Predicting Quarterly S&P 500 Updates

By: Zubeir Said

The Problem

The S&P 500

- Index tracking the 500 best performing stocks listed on USA exchanges
- Every quarters there are companies who exit and those that enter based on specific criteria
- The Criteria (we care about)
    - Market Cap of certain size
    - Minimum 250,000 shares traded monthly over trailing 6 months
    - Publicly traded for minimum of a year
    - Positive sum of the previous 4 quarters of earnings
    - Positive sentiment

Significance Of Project

- Being able to predict future components helps us understand the trend of the index
- Knowing which companies will enter and leave lets us know how to handle our positions on those individual stocks
- We learn how much of an influence each particular criteria weighs on the decision making
- Ultimately we get to make more money

Problem Statement: Build a model that given specific criteria about a stock will be able to predict whether or not it will be part of the S&P 500

The Data

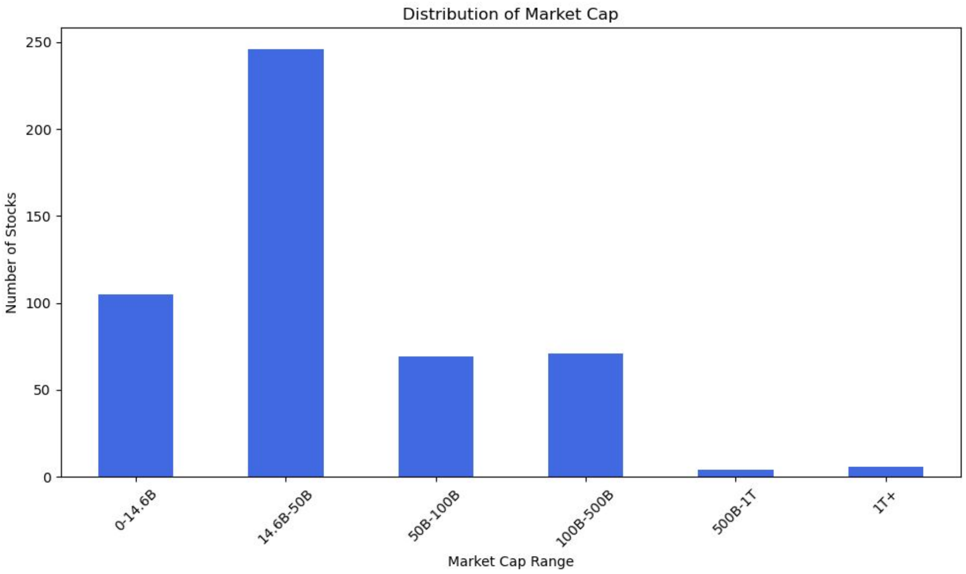
Stock Data

- 500 Stocks from the S&P 500 and 500 stocks from the bottom half of Russell 1000
- Features
    - Ticker
    - Market Cap
    - 8 quarters (2 years) of trailing Net Income
    - 24 months of trailing Monthly Volume
    - Outstanding Shares
    - Age of company

