Algorithmic Trading Strategy for a Single Instrument

Aryaman, Jason, Pranitha, Baha

Strategy Development

- We test multiple models and choose the best performing one; the one with low RMSE and high profit.
- We use a polynomial regression model to predict tomorrow's closing price given the past 15 days closing price data. If the closing price for today is greater than the model's predicted price, we place a buy order. And if it is less than, we place a sell order.
- We also choose to buy all in/ sell all out

Pseudocode

```
# prediction predicts closing price for next day from past 15 days data

<generate dataset based on closing price for last 15 days>
<Train polynomial regression on the dataset>
//predict tomorrow's price
prediction = polynomial_model.predict(tomorrow)

if prediction > todays close price:
     buy tomorrow

else if prediction < todays closing price:
     sell tomorrow
```

```
def next(self):
    days = 15
    data = {
            'Close' : [self.dataclose[i] for i in range(-days + 1, 1)],
            'row_num' : [i for i in range(0, days)]
        }
    df = pd.DataFrame (data, columns = ['Close', 'row_num'])
    stock_df = df
    X = stock_df[['row_num']]
    y = stock_df.Close
    poly_features = PolynomialFeatures(degree = 2, include_bias = False)
    X_poly = poly_features.fit_transform(X)
    reg_f = LinearRegression()
    reg_f.fit(X_poly, y)
    predicted = reg_f.predict([[days, (days)^2]])[0]
    if self.dataclose[0] < predicted:
        if (len(self.position) == 0):
            self.order = self.order_target_percent(target=1)
            self.buy_count += len(self.position)
    if self.dataclose[0] > predicted:
        self.order = self.sell(size=len(self.position))
        self.sell_count += len(self.position)
```

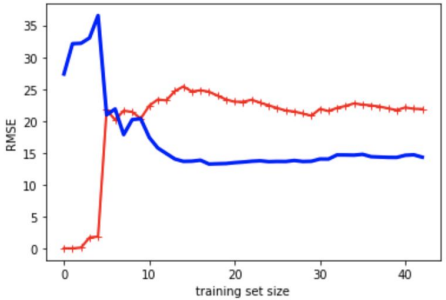
Risks

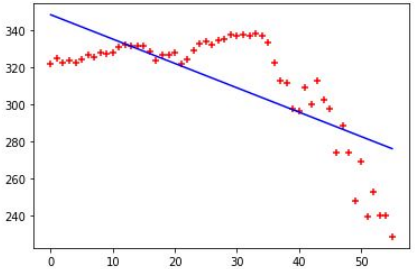
The strategy like any other investment strategy has its own risks.

- We use up all our capital for each buy and sell transaction
- No risk calculations
- Can’t predict daily dips
    - We cannot predict sudden changes in stock prices
- High frequency of trade

Predictive Models

Linear Regression

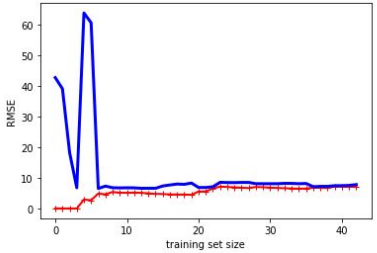


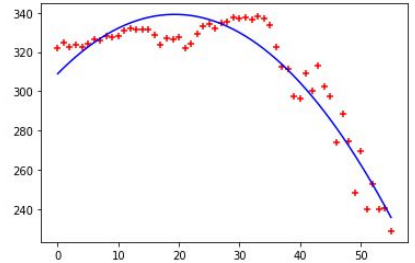


Blue = validation set
Red = training set

RMSE train 21.529
RMSE val 15.410
R^2 train 0.530
R^2 val 0.211

Polynomial regression (second degree)



