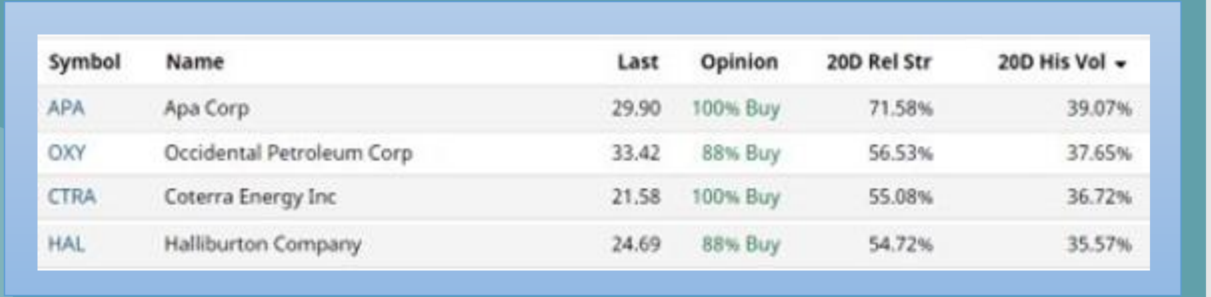
Team 9

QUANTT ML & Momentum Financial Algorithm

Market Strategy



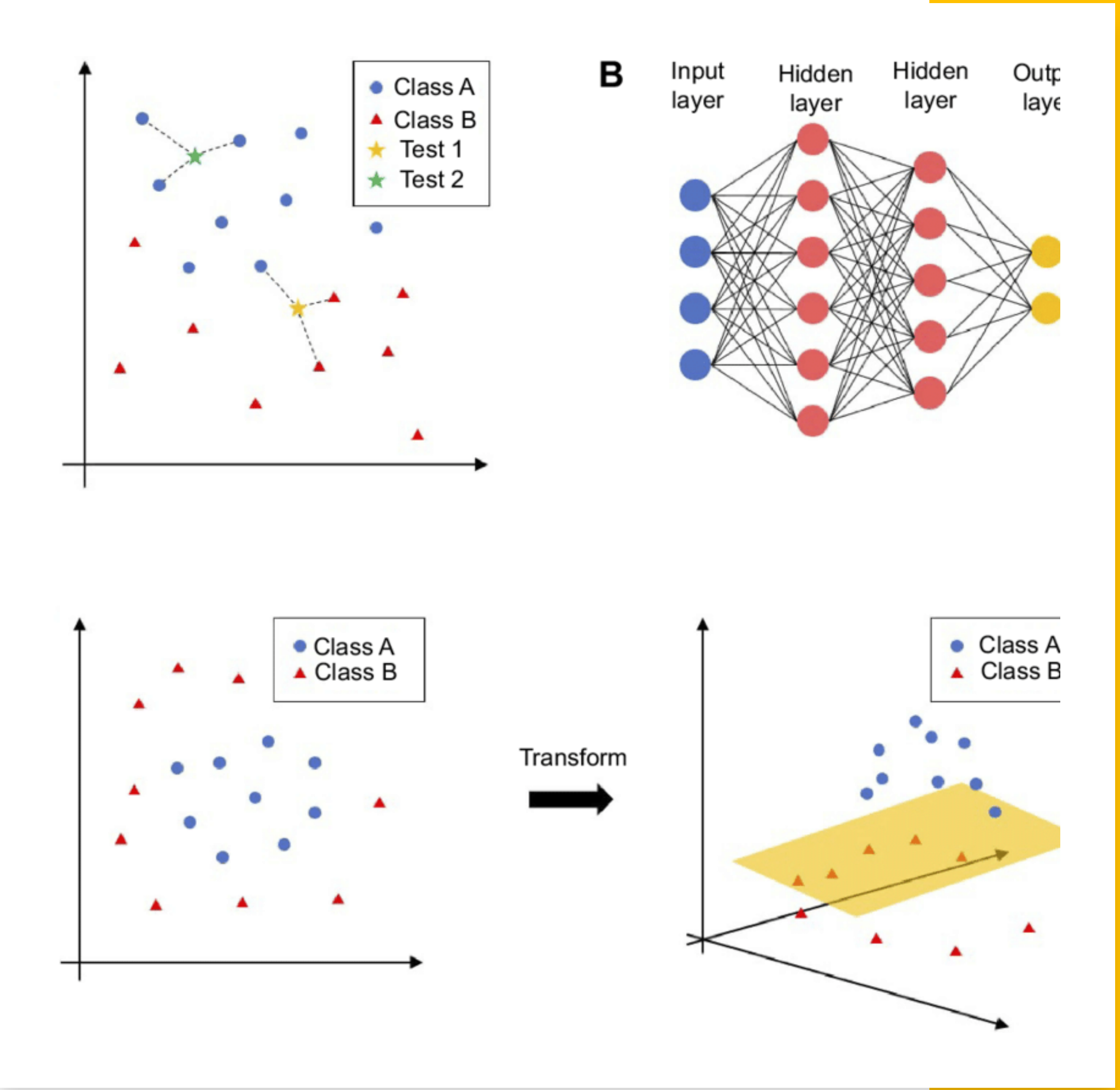
Symbol	Name	Last	Opinion	20D Rel Str	20D His Vol ▾
APA	Apa Corp	29.90	100% Buy	71.58%	39.07%
OXY	Occidental Petroleum Corp	33.42	88% Buy	56.53%	37.65%
CTRA	Coterra Energy Inc	21.58	100% Buy	55.08%	36.72%
HAL	Halliburton Company	24.69	88% Buy	54.72%	35.57%