EDA

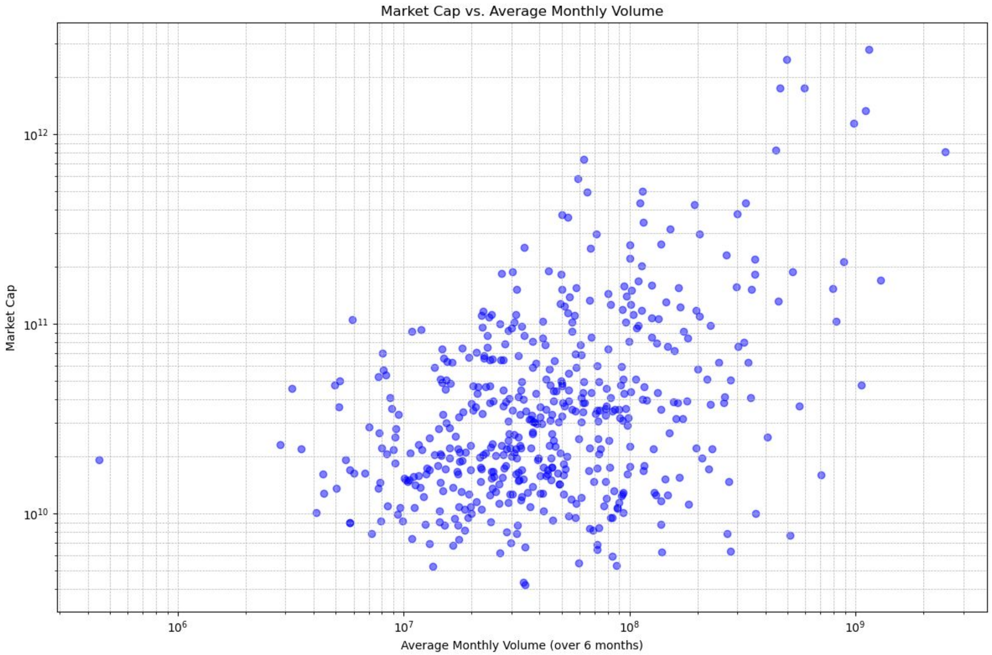
Market Cap Distribution

- Majority of companies are on the smaller end making them victim of leaving the index
- Meaning market cap alone may not be decisive enough as we thought



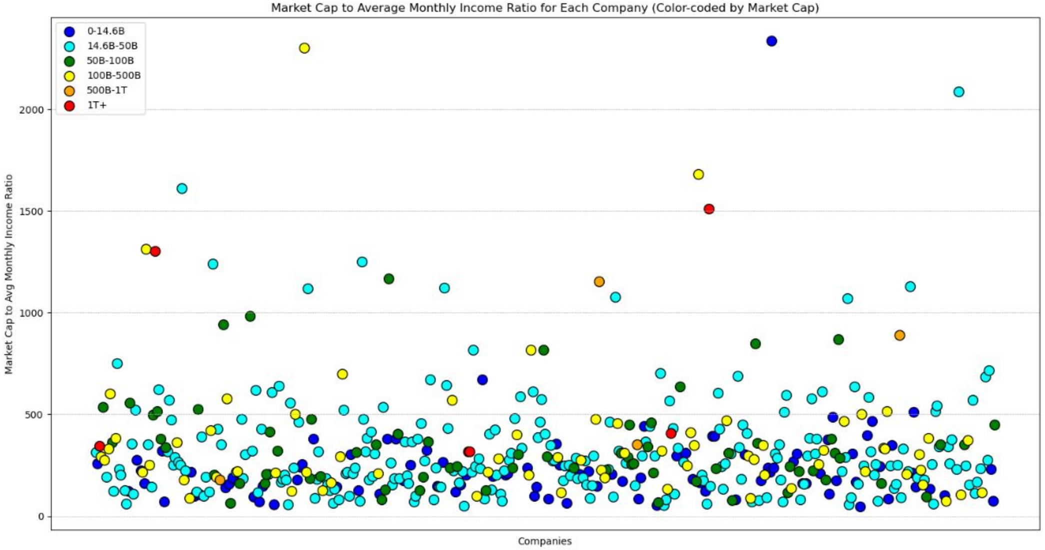
Smaller Insignificant companies are first to go

- The more traded the larger the market cap and vice versa
- Model should be able to distinguish that companies on the smaller and less traded end should be subject to potential removal



