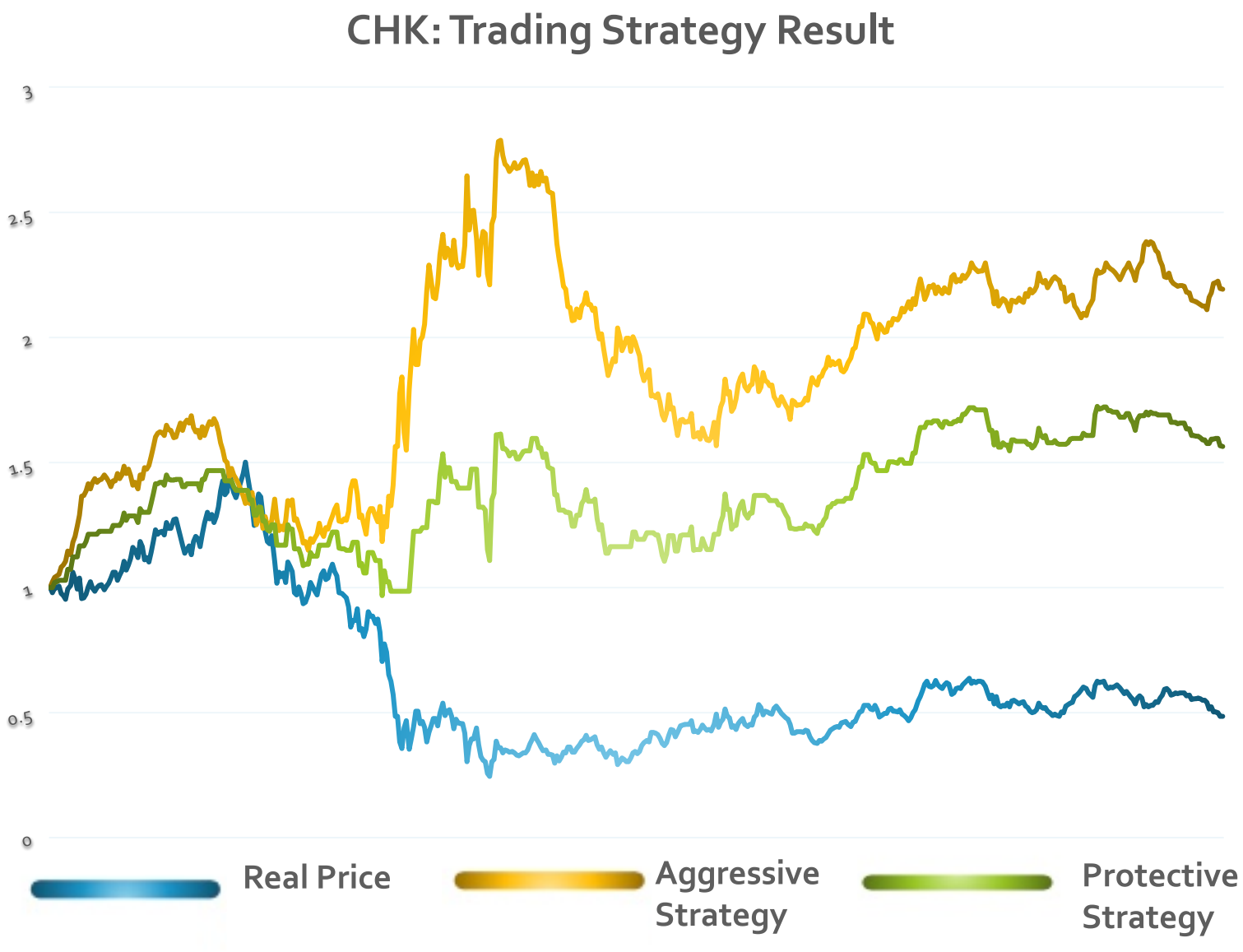
Results: BAC


BAC: Trading Strategy Result
18
16
14
12
10
8
6
4
2
0
Real Price
Aggressive Strategy
Protective Strategy

Results: CHK


CHK: Stairs for prediction
30
20
10
0
-10
-20
-30
Actual Movement
Predicted Movement

Results: CHK


CHK: Trading Strategy Result
3
2.5
2
1.5
1
0.5
0
Real Price
Aggressive Strategy
Protective Strategy

Conclusions and Insights

- Accurate historical stock news is hard to acquire!
- Each 'model' has strengths & weaknesses
  - Model accuracy is for relevance of text content to sentence subject
    - Does not imply impact of news event
- Our pricing model naively assumes stocks open at previous close value
- Model only predicts direction of price movement, not magnitude
- This is a new approach to financial modeling is bound to grow in popularity as tools for text processing grow and historical data become more available
- A next step would be to categorize events
  - Ie: including a corporate events overlay to better identify scheduled corporate actions

Q&A

Growing Popularity of Event-Driven Strategies

- 19% increase in net demand for Event-Driven strategies, according to 2014 Credit Suisse survey
- Event-Driven strategies provide an alternative investment strategy with low correlations to economic and market trends