Hypothesis

“Since the Energy sector and more specifically the ‘APA’, ‘OXY’, ‘CTRA’, ‘HAL’ had the highest **volatility** based on the Beta factor from Vanguard ETF’s Analysis, this equities were chosen for the algorithm which would take advantage of these quickly changing patterns to better predict behavior.”

ML Algorithms

- Implemented KNN algorithm to estimate output for next purchase
- KNN is a supervised classification algorithm
  - Computes distances between select number of neighboring data points to fit data into classes
- Linear regression, Random Forest, SVR and XGB Regressor models were tested
  - KNN was ultimately chosen because it had the highest percent return
  - Phase 1 of Algorithm with only S&P500



Momentum Portfolio Strategy

- Selection of Securities
  - Energy Sector Volatility
  - Betas
  - Limited Diversification
- Use of Leverage
  - Avoiding Insufficient Funds Error
  - Amplified Returns

High-Level Function Overview:

1. Retrieve QuantConnect MOMP values for each security
2. Exclude negative momentum values; liquidate portfolio if all momentums are negative
3. Allocate funds to securities based on their relative momentum values

Bullish Function

- Momentum Strategy

- linear relationship to estimate the next peak, trough and data point

- Improve Strategy with ML Algorithm

- Use ML Algorithm instead of linear relationship: non-linear relationship, prediction