Market Cap to Monthly Income Ratio

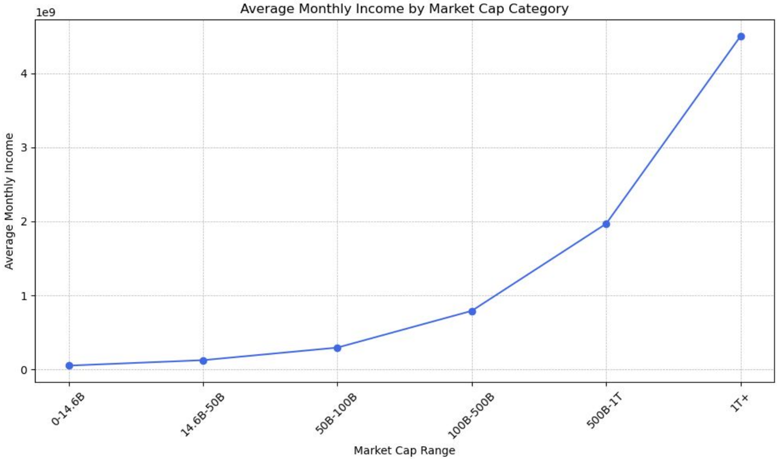
- Most companies are healthy but about 10% are over-inflated
- Meaning these will be the companies that trick our model
- This is an important relationship

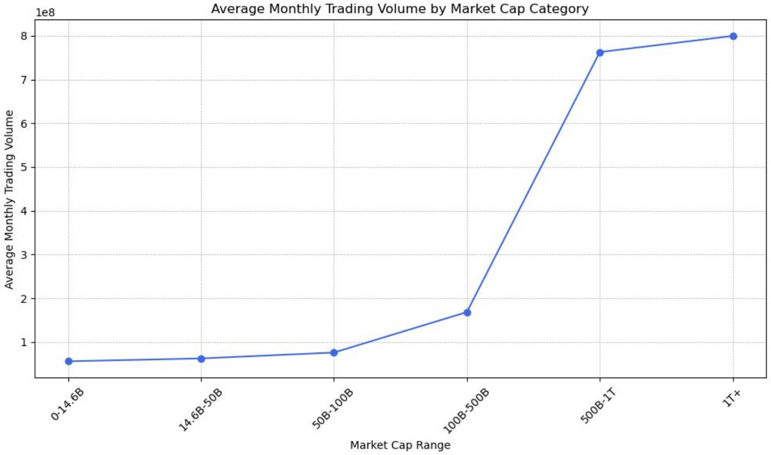


Market Cap Size behavior

The averages of each bucket of market size are performing as they should which is good for our model

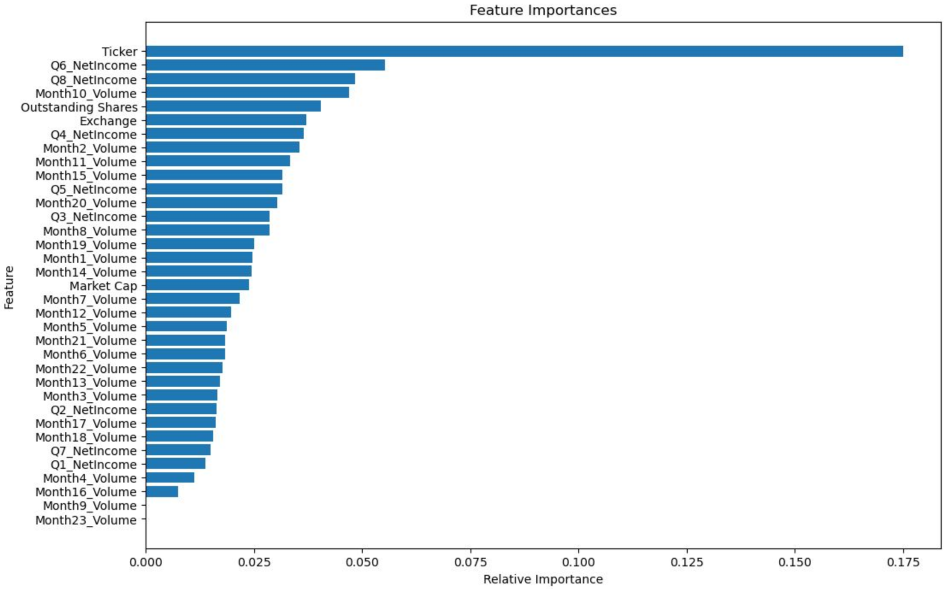
Significance: Strengthens the models belief that the larger you are all around the more important you are





