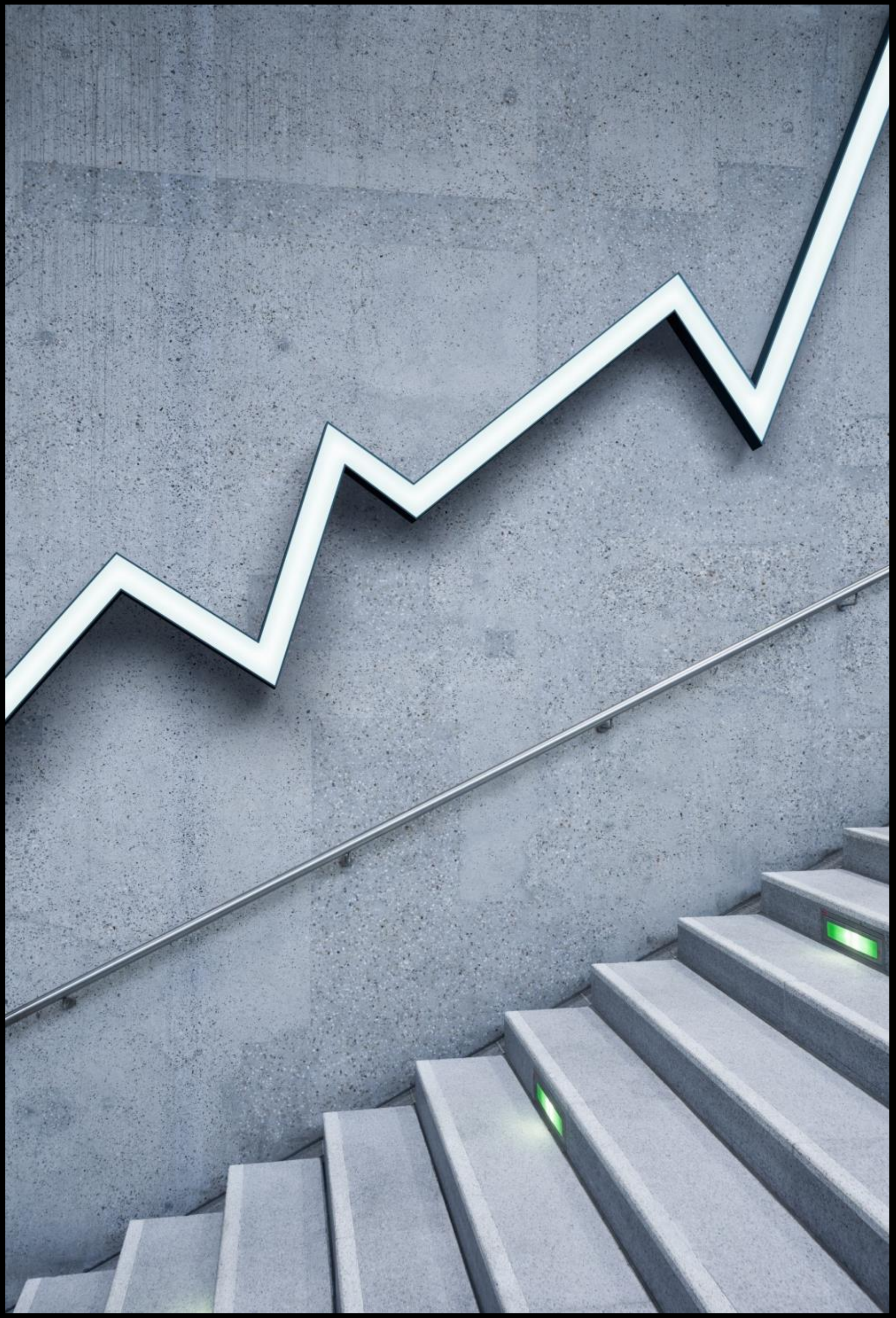
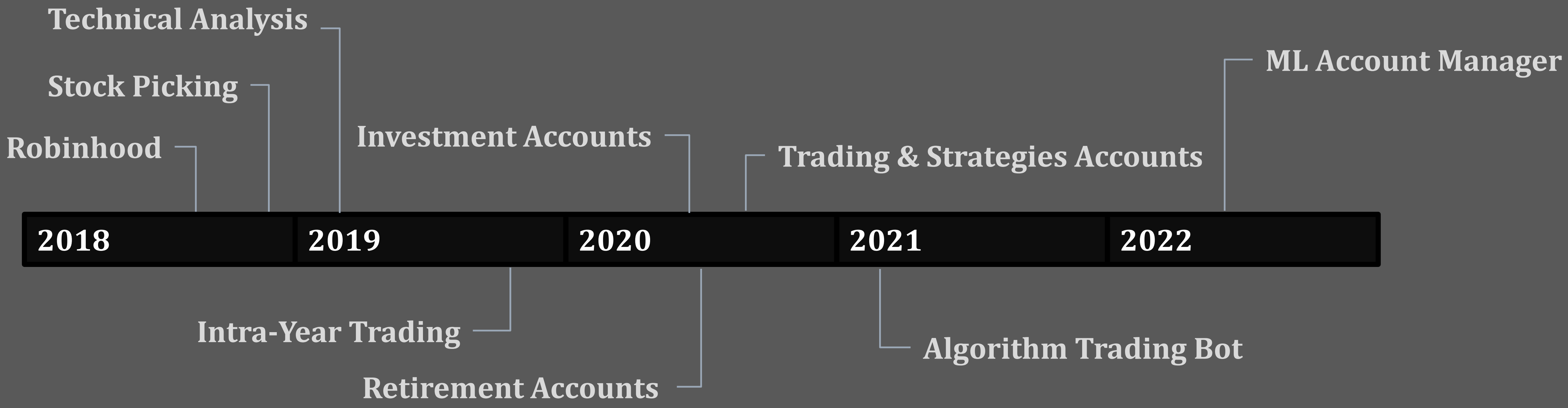


