STOCK PRICE REVERSAL POINTS

CAPSTONE

AGENDA

- Problem Statement
- Data Collection
- Data Cleaning
- EDA
- Feature Engineering
- Classification Method
- Regression Method
- Final Model
- Conclusion
- Future Progress

PROBLEM STATEMENT

- In the financial markets, traders have always said, Buy low and Sell High (Motto of the Century)

- However, the problem that everyone faces in the investing/ trading world is, at any given moment, we are all questioning, is this really the lowest price now and we should buy the stock now / is this really the highest price now and we should sell the stock to take profit or short the stock?

- In this project, I am going to solve this problem and identify the reversal points (buy/sell points) for a few selected stocks.

DATA COLLECTION

- Data is scraped and collected from Yahoo Finance
- Features collected:
  - High, Low, Open, Close, Volume, Time
- Stocks Scraped:
  - Apple, Facebook, Visa, Intel, Disney, Saleforce, Nike, General Motors, Twitter and TripAdvisor

DATA CLEANING

1)

```python
# Clean up DataFrame
def clean_data(data):
    # Drop all rows with volume == 0. We do not need premarket/postmarket data since we cant trade during these hours
    data = data[data['volume']!=0]

    # Drop Duplicates
    data = data.drop_duplicates(subset=['stock', 'time'], keep = 'last')

    # Drop Null Values
    data = data.dropna()

    # Sort by ticker symbol and time
    data = data.sort_values(by = ['stock','time']).reset_index(drop=True)

    # Create New Column for localtime
    for i, epoch in enumerate(data['time']):
        if data['local_time'][i] == '':
            data['local_time'][i] = time.strftime('%m/%d/%Y %H:%M:%S', time.localtime(epoch))

    # Change local_time datatype from object to date time
    data['local_time'] = pd.to_datetime(data['local_time'])

    return data
```

2)

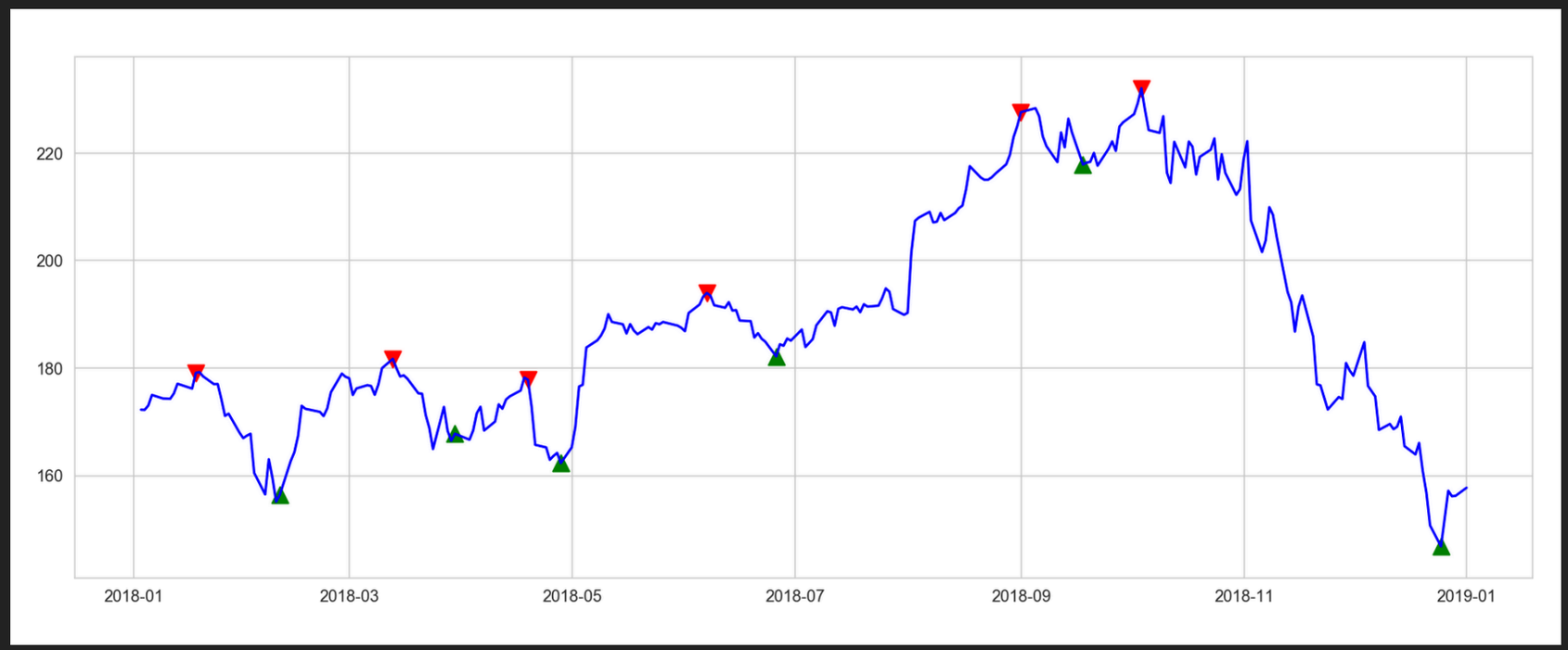
```python
# Combine Scraped Stocks with current dataset
def scraper(ticker):

    # CHECK IF THERE IS ALREADY A CSV FILE
    try:
        all_df = pd.read_csv('../datasets/technical_data.csv')
    except:
        # Create empty dataframe if there is no csv file
        all_df = pd.DataFrame()
        print("No Dataset yet")

    # Get Current time and last day of scraping
    current_time, four_years_ago = get_current_time()

    # Get DataFrame of Scraped Stocks
    scraped_df = scrape_yahoo(ticker, current_time, four_years_ago)

    # Combine current data with scraped data
    all_df = all_df.append(scraped_df).reset_index(drop=True)

    # Clean DataFrame
    cleaned_df = clean_data(all_df)

    # Save DataFrame to CSV
    cleaned_df.to_csv('../datasets/technical_data.csv', index = False)

    return cleaned_df
```

