TRADING ACTION CLASSIFICATION

By: Quang Lam, Thu Pham

1. Introduction
2. Data Source
3. Data Overview
4. Model Implementation
5. Report Poster

INTRODUCTION

- Our purpose is to utilize machine learning to predict stock trading action (long position/short position) in the monthly-basis.
- Our model is currently applied to a single stock symbol.

DATA SOURCE

Stock Price

- Alpha Vantage API
- Monthly
- Since 2000

timestamp	open	high	low	close	volume
2019-05-14	130.530	130.650	123.0400	124.73	285095390
2019-04-30	118.950	131.370	118.1000	130.60	433157868
2019-03-29	112.890	120.820	108.8000	117.94	589045341
2019-02-28	103.775	113.240	102.3500	112.03	469095970
2019-01-31	99.550	107.900	97.2000	104.43	714204787
2018-12-31	113.000	113.420	93.9600	101.57	944287635
2018-11-30	107.050	112.240	99.3528	110.89	720228643
2018-10-31	114.750	116.180	100.1100	106.81	927547942
2018-09-28	110.850	115.290	107.2300	114.37	480255674
2018-08-31	106.030	112.777	104.8400	112.33	456630721

DATA SOURCE

Convert the dataset to classification data

- Long (1) if the stock price increases by a threshold. Short if otherwise.

```
expectedReturn = 0.025 # Long if the stock price increases by 2.5%

df['prev_close'] = df['close'].shift(-1)

df['action'] = np.nan

for i, row in df.iterrows():
    realReturn = (df.loc[i, 'close'] / df.loc[i, 'prev_close']) - 1
    df.loc[i, 'action'] = 1 if (realReturn >= expectedReturn) else 0 # 1 = Long / 0 = Short
```

DATA SOURCE

Sentimental Analysis

- New York Times API

```
'We Just Waited for Our Moment to Be Killed'
Giving the Globe A Networked Skin
Chief Justice's Annual Report Notes Progress in the Judiciary
Haven't I Seen These Shows Before?
Great Hits Headed for The Attic
Russian Troops Are Edgy as They Prepare to Go 'Clean' Grozny
Works In Progress From All Over; Eliot's Sly Revenge Against a Darwinist
Looking Back, With an Eye to the Future
Nets to Face A Truer Test On the Road
Works In Progress From All Over; Bloodless Knife for Epilepsy
In the City's Bunker, Much Ado About Not Much as the Celebrations Proceed
Stanford Hopes To Follow Path Wisconsin Cut
Get Set to Say Hi To the Neighbors
Paid Notice: Deaths  TAYLOR, FRIEDA
A Glittering Party For Times Square
Paid Notice: Memorials  BARISON, DAVID ANDREW
```

2,172,712

headlines

DATA SOURCE

Sentimental Analysis

- Assign a positive/negative value to each headline.
- Calculate the percentage of positive headline each month.

```
positive_count = 0.0
negative_count = 0.0
total_count = 0.0

for i in range(len(NYTimes_data["response"]["docs"][:])):
    try:
        headline = NYTimes_data["response"]["docs"][:][i]['headline']['main']
        analysis = TextBlob(headline)
        total_count += 1
        # set sentiment
        if analysis.sentiment.polarity > 0:
            positive_count += 1
        elif analysis.sentiment.polarity == 0:
            positive_count += 0.5
            negative_count += 0.5
        else:
            negative_count += 1
    except:
        pass

return positive_count / total_count
```

DATA SOURCE

Sentimental Analysis

- Pick out the headline related to the stock symbol.
- Assign a positive/negative value to each headline.
- Count the number of company appearance on news.
- Calculate the percentage of positive appearance each month.

```
keywords = ['microsoft', 'msft']

positive_count = 0.0
negative_count = 0.0
total_count = 0.0

for i in range(len(NYTimes_data["response"]["docs"][:])):
    try:
        headline = NYTimes_data["response"]["docs"][:][i]['headline']['main']
        analysis = TextBlob(headline)
        # set sentiment
        if stringContainsKeywords(headline, keywords):
            total_count += 1
            if analysis.sentiment.polarity > 0:
                positive_count += 1
            elif analysis.sentiment.polarity == 0:
                positive_count += 0.5
                negative_count += 0.5
            else:
                negative_count += 1
    except:
        pass


result = 0.0
try:
    result = positive_count / total_count
except:
    pass

return (int(total_count), result)
```