Generating S&P 500 Futures with Machine Learning

Maurice Roots
DATA 602: Term Project (12/14/2021)

- Motivation
  - Investment Journey
  - Dataset Selection
- Background
  - Exploratory Data Analysis
- Model Performance
  - Support Vector Machines (SVM)
  - K-Nearest Neighbors (KNN)
  - Random Forests (RF)
  - Ensemble Models
- Summary



2018

Robinhood

Stock Picking

Technical Analysis

2019

Intra-Year Trading

Investment Accounts

2020

Retirement Accounts

Trading & Strategies Accounts

2021

Algorithm Trading Bot

2022

ML Account Manager

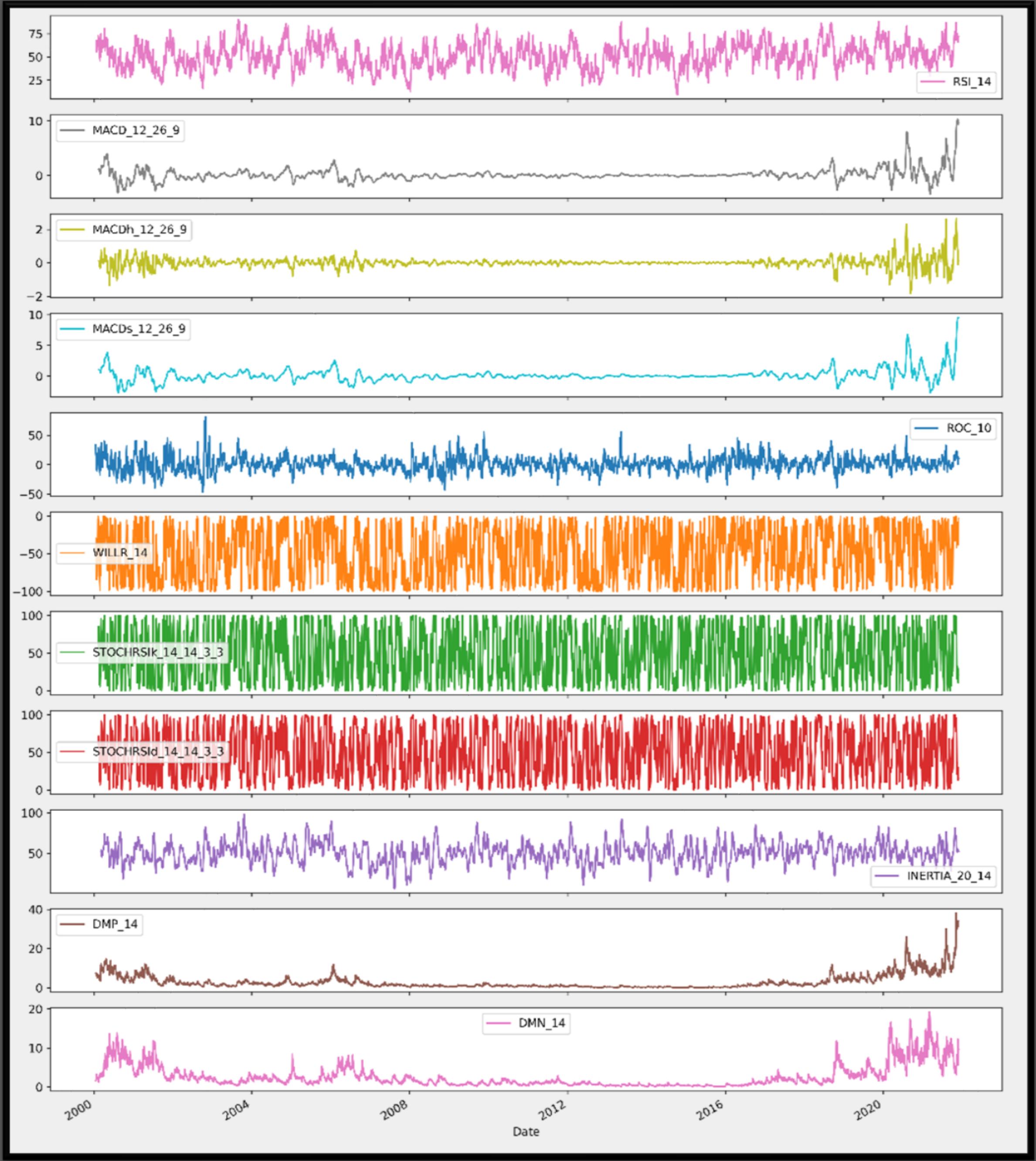
Motivation: Investment Journey

Hypothesis:

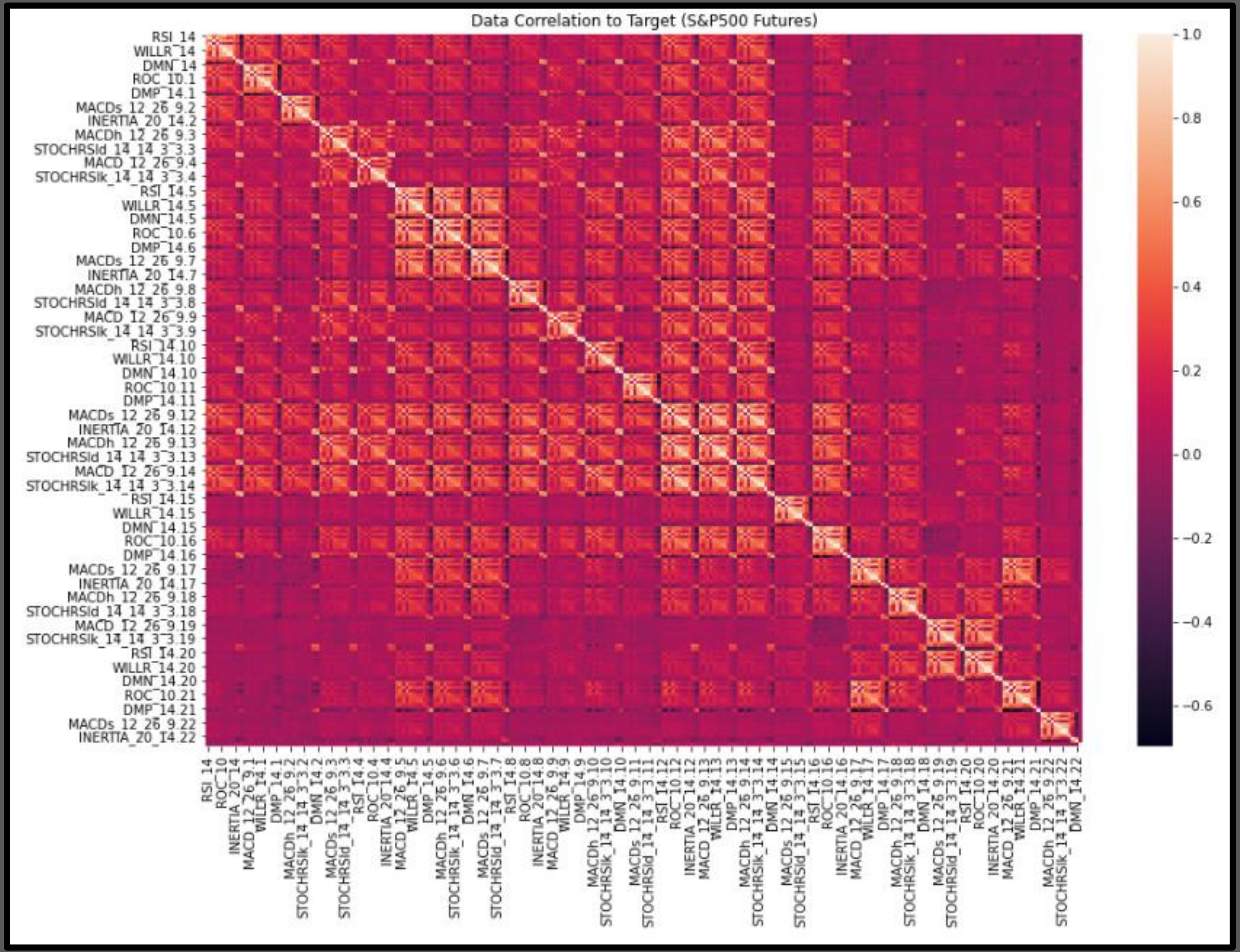
An ensemble model trained on handpicked momentum indicators from selective stocks, indices, and commodities futures may yield sufficient results in predicting S&P 500 futures.

- Price has too much noise!
  - Its not repeatable
  - Varying Scales

- Momentum Oscillators
  - Repeatable!
  - Scaled!