Feature Importance To Target Variable

Doesn’t really tell the picture we expect but wanted to highlight since it'll come up again soon



Lessons Learned:

- The larger the stock the more critical it is to the index
- Averages are strong across company sizes but quite a few outliers
- Market Cap and Income tell the strongest story of weighting

The Model

Data Management

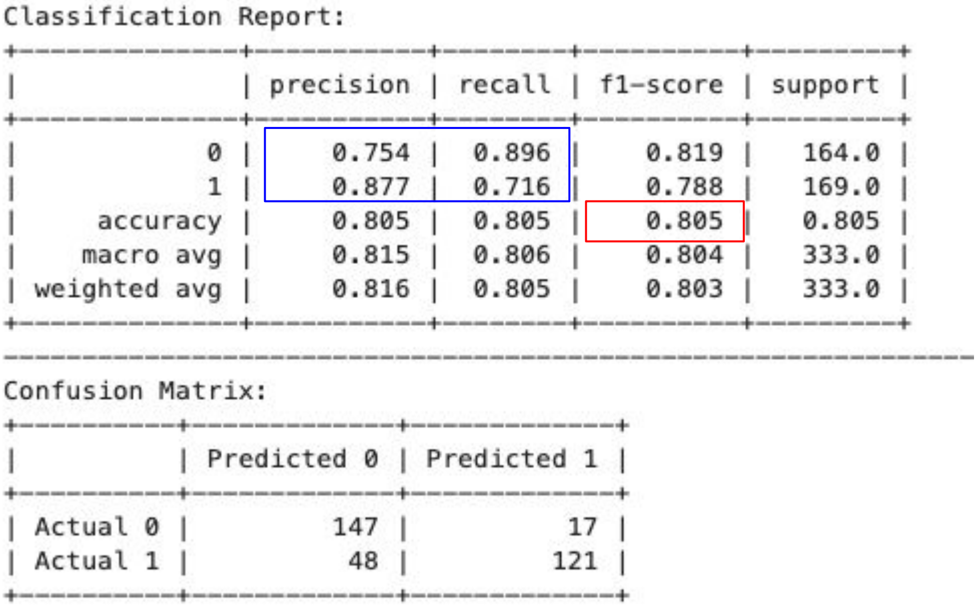
- Net Income
  - Nan values were set to 0
- Monthly Volumes
  - Nan values were set to 0
- Data was scaled
- Large Values were Imputed
- Data was managed for imbalances
- Features: All Numerical Columns were used
- Target Variable: “in_index” (Denotes whether a stock is in the S&P or not0

Classification Models Attempted

- Logistic Regression
- Random Forest Classifier
- Gradient Boosted Tree
- Neural Network
- Bagging Model