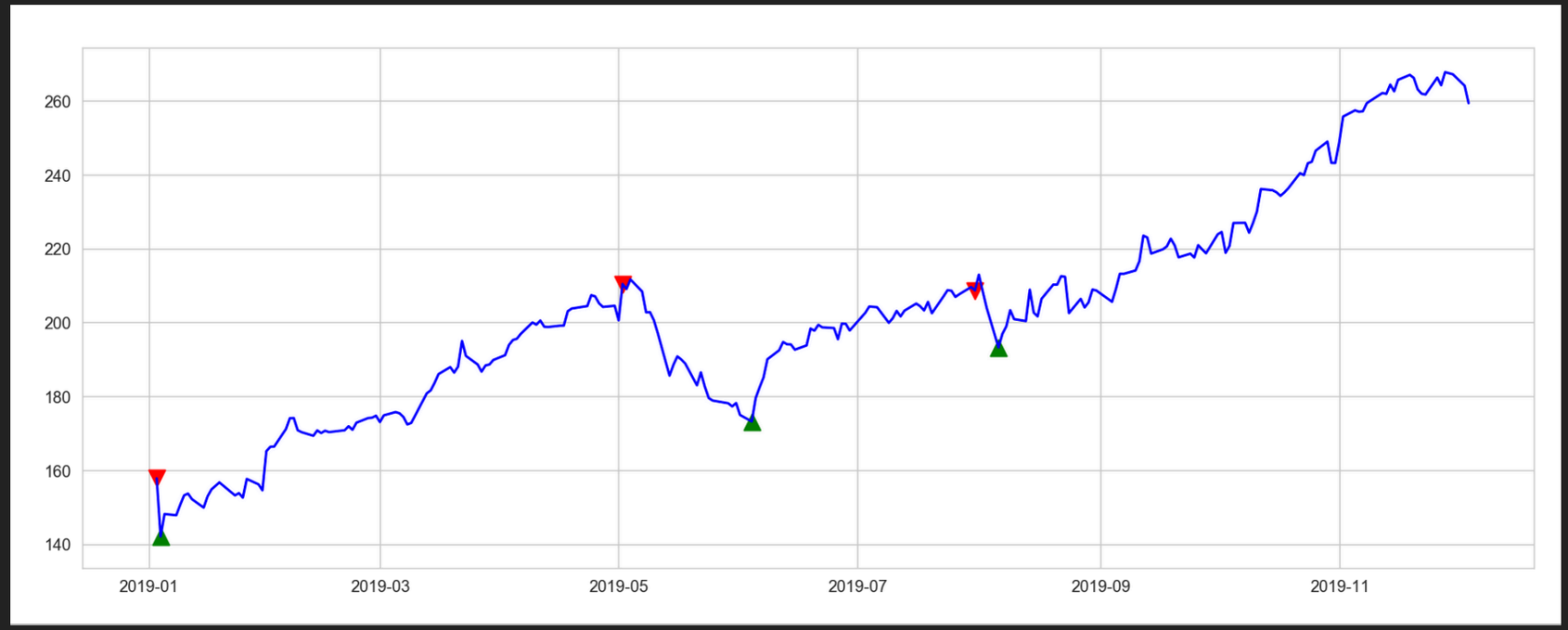
DATA CLEANING

- Results: 10060 Observations, 8 Columns
- Applied a function to get reversal points