Background: EDA

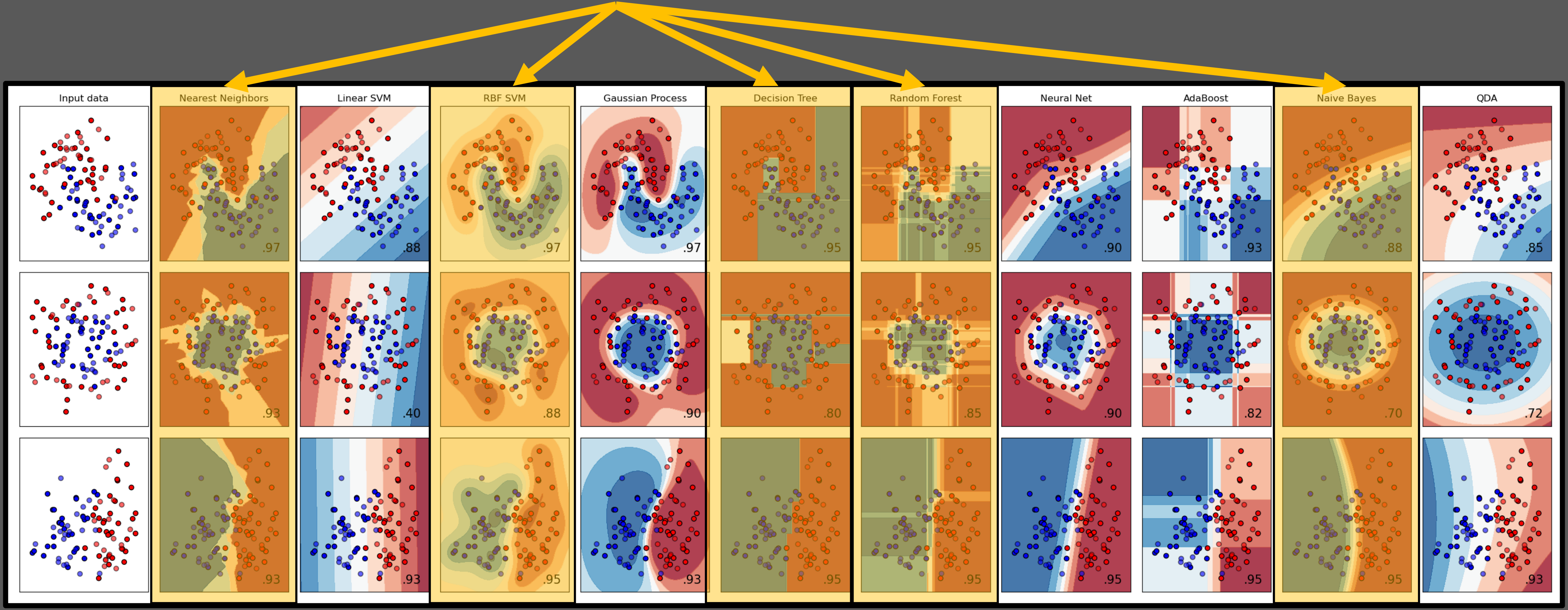


- No Multi-collinearity!
- Low Correlations
- 253 features
- Ignore the names as the repeat.
- The Target feature is the last column, and row
- Built using pandas-ta

$$Target = \begin{cases} 1\ if\ \left\{\dfrac{EMA(S\&P500)_{i+30} - EMA(S\&P500)_i}{EMA(S\&P500)_i}\right\} > 0.01 \\ -1\ if\ \left\{\dfrac{EMA(S\&P500)_{i+30} - EMA(S\&P500)_i}{EMA(S\&P500)_i}\right\} < -0.01 \\ 0\ if\ else \end{cases}$$

- **EMA is smooth!**
- **3 Classes [-1, 0, 1]**
- **10% Tolerance build in!**

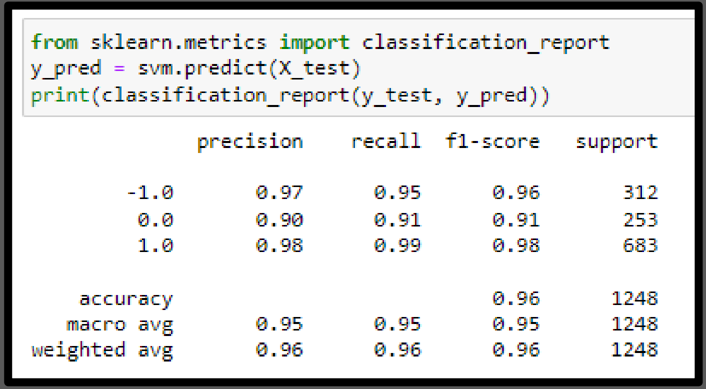
Models Evaluated