Performances: Logistic Regression

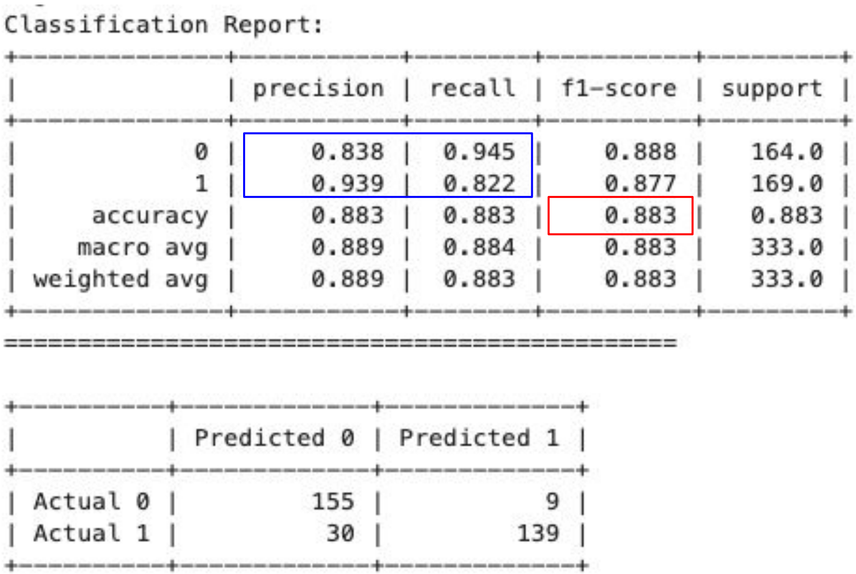
- We are optimizing for f1 score since we want a balance between the two precision and recall
- However we understand if we forced to pick one than it would be recall.



```
Classification Report:
+--------------+-------------+----------+------------+-----------+
|              | precision   | recall   | f1-score   | support   |
+--------------+-------------+----------+------------+-----------+
|            0 |       0.754 |    0.896 |      0.819 |     164.0 |
|            1 |       0.877 |    0.716 |      0.788 |     169.0 |
|     accuracy |       0.805 |    0.805 |      0.805 |     0.805 |
|    macro avg |       0.815 |    0.806 |      0.804 |     333.0 |
| weighted avg |       0.816 |    0.805 |      0.803 |     333.0 |
+--------------+-------------+----------+------------+-----------+
----------------------------------------------------------------------
Confusion Matrix:
+----------+---------------+---------------+
|          | Predicted 0   | Predicted 1   |
+----------+---------------+---------------+
| Actual 0 |           147 |            17 |
| Actual 1 |            48 |           121 |
+----------+---------------+---------------+
```

Performances: Random Forest Classifier

- Best parameters:
    - N-Estimators:
    - Max Depth:
    - Min Samples Split:
    - Min Samples Leaf:
    - Max Features:
- 5 KFold



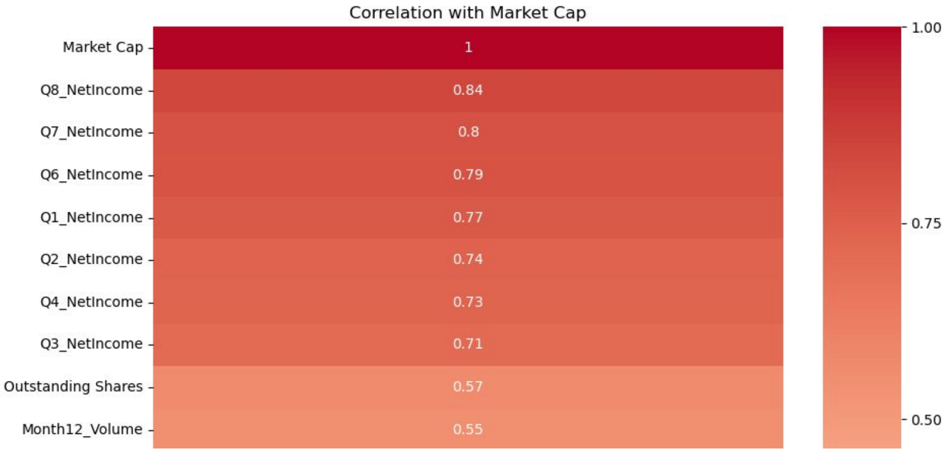
```
Classification Report:
+--------------+-----------+---------+----------+---------+
|              | precision | recall  | f1-score | support |
+--------------+-----------+---------+----------+---------+
|            0 |     0.838 |   0.945 |    0.888 |   164.0 |
|            1 |     0.939 |   0.822 |    0.877 |   169.0 |
|     accuracy |     0.883 |   0.883 |    0.883 |   0.883 |
|    macro avg |     0.889 |   0.884 |    0.883 |   333.0 |
| weighted avg |     0.889 |   0.883 |    0.883 |   333.0 |
+--------------+-----------+---------+----------+---------+
==================================================

+----------+-------------+-------------+
|          | Predicted 0 | Predicted 1 |
+----------+-------------+-------------+
| Actual 0 |         155 |           9 |
| Actual 1 |          30 |         139 |
+----------+-------------+-------------+
```