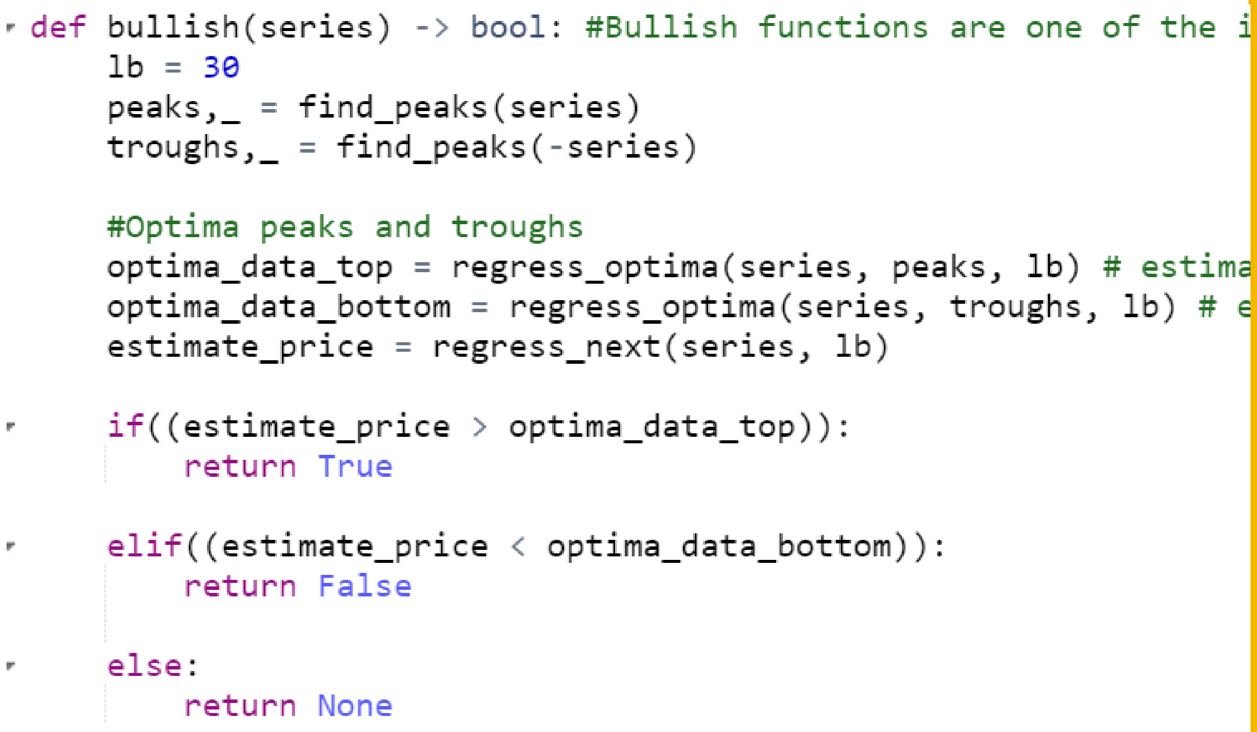
High-Level Function Overview:

1. Find historical 30 peaks, troughs and prices.
2. Estimate the next peak, trough and price by different ML methods.
3. If prediction result of next price > next peak, then choose to buy stock. Otherwise not buy the stock.

Explain Algorithms

- How we use the ML Algorithm
- Use historical prices predicts next peak/trough instead of linear method
- KNN - lazy learner, low time and high accuracy
- Xgboost - gradient boost tree
- Random forest – bagging + decision tree
- Tree methods has high time computational cost, KNN has low time consuming