https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html#sphx-glr-auto-examples-classification-plot-classifier-comparison-py

Model Performance: SVM

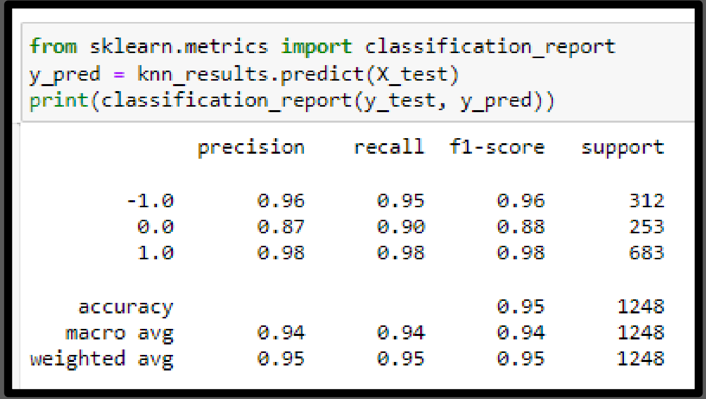
- Support Vector Machine Classifier (SVC)
- RBF Scored 98% on training



```
from sklearn.metrics import classification_report
y_pred = svm.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

        -1.0       0.97      0.95      0.96       312
         0.0       0.90      0.91      0.91       253
         1.0       0.98      0.99      0.98       683

    accuracy                           0.96      1248
   macro avg       0.95      0.95      0.95      1248
weighted avg       0.96      0.96      0.96      1248
```