	stock	time	local_time	open_price	close_price	highest_price	lowest_price	volume	gain	reversal
0	AAPL	1449239400	2015-12-04 22:30:00	115.290001	119.029999	119.250000	115.110001	57777000	1	2
1	AAPL	1449498600	2015-12-07 22:30:00	118.980003	118.279999	119.860001	117.809998	32084200	-1	0
2	AAPL	1449585000	2015-12-08 22:30:00	117.519997	118.230003	118.599998	116.860001	34309500	1	0
3	AAPL	1449671400	2015-12-09 22:30:00	117.639999	115.620003	117.690002	115.080002	46361400	-1	0
4	AAPL	1449757800	2015-12-10 22:30:00	116.040001	116.169998	116.940002	115.510002	29212700	1	0
5	AAPL	1449844200	2015-12-11 22:30:00	115.190002	113.180000	115.389999	112.849998	46886200	-1	0
6	AAPL	1450103400	2015-12-14 22:30:00	112.180000	112.480003	112.680000	109.790001	64318700	1	0
7	AAPL	1450189800	2015-12-15 22:30:00	111.940002	110.489998	112.800003	110.349998	53323100	-1	0
8	AAPL	1450276200	2015-12-16 22:30:00	111.070000	111.339996	111.989998	108.800003	56238500	1	0
9	AAPL	1450362600	2015-12-17 22:30:00	112.019997	108.980003	112.250000	108.980003	44772800	-1	0

EDA (APPLE)





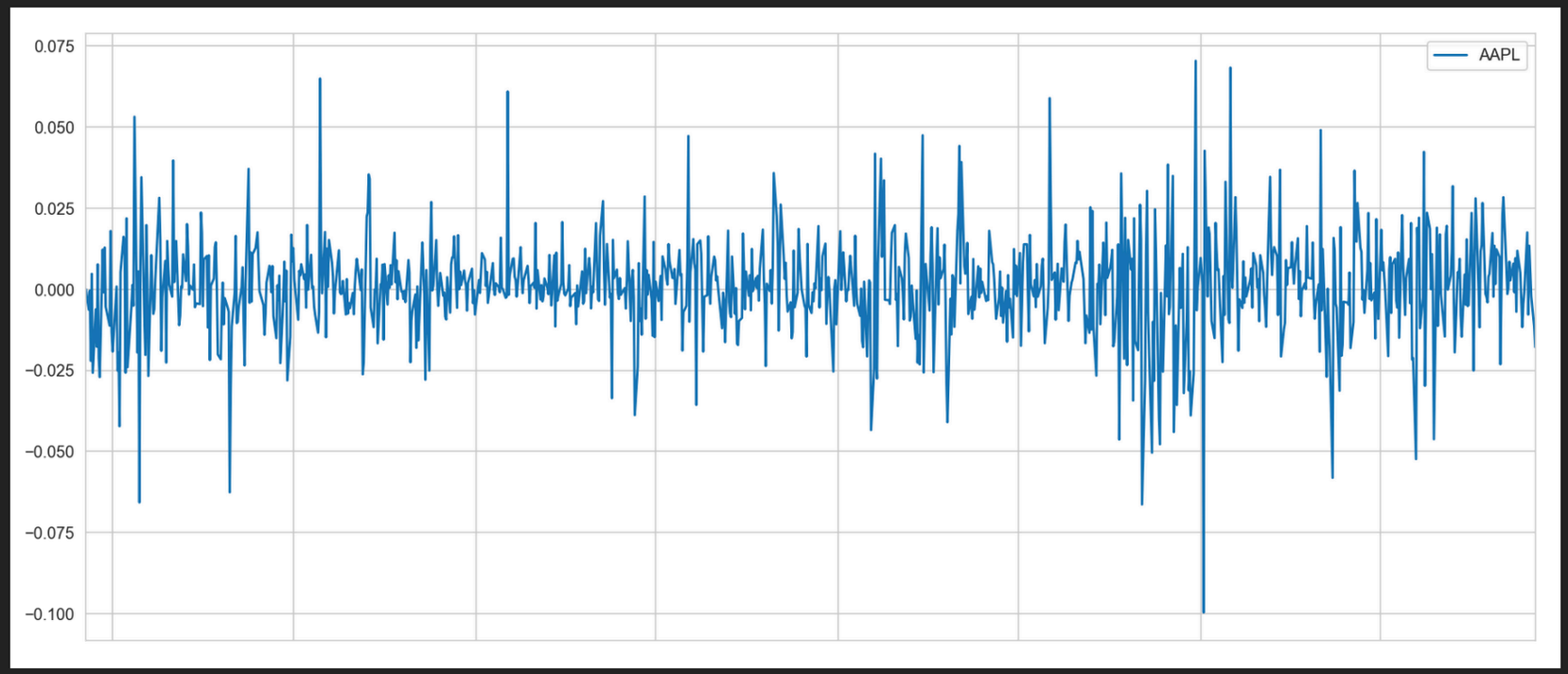
- Blue - Price
- Red - Reversal Point (Sell)
- Green - Reversal Point (Buy)

FEATURE ENGINEERING

- Check for Stationarity

```
ADF Statistic for AAPL: -0.17336167360079732
p-value for AAPL: 0.9416232414164956
Critial Values:
   1%, -3.4369325637409154
Critial Values:
   5%, -2.8644462162311934
Critial Values:
   10%, -2.568317409920808
P-Value is more than 0.05, Null Hypothesis cannot be Rejected. Hence, AAPL is not Stationary.
----------------------------------------
```