DATA SOURCE

timestamp	positive	appearance	positive2
2019-05-14	0.544137	3	0.666667
2019-04-30	0.562695	4	0.500000
2019-03-29	0.556610	2	0.500000
2019-02-28	0.548660	9	0.500000
2019-01-31	0.561755	0	0.000000
2018-12-31	0.552921	2	1.000000
2018-11-30	0.548968	3	0.166667
2018-10-31	0.553506	0	0.000000
2018-09-28	0.565977	8	1.000000
2018-08-31	0.559895	5	0.600000

Data Overview

STOCH

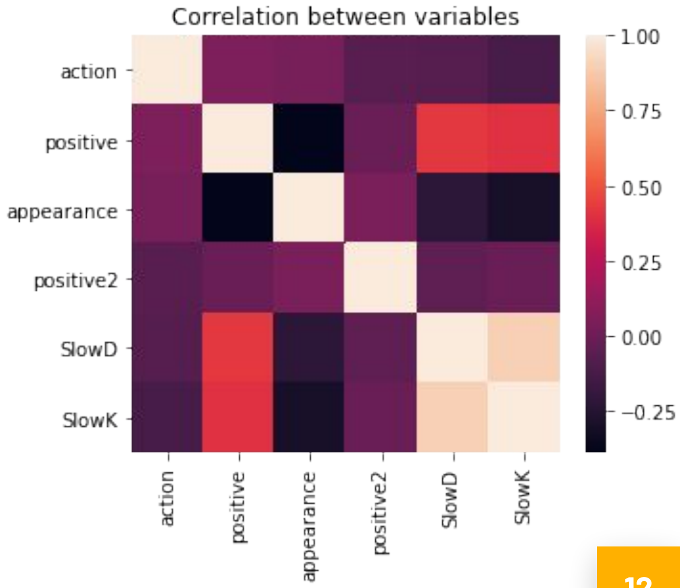
Stochastic oscillator is a momentum indicator comparing a particular closing price of a security to a range of its price over a period of time.

- SlowK
- SlowD

time	SlowD	SlowK
2019-04-30	72.1060	89.5142
2019-03-29	59.1869	72.5735
2019-02-28	52.2875	54.2304
2019-01-31	58.3607	50.7567
2018-12-31	68.6283	51.8754

Data Overview

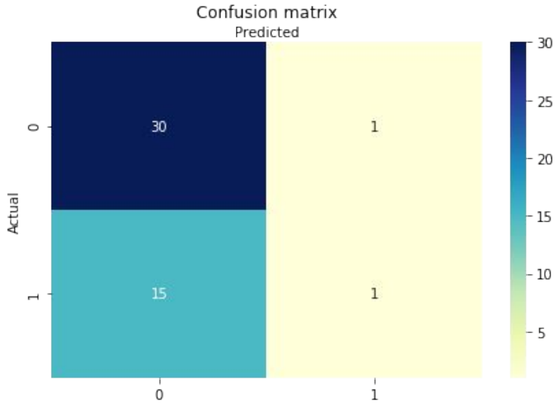
	timestamp	action	positive	appearance	positive2	SlowD	SlowK
0	2019-05-14	0.0	0.544137	3	0.666667	72.1060	89.5142
1	2019-04-30	1.0	0.562695	4	0.500000	59.1869	72.5735
2	2019-03-29	1.0	0.556610	2	0.500000	52.2875	54.2304
3	2019-02-28	1.0	0.548660	9	0.500000	58.3607	50.7567
4	2019-01-31	1.0	0.561755	0	0.000000	68.6283	51.8754



MODEL IMPLEMENTATION

Logistic Regression

```
from sklearn.model_selection import train_test_split

all_data['timestamp'] = pd.to_datetime(all_data['timestamp'], format='%Y-%m-%d')
all_data.index = all_data['timestamp']

feature_cols = ['positive', 'appearance', 'positive2', 'SlowD', 'SlowK']
# feature_cols = ['positive2']
target = 'action'
X = all_data[feature_cols]
y = all_data[target]

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=2)

# instantiate the model (using the default parameters)
logreg = LogisticRegression()

# fit the model with data
logreg.fit(X_train,y_train)

#
y_pred=logreg.predict(X_test)
y_conf=logreg.decision_function(X_test)
```