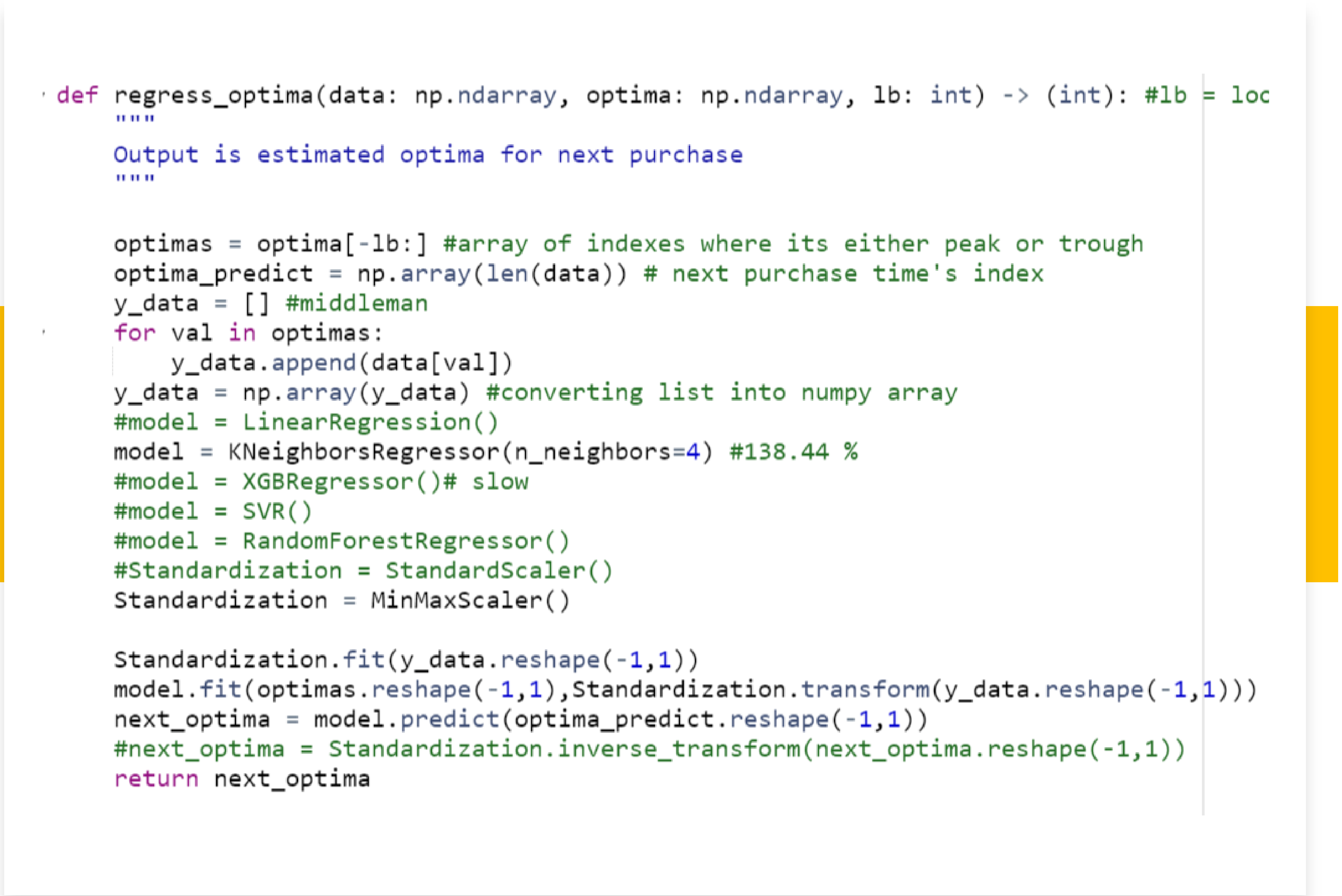
Sample code for bullish function:



```
def bullish(series) -> bool: #Bullish functions are one of the i
    lb = 30
    peaks,_ = find_peaks(series)
    troughs,_ = find_peaks(-series)

    #Optima peaks and troughs
    optima_data_top = regress_optima(series, peaks, lb) # estima
    optima_data_bottom = regress_optima(series, troughs, lb) # e
    estimate_price = regress_next(series, lb)

    if((estimate_price > optima_data_top)):
        return True

    elif((estimate_price < optima_data_bottom)):
        return False

    else:
        return None
```

Other Sample Code (Example)

Optima Regression

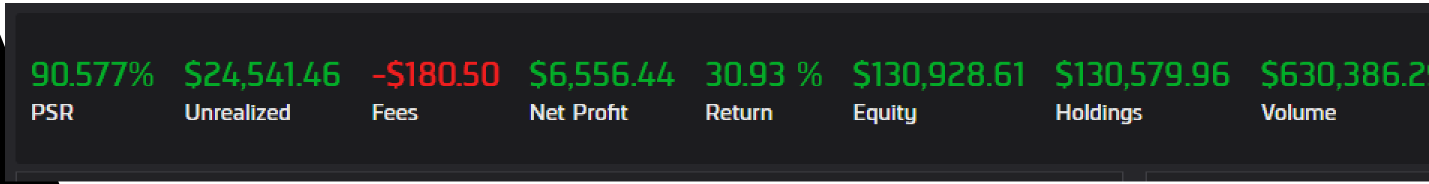


```
def regress_optima(data: np.ndarray, optima: np.ndarray, lb: int) -> (int): #lb = loc
    """
    Output is estimated optima for next purchase
    """

    optimas = optima[-lb:] #array of indexes where its either peak or trough
    optima_predict = np.array(len(data)) # next purchase time's index
    y_data = [] #middleman
    for val in optimas:
        y_data.append(data[val])
    y_data = np.array(y_data) #converting list into numpy array
    #model = LinearRegression()
    model = KNeighborsRegressor(n_neighbors=4) #138.44 %
    #model = XGBRegressor()# slow
    #model = SVR()
    #model = RandomForestRegressor()
    #Standardization = StandardScaler()
    Standardization = MinMaxScaler()

    Standardization.fit(y_data.reshape(-1,1))
    model.fit(optimas.reshape(-1,1),Standardization.transform(y_data.reshape(-1,1)))
    next_optima = model.predict(optima_predict.reshape(-1,1))
    #next_optima = Standardization.inverse_transform(next_optima.reshape(-1,1))
    return next_optima
```