- To make it Stationary, we take the %change



FEATURE ENGINEERING

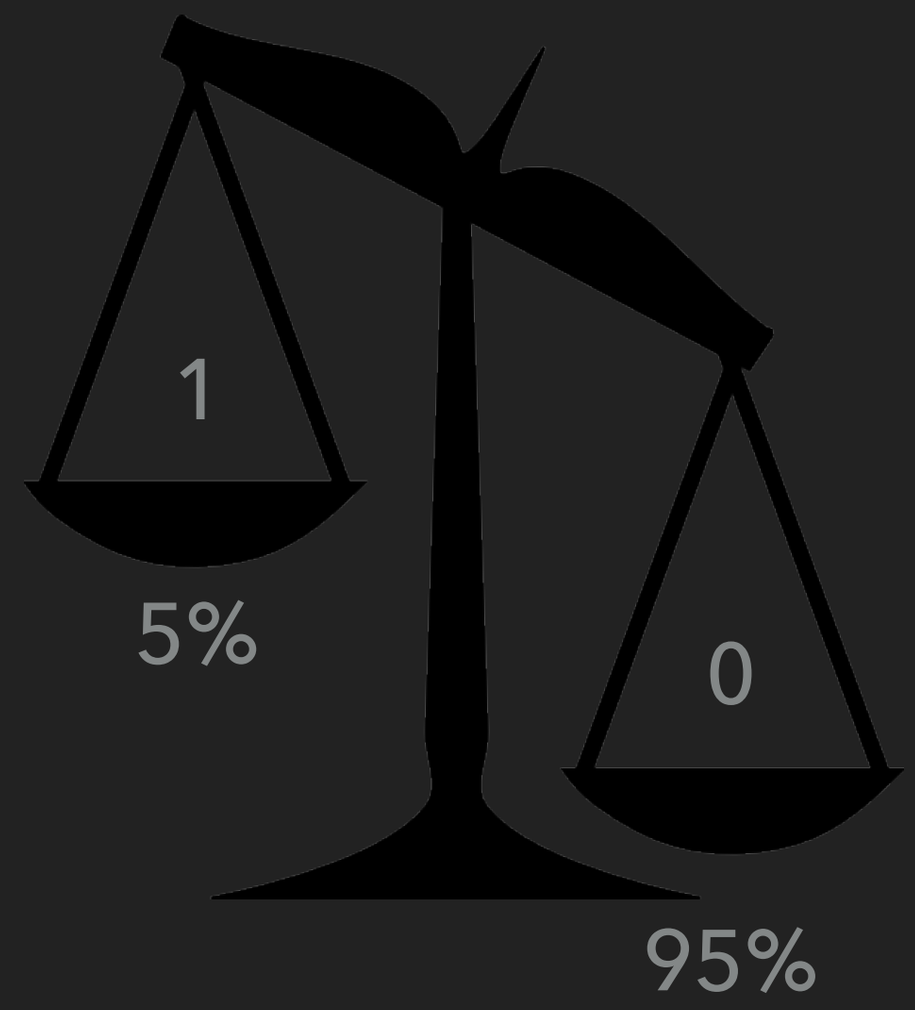
- Created other features such as:
  - Returns
  - Exponential Moving Average
  - Volatility
  - Etcetc
- Total Number of Features: 80

```
df.shape
```

```
(10060, 80)
```

CLASSIFICATION METHOD

Unbalanced Dataset



1 = Reversal Point

0 = Non Reversal Point

CLASSIFICATION

- Models Used:
  - Logistic Regression
  - Random Forest
  - KNearestNeighbor
  - SVM
  - Naive Bayes
  - GLM Poisson
  - Ada Boost
  - Gradient Boost