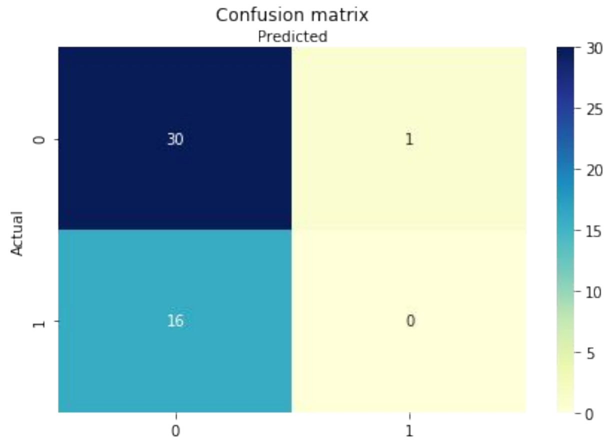
Accuracy: 0.6595744680851063

MODEL IMPLEMENTATION

Support Vector Machine

```
from sklearn import svm

# Initiate model
svc = svm.SVC(kernel='rbf')

# Fit model
svc.fit(X_train, y_train)
```

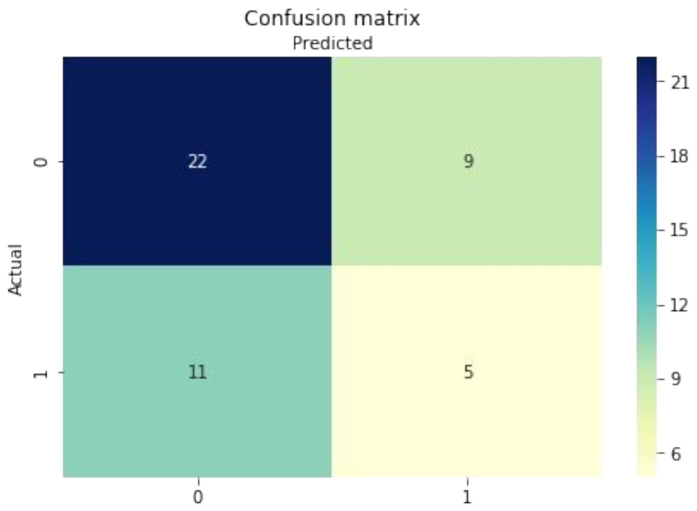
Accuracy: 0.6382978723404256



MODEL IMPLEMENTATION

Sequential Kernel

```
# Import `Sequential` from `keras.models`
from keras.models import Sequential

# Import `Dense` from `keras.layers`
from keras.layers import Dense

# Initialize the constructor
model = Sequential()

# Add an input layer
model.add(Dense(12, activation='relu', input_shape=(5,)))

# Add one hidden layer
model.add(Dense(8, activation='relu'))

# Add an output layer
model.add(Dense(1, activation='sigmoid'))
```



Accuracy: 0.574468085106383

Trading Action Classification using Machine Learning

Quang Lam | Thu Pham

Luther College, Department of Computer Science & Data Science

ABSTRACT

Stock prices fluctuate within seconds and are affected by complicated financial and non-financial indicators. Hence, stock prices prediction is an ambitious project. However, thanks to Machine Learning techniques, we have the capacity of classify trading action from massive amounts of data that capture the underlying stock price dynamics.

INTRODUCTION

In this project, we utilized past data, technical indicators, and economic indexes and applied supervised learning methods to predict stock price action trading (long position/short position) on the next trading month.

As opposed to predicting the trend in short-term which is used in the high-frequency trading market, we intend to forecast the upward and downward movement in the weekly-basis not solely for algorithmic trading, but as a supplement to help investors alike on decision-making.

Our model is currently applied to a single stock symbol.

DATA SOURCE

The project uses the free API from Alpha Vintage (alphavantage.co) to the monthly stock market price historical data in the past 20 years.

Additionally, Alpha Vintage also provides the STOCH index data. By definition, stochastic oscillator is a momentum indicator comparing a particular closing price of a security to a range of its price over a period of time.