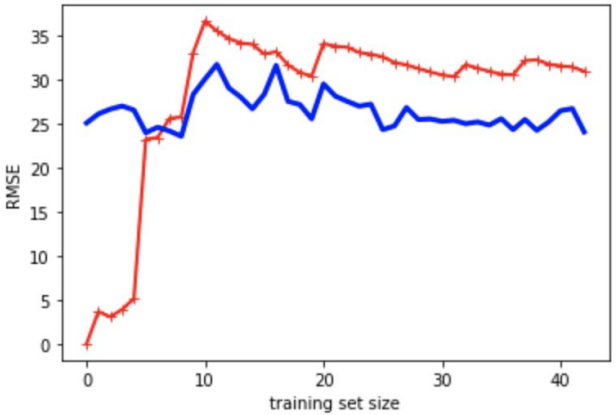


Blue = validation set
Red = training set

RMSE train 9.222
RMSE val 9.670
R^2 train 0.917
R^2 val 0.473

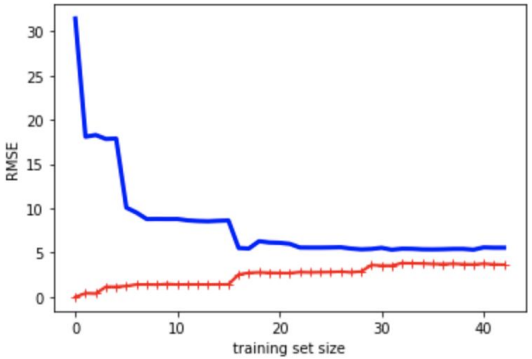
Pred. Models (Contd.)

Random forest



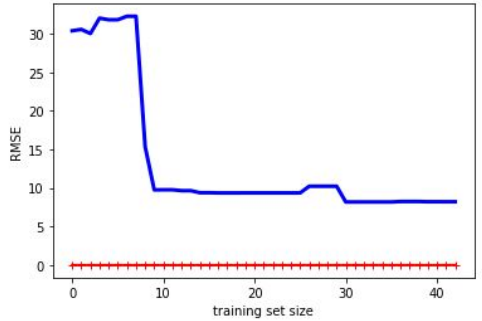
Modelled on 3 year stock data:
RMSE train : 29.734
RMSE val : 28.961
R^2 train : -0.013
R^2 val : 0.007

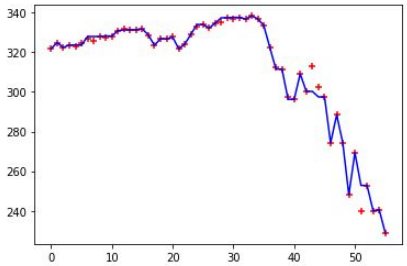
Gradient boosting regressor RMSE



Modelled on 3 year stock data:
RMSE train 3.244
RMSE val 5.057
R^2 train 0.987
R^2 val 0.967

Xg boost



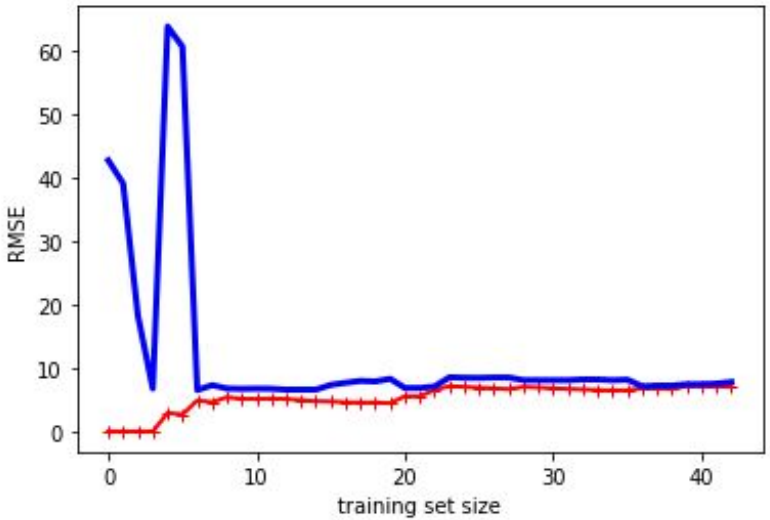


Model Selection

We choose Polynomial Regression as our predictive model because of its:

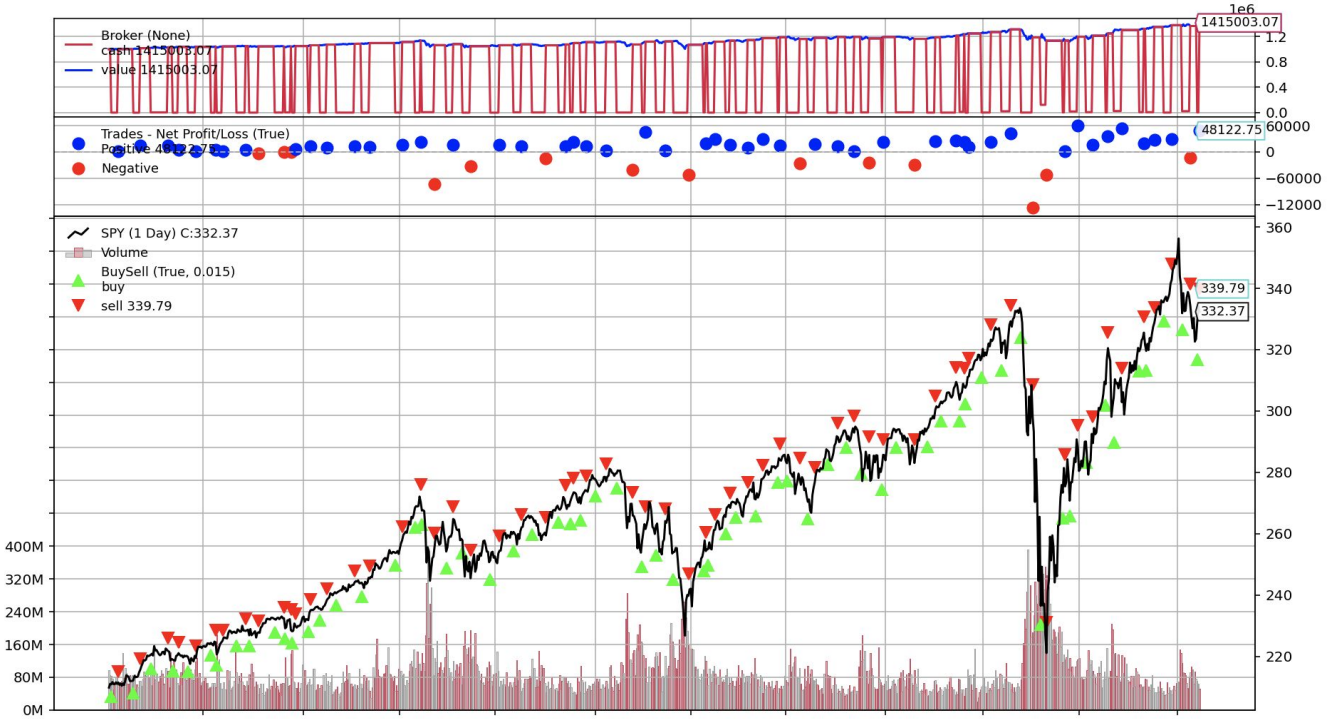
- no bias-variance trade off, and
- low RMSE score

Model is trained on closing price data of 15 days.



Back-testing

```
Starting Portfolio Value: $1000000.00
ROI:         41.50%
Final Portfolio Value: $1415003.07
P/L: $415003.06999999983
```



Comparative Analysis

Buy and hold ROI (benchmark) = 57.68% = $576803.37 (Profit made)

Polynomial regression model ROI = 41.5%

- We observe a profit of $576803.37 on our initial capital of $1000000 USD.
- A long position in the ETF generates a 57.68% return while the ROI on the polynomial regression model generates 41.5%.

```
Starting Portfolio Value: $1000000.00
Position: 4719
ROI:        57.68%
Final Portfolio Value: $1576803.37
P/L: $576803.3700000001
```