Model Intermission

- As I played with hyperparameters and the parameter grid I was having little to no change in performance
- Dropping an singular feature would make all models perform widely worse
  - We need more data not less
- Heavy Weighted Features through feature interaction:
  - Market Cap, Q8 Net Income, Q7 Net Income

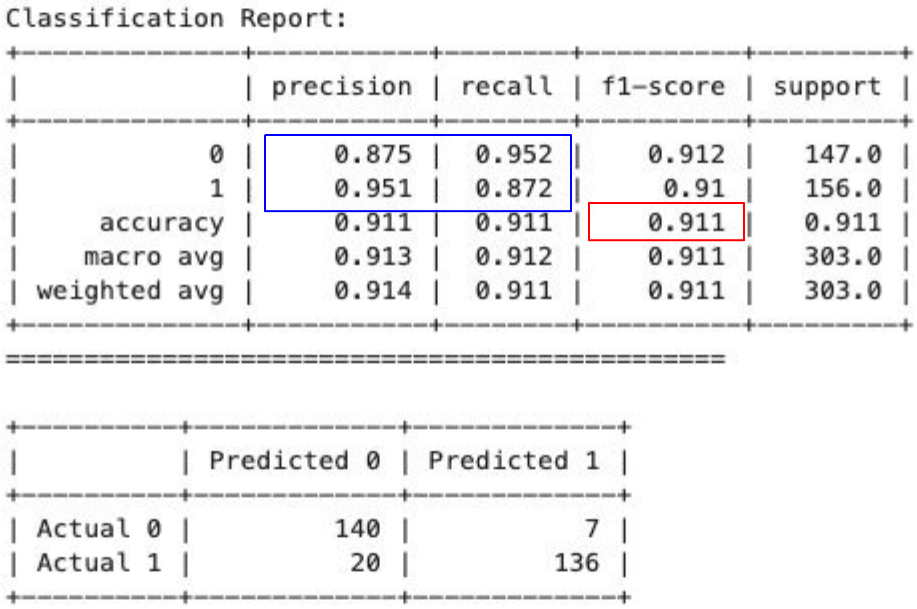
Market Cap Correlation

- We know Market Cap is the truest leading indicator of being in the S&P 500
- As expected trailing income has the highest correlation so we will do Feature Interaction



Performances: Gradient Boosted Tree (Winner)