CLASSIFICATION

Logistic Regression

True Negative

False Positive

```
[1323,  616]
[  35,   46]
```

False Negative

True Positive

F1 Score: 0.123

CLASSIFICATION

Random Forest

True Negative

False Positive

```
[1931,    8]
[  81,    0]
```

False Negative

True Positive

F1 Score: 0.000

CLASSIFICATION

K Nearest Neighbors

True Negative

False Positive

```
[1939,    0]
[  81,    0]
```

False Negative

True Positive

F1 Score: 0.000

CLASSIFICATION

SVM

True Negative

False Positive

```
[1939,    0]
[  81,    0]
```

False Negative

True Positive

F1 Score: 0.000

CLASSIFICATION

Naive Bayes

True Negative

False Positive

```
[1749,  190]
[  67,   14]
```

False Negative

True Positive

F1 Score: 0.098

CLASSIFICATION

Ada Boost

True Negative

False Positive

```
[1791,  148]
[  67,   14]
```

False Negative

True Positive

F1 Score: 0.115

CLASSIFICATION

Gradient Boost

True Negative

False Positive

```
[1753,  186]
[  74,    7]
```

False Negative

True Positive

F1 Score: 0.053

REGRESSION

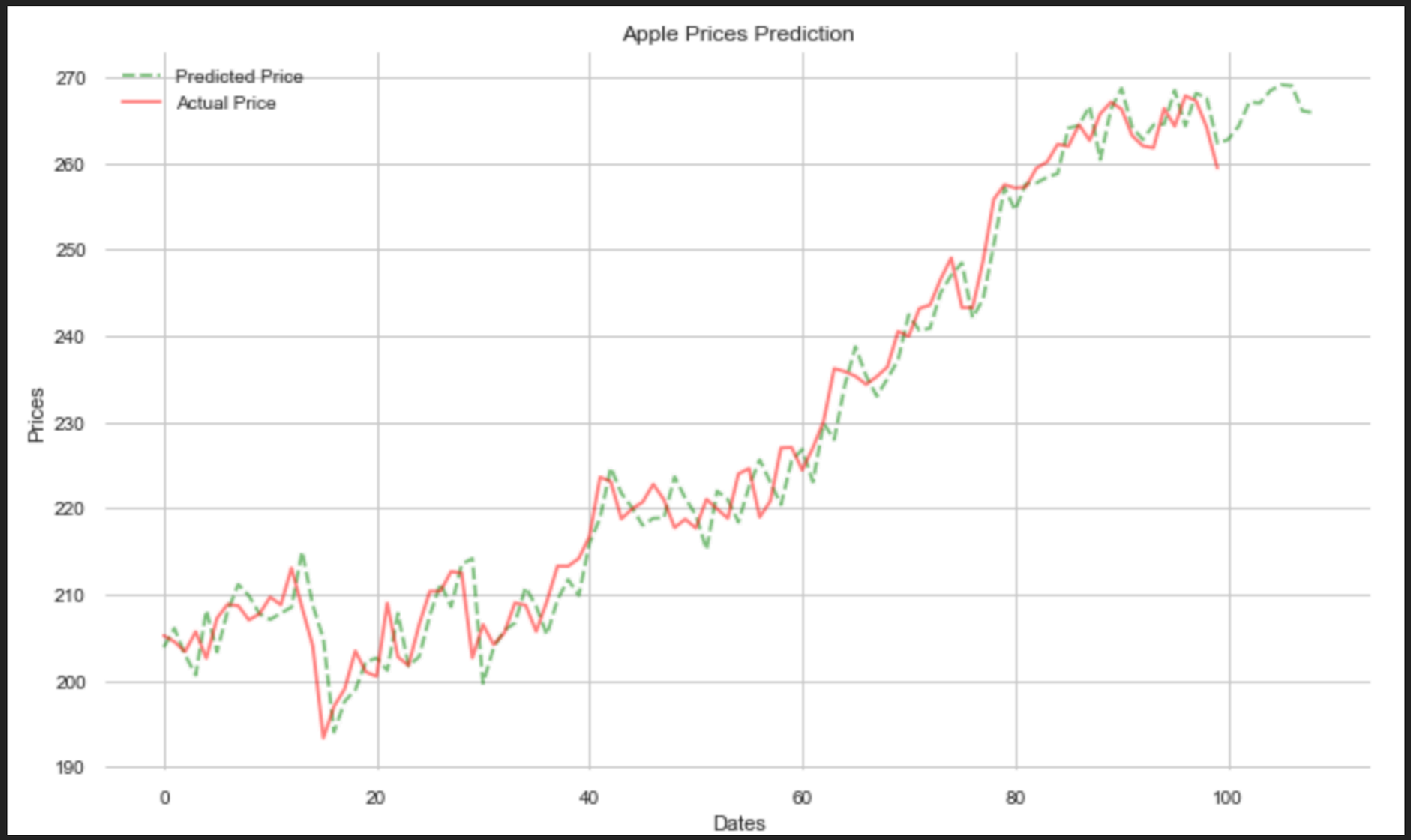
- Forecast the future prices
- Use Forecasted price to Classify Reversal Points
- Models Used:
  - LassoCV
  - Random Forest
  - AdaBoost
  - Gradient Boost
  - SVM
  - XGBoost
  - ARIMA
  - SARIMA

REGRESSION

Regression Model	RMSE Score
ARIMA	11.30
SARIMA	13.72
LassoCV	15.64
XGBoost	20.16
Random Forest	24.04
Ada Boost	27.60
SVM	408.15
Gradient Boost	136889264623 82.30