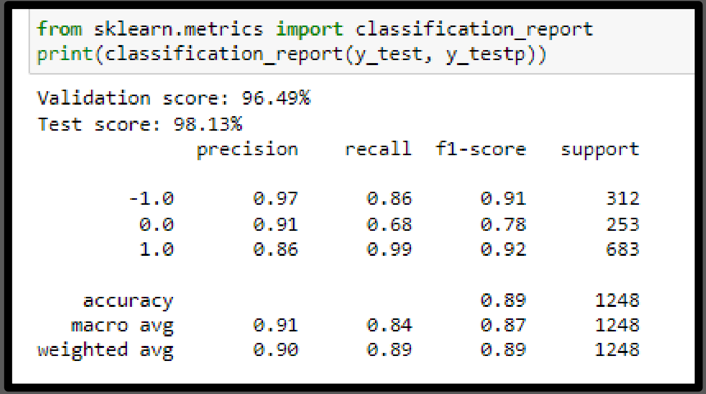
- 4-Nearest Neighbors (KNN)
- Scored 99% on training



```
from sklearn.metrics import classification_report
y_pred = knn_results.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

        -1.0       0.96      0.95      0.96       312
         0.0       0.87      0.90      0.88       253
         1.0       0.98      0.98      0.98       683

    accuracy                           0.95      1248
   macro avg       0.94      0.94      0.94      1248
weighted avg       0.95      0.95      0.95      1248
```

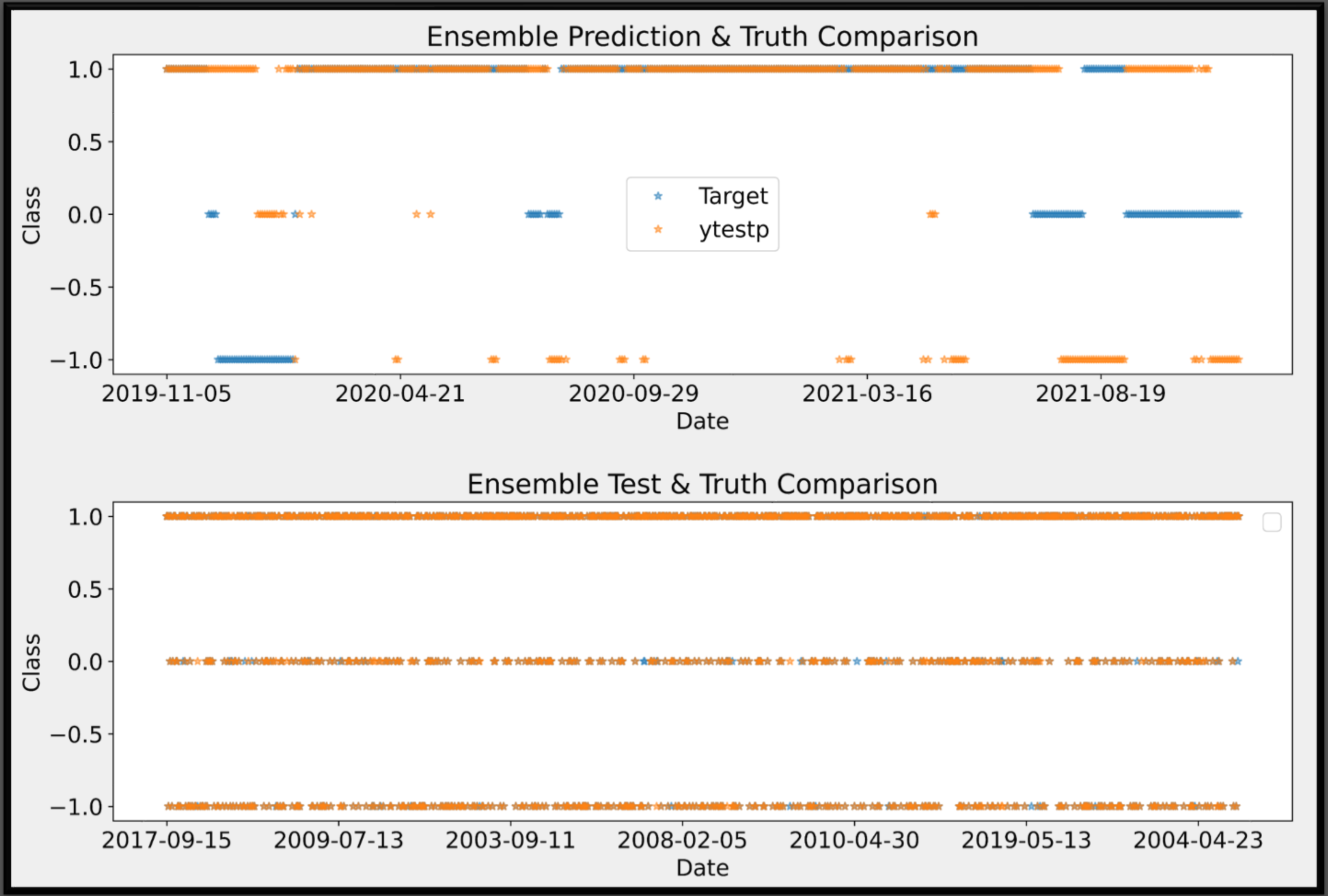
- Random Forest (RF)
- PCA: 20, Depth: 20
- Scored 96% on training