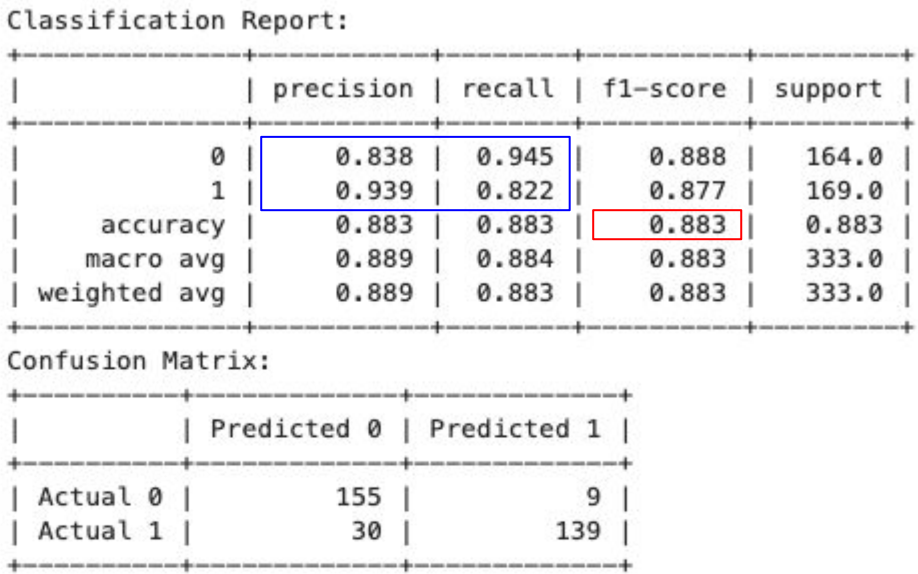
- Feature Correlations
  - Market Cap * Market Cap
  - Market Cap * Q7
  - Market Cap * Q8
- Features
  - Learning Rate: 0.1
  - Max Depth 3
  - N-Estimators: 50
  - Tree Sample: 0.8



```
Classification Report:
+--------------+-----------+--------+----------+---------+
|              | precision | recall | f1-score | support |
+--------------+-----------+--------+----------+---------+
|            0 |     0.875 |  0.952 |    0.912 |   147.0 |
|            1 |     0.951 |  0.872 |     0.91 |   156.0 |
|     accuracy |     0.911 |  0.911 |    0.911 |   0.911 |
|    macro avg |     0.913 |  0.912 |    0.911 |   303.0 |
| weighted avg |     0.914 |  0.911 |    0.911 |   303.0 |
+--------------+-----------+--------+----------+---------+
==================================================

+-----------+-------------+-------------+
|           | Predicted 0 | Predicted 1 |
+-----------+-------------+-------------+
| Actual 0  |         140 |           7 |
| Actual 1  |          20 |         136 |
+-----------+-------------+-------------+
```

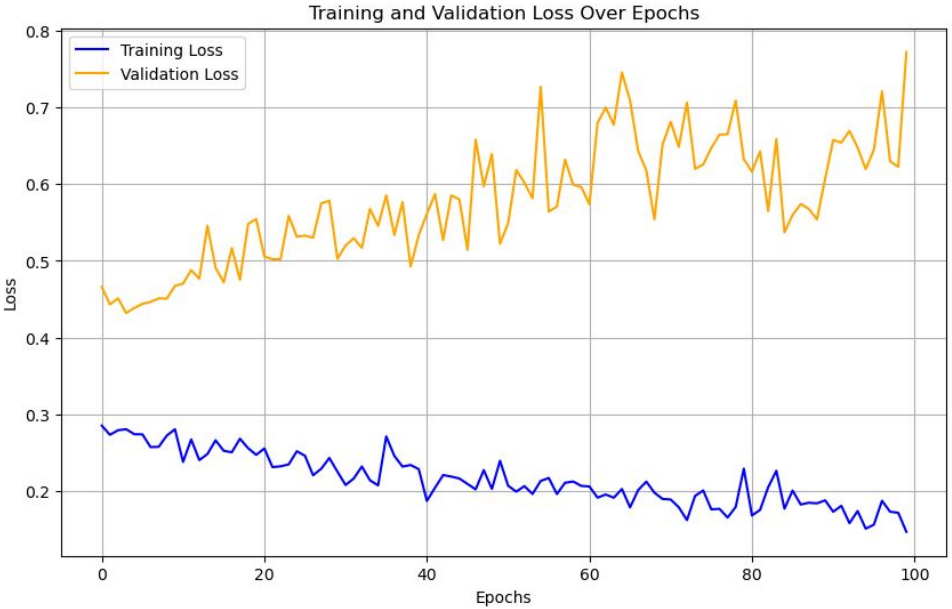
Performances: Neural Networks

- Parameters:
  - Learning rate: 1e-4
  - L2 Regularization
  - Dropout
  - Early Stopping
  - 5 Layers
- Stopped at 56/100 epochs
- Severe Overfitting to less results