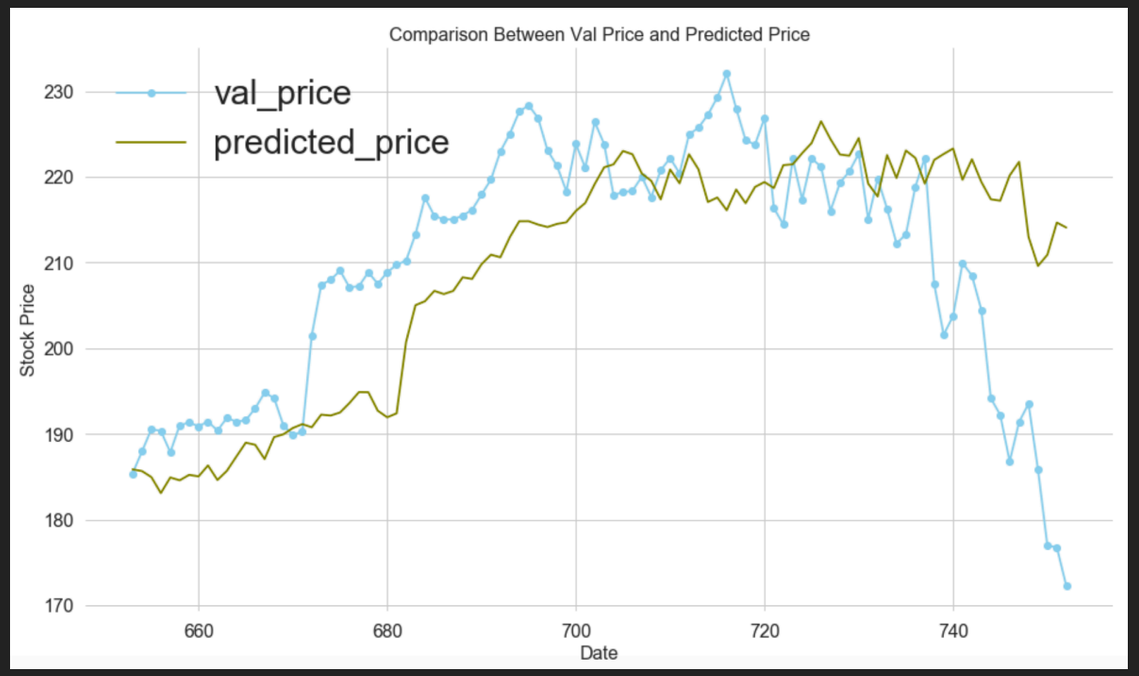
FINAL MODEL

- SARIMA (Order = (3,0,3), Seasonal Order = (3,1,0,8))