Most importantly, the project uses the New York Times Articles API to retrieve all the news headlines New York Times published since January 2000.

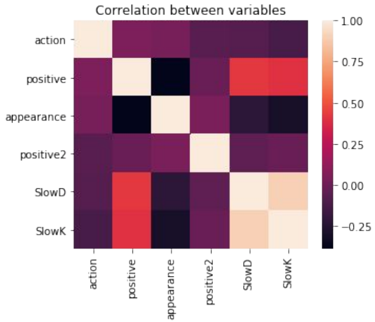
DATA PREPROCESSING

To preprocess the stock market price historical data, we set an expected return value (for example, 2.5%), which is minimal change in the stock price compared to the previous month for a long position. With this threshold value, we add a new variable called **action** with 2 values: 1 represents long position and 0 represents short position.

Then we use sentimental analysis to analyze the New York Times headlines and add 3 variables: **positive** represents the percentage of positive headlines, **appearance** and **positive2** represents the number of times the company appears on the news and the percentage of its positive appearance.

Then STOCH variables and the three mentioned variables are matched with upcoming month while action variable is matched with its real month. It means we would use other variables to predict what to do the upcoming months (**action**).

The correlation of our completed dataset is shown below, using heatmap:

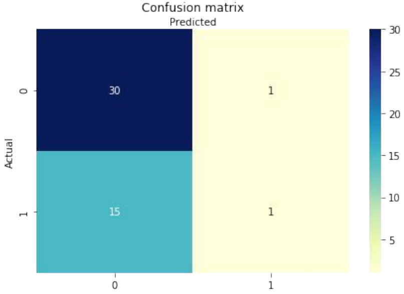


	timestamp	action	positive	appearance	positive2	SlowD	SlowK
0	2019-05-14	0.0	0.544137	3	0.666667	72.1060	89.5142
1	2019-04-30	1.0	0.562695	4	0.500000	59.1869	72.5735
2	2019-03-29	1.0	0.556610	2	0.500000	52.2875	54.2304
3	2019-02-28	1.0	0.548660	9	0.500000	58.3607	50.7567
4	2019-01-31	1.0	0.561755	0	0.000000	68.6283	51.8754

METHODOLOGY

Logistic Regression (LR)

Logistic regression is a simple linear model for classification. The accuracy of this method is approximately 66%. The confusion matrix is presented below:



Support Vector Machine (SVM)

Similar to Logistic Regression, SVM is an algorithm used for classification problems. However, in large dataset, SVM performs marginally better and less sensitive to outliers than LR. In our case, this dataset is considered not big, we do not see an improvement in performance when applying SVM. The accuracy of SVM is approximately 64%.

Neural Networks

Since we are performing binary classification, a multi-layer perceptron is an appropriate method for such model. We implement a **Dense** layer, which is a connect layer. The first two layers take **activation** argument **'relu'** while the last layer takes '**stigmoid'** . The accuracy of this model is roughly 57%

RESULT

The models successfully predict the actions at least 55% of the time with the expected return value close to 0. The expected return value significantly affects the accuracy of the models. The smaller the expected return value, the more likely the decision making fluctuates with the market, so the more least accurately the models predict.

CONCLUSION

As the models only predict between two values: long position or short position, we consider the accuracy of our trained models is not sufficient enough to use in the real world.

With the results of this project, even though we intend to analyze the dataset in the monthly basis instead of daily or weekly to minimize to volatility, we conclude that the stock market is much more volatile than we expected and cannot be predicted accurately using the sentimental analysis of news headline but requiring additional data and methods. Also, our dataset in monthly basis is too small to train the value thoroughly.

Additionally, as we analyze the dataset using three different Machine Learning methods, including one using complex neutral networks, the results are not much different between them. So we conclude that the methodology of our project is not the primary factor of its inaccuracy.

To improve the results, we suggest adding new variables to the models, analyzing the data in the weekly-basis to have more breakpoints for the models to adjust to the volatile market.

ACKNOWLEDGEMENT

Thank you Professor Shafqat Shad and Professor Kent Lee, our instructors, for your support and instructions throughout this project.

We acknowledge the research papers we use to help us conduct this project (available at the link below)

The project is available at: github.com/thu2pham/StockForecasting

THANKS!

Any questions?