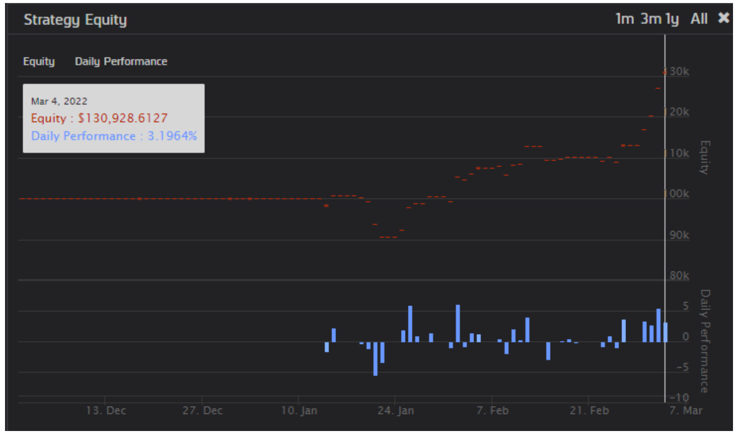
Current Results

Further Information:

- - No orders running on insufficient funds!
- - Win Rate: 61%
- - Compounding Annual Return: 187.958%
- - Beta: 0.165

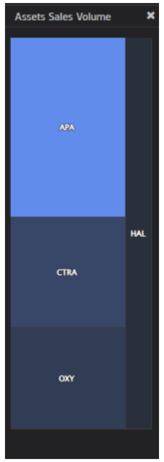


90.577%	$24,541.46	-$180.50	$6,556.44	30.93 %	$130,928.61	$130,579.96	$630,386.2
PSR	Unrealized	Fees	Net Profit	Return	Equity	Holdings	Volume

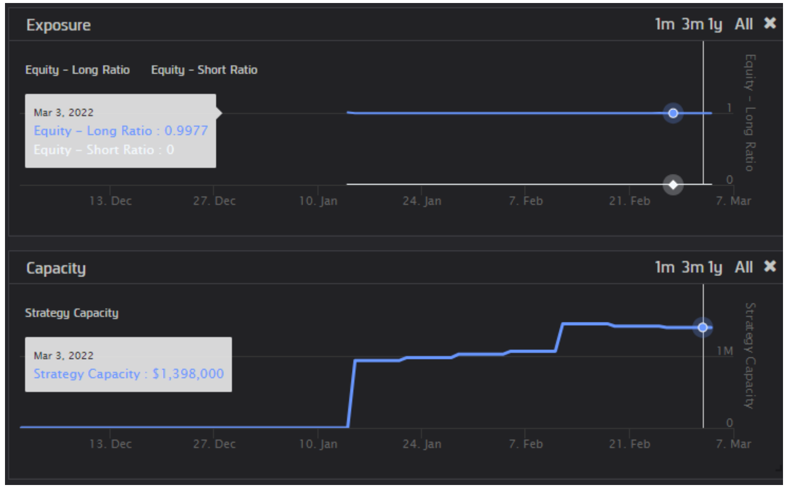


Strategy Equity

Mar 4, 2022
Equity : $130,928.6127
Daily Performance : 3.1964%



Assets Sales Volume



Exposure

Mar 3, 2022
Equity – Long Ratio : 0.9977
Equity – Short Ratio : 0

Capacity

Mar 3, 2022
Strategy Capacity : $1,398,000