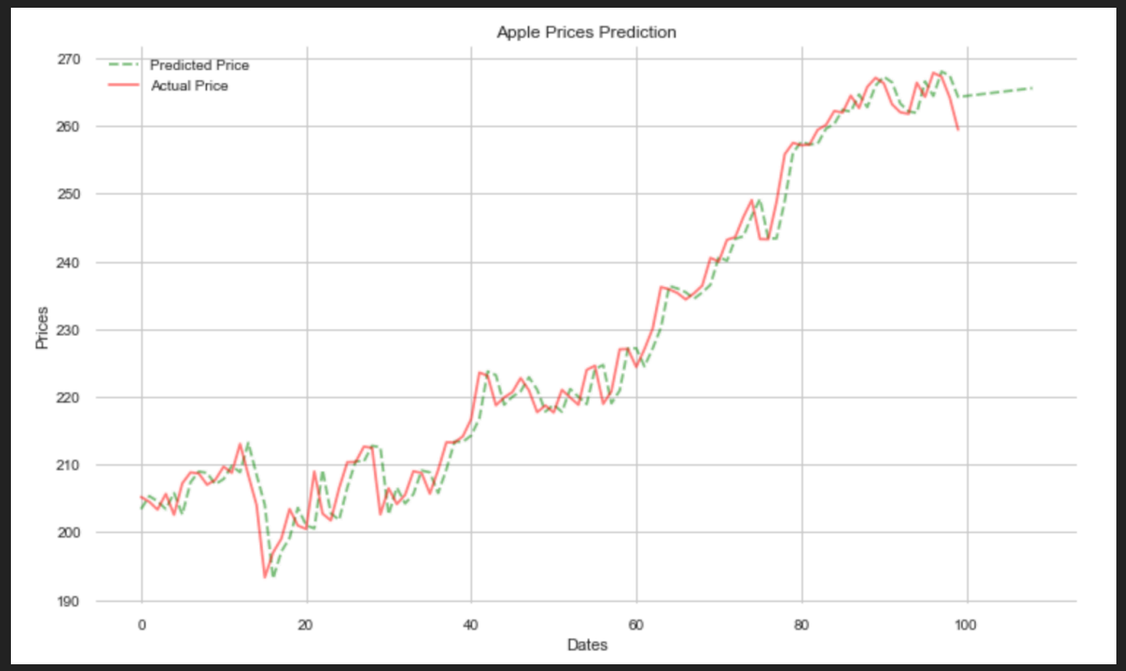


FINAL MODEL

LassoCV

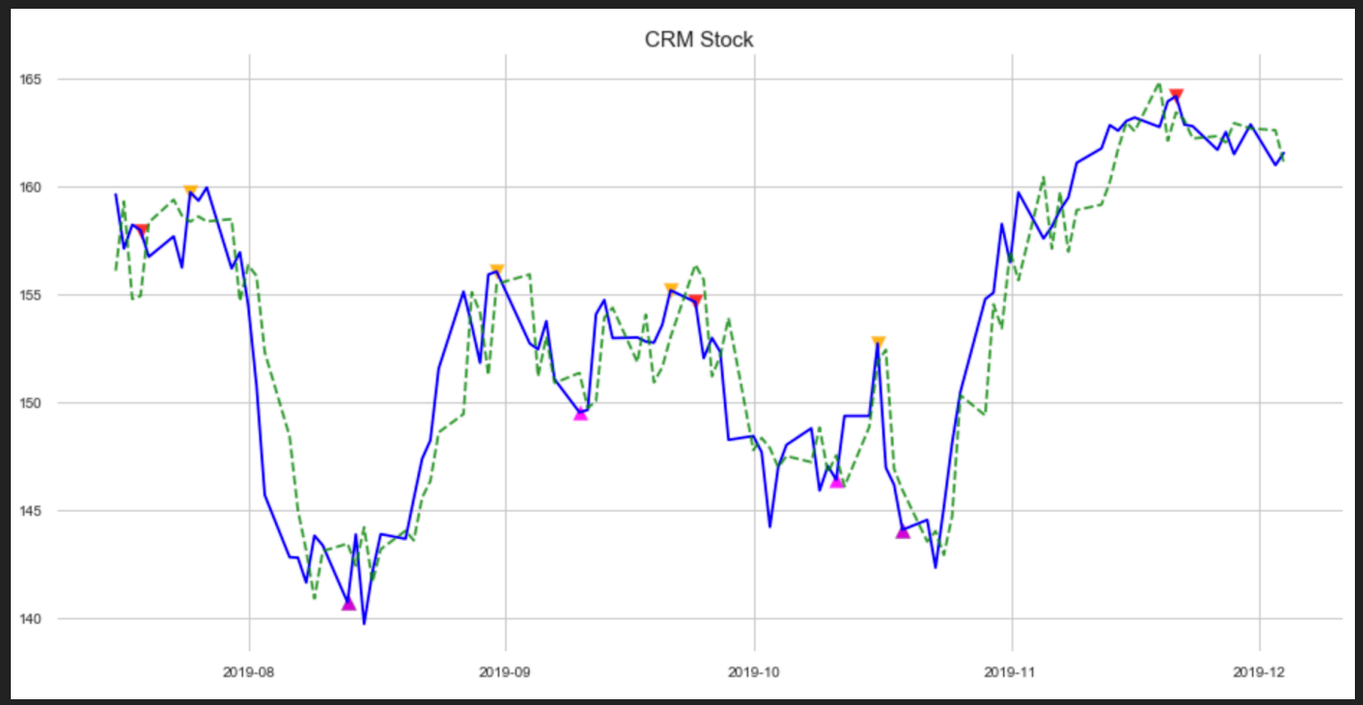


ARIMA



CONCLUSION

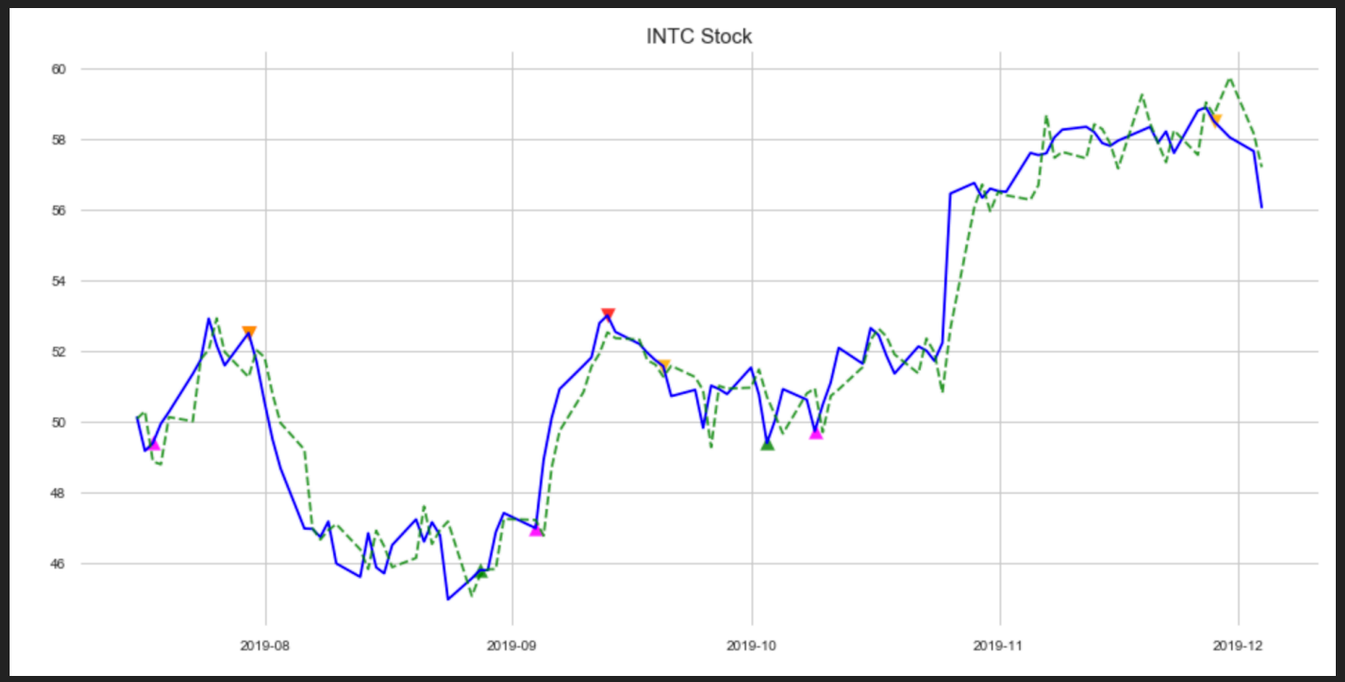
- Although the confusion matrix is still bad, I think the model is still quite acceptable to be used for its purpose.