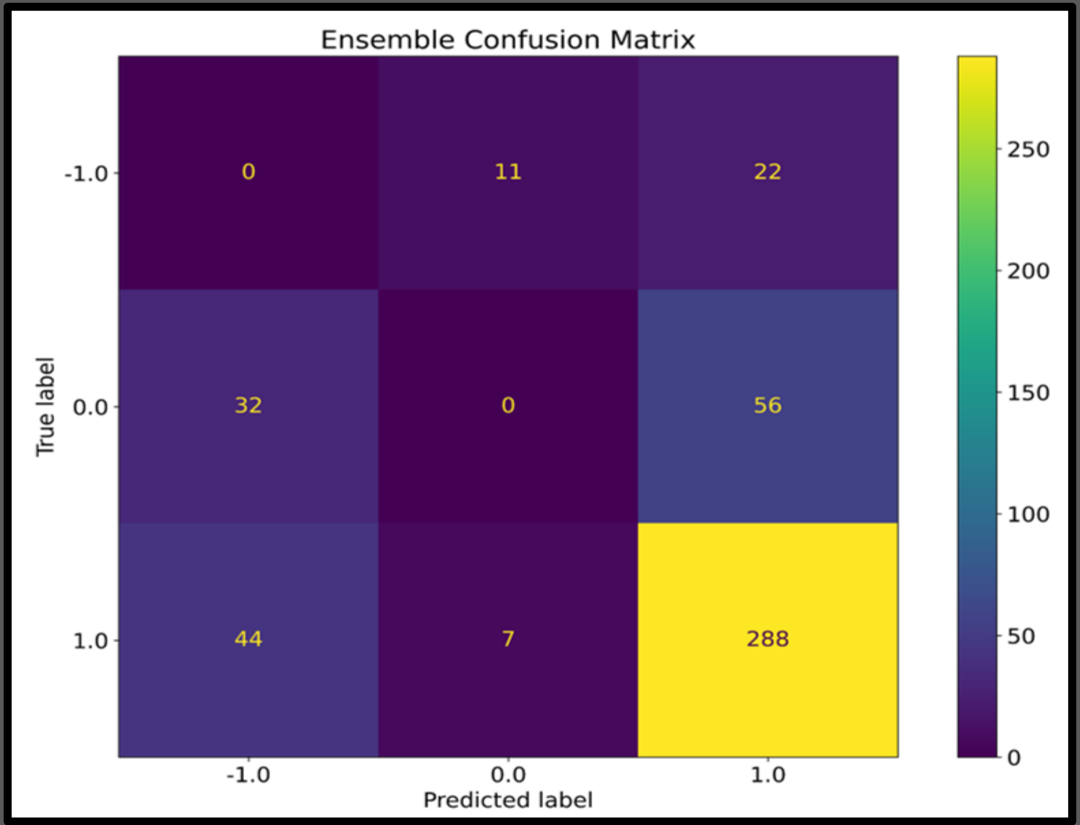
```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_testp))
```