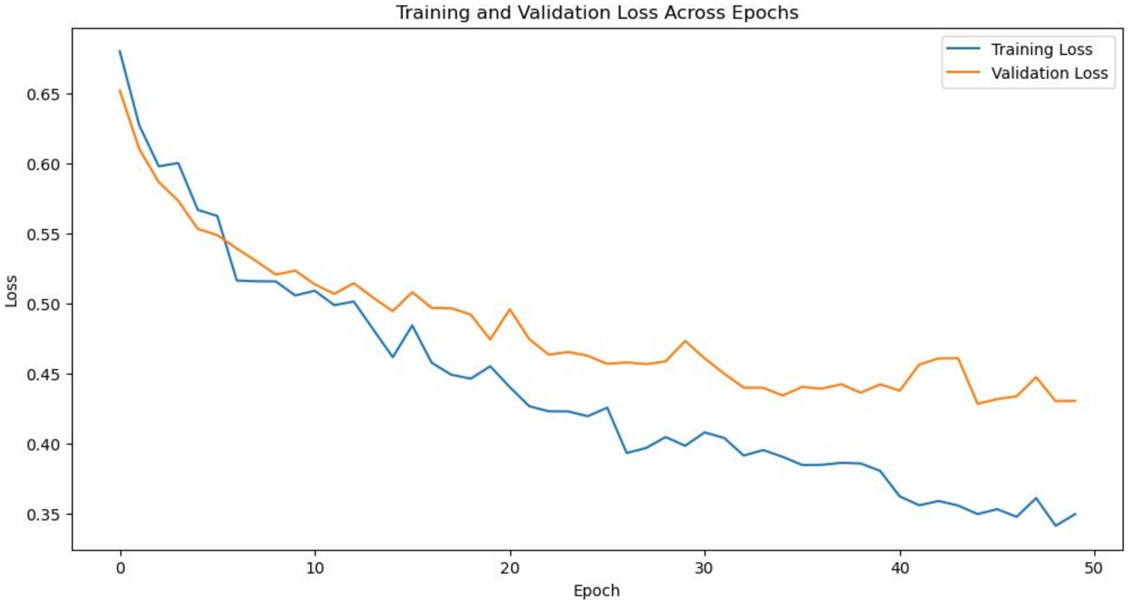


```
Classification Report:
+--------------+-----------+----------+-----------+----------+
|              | precision |   recall |  f1-score |  support |
+--------------+-----------+----------+-----------+----------+
|            0 |     0.838 |    0.945 |     0.888 |    164.0 |
|            1 |     0.939 |    0.822 |     0.877 |    169.0 |
|     accuracy |     0.883 |    0.883 |     0.883 |    0.883 |
|    macro avg |     0.889 |    0.884 |     0.883 |    333.0 |
| weighted avg |     0.889 |    0.883 |     0.883 |    333.0 |
+--------------+-----------+----------+-----------+----------+
Confusion Matrix:
+----------+-------------+-------------+
|          | Predicted 0 | Predicted 1 |
+----------+-------------+-------------+
| Actual 0 |         155 |           9 |
| Actual 1 |          30 |         139 |
+----------+-------------+-------------+
```

Graph of Training and Validation Loss



Best Graph of Loss But Only 80% Accuracy



Lessons Learned:

- XGB is the best Model by far
- The best were going to get is 91% accuracy unless we obtain more financial data
- We need to give more weight to more critical features

Conclusion

- When assessing model performances we should look to see which stocks actually got misclassified and examine their features to see why the model got it wrong
- Not Included in SP but should
  - AXON - 16B
  - TYL - 16B
  - MTD - 23B
  - RCL - 22B
- Included in SP but shouldn't
  - IBKR - 33B
  - ICLR - 18B
  - LPLA - 17B

Conclusion

- Favoring Market misses some big ones but produces the best results
  - Doesn't miss the mega ones
- We have a high recall when we want to ask if a stock will end up in the SP in the future
- This model is highly recommendable when wanting to know if a company is probable to be in the SP 500
- Model is okay when it comes to accurately depicting the entire SP as a whole which is fine

Thank You!