Blue - Actual Price
Green-Dotted - Predicted Price
Red Arrow - Actual Reversal (Sell)
Green Arrow - Actual Reversal (Buy)
Orange Arrow - Pred Reversal (Sell)
Purple Arrow - Pred Reversal (Buy)

CONCLUSION

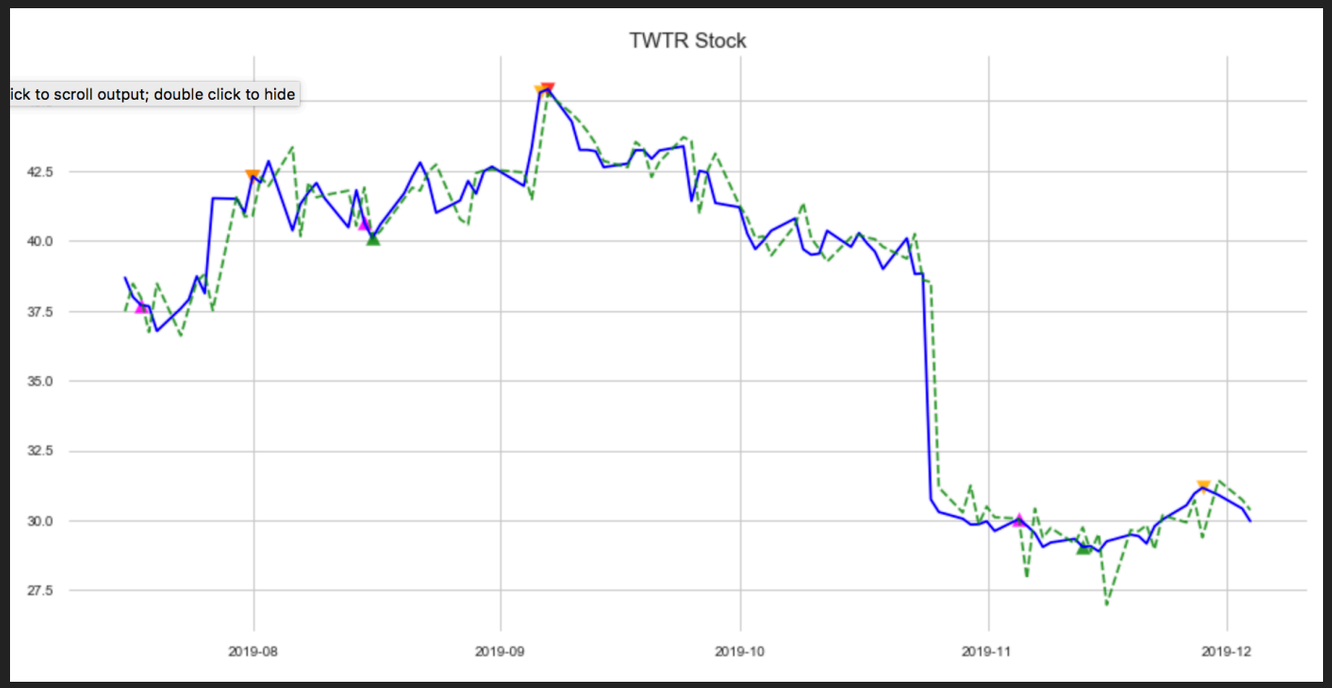
- Although the confusion matrix is still bad, I think the model is still quite acceptable to be used for its purpose.



Blue - Actual Price
Green-Dotted - Predicted Price
Red Arrow - Actual Reversal (Sell)
Green Arrow - Actual Reversal (Buy)
Orange Arrow - Pred Reversal (Sell)
Purple Arrow - Pred Reversal (Buy)

CONCLUSION

- Although the confusion matrix is still bad, I think the model is still quite acceptable to be used for its purpose.



Blue - Actual Price
Green-Dotted - Predicted Price
Red Arrow - Actual Reversal (Sell)
Green Arrow - Actual Reversal (Buy)
Orange Arrow - Pred Reversal (Sell)
Purple Arrow - Pred Reversal (Buy)

FURTHER DEVELOPMENT

- Deployment of Model (Returns will be Revealed during Meet and Greet)
- Improve Forecasting Model
  - SARIMAX + GARCH
  - Using Fundamentals as features
    - Sentiment Analysis on the News
    - Deep Learning Models
- Create Evaluation Metrics to evaluate Model
- Use different GridSearch Metric (AIC, BIC)