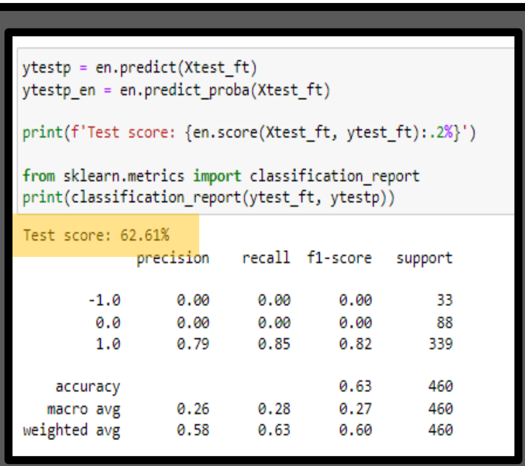
```
Validation score: 96.49%
Test score: 98.13%
              precision    recall  f1-score   support

        -1.0       0.97      0.86      0.91       312
         0.0       0.91      0.68      0.78       253
         1.0       0.86      0.99      0.92       683

    accuracy                           0.89      1248
   macro avg       0.91      0.84      0.87      1248
weighted avg       0.90      0.89      0.89      1248
```

Model Performance: Ensemble



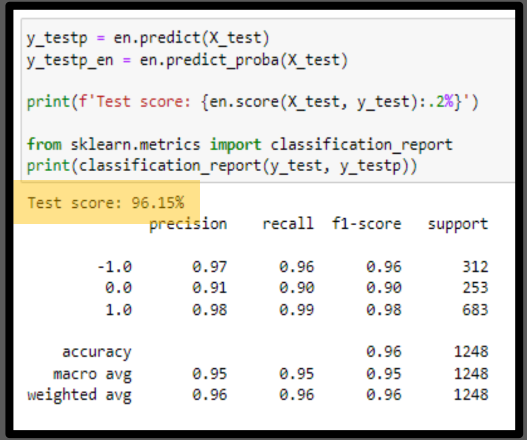
- The Ensemble seem pretty good at predicting increases in S&P500. Not surprising since the past year has been increasing.
- Testing on the "Past" Data was excellent at 96% while the test on "prediction" data was at 62%. Better than a coin toss!





```
ytestp = en.predict(Xtest_ft)
ytestp_en = en.predict_proba(Xtest_ft)

print(f'Test score: {en.score(Xtest_ft, ytest_ft):.2%}')

from sklearn.metrics import classification_report
print(classification_report(ytest_ft, ytestp))
```

```
Test score: 62.61%
              precision    recall  f1-score   support

        -1.0       0.00      0.00      0.00        33
         0.0       0.00      0.00      0.00        88
         1.0       0.79      0.85      0.82       339

    accuracy                           0.63       460
   macro avg       0.26      0.28      0.27       460
weighted avg       0.58      0.63      0.60       460
```



```
y_testp = en.predict(X_test)
y_testp_en = en.predict_proba(X_test)

print(f'Test score: {en.score(X_test, y_test):.2%}')

from sklearn.metrics import classification_report
print(classification_report(y_test, y_testp))
```

```
Test score: 96.15%
              precision    recall  f1-score   support

        -1.0       0.97      0.96      0.96       312
         0.0       0.91      0.90      0.90       253
         1.0       0.98      0.99      0.98       683

    accuracy                           0.96      1248
   macro avg       0.95      0.95      0.95      1248
weighted avg       0.96      0.96      0.96      1248
```

A Soft Voting Ensemble Classification model comprised of an SVM, KNN, and RF was created to generate S&P 500 futures using handpicked! 62% of last few years.

- Better than a coin toss!
- Simple Implementation!
- Excellent Learning Experience!
- <u>Next Steps!!!</u>
  - Needs more tuning on Dataset & Models…
  - Needs another thorough look through…
  - Conversion into something deployable
  - Train for 14-Day and 7-Day forecasts