COMP9417 MACHINE LEARNING

# MLB Player Digital Engagement Forecasting

Akeef Mamoon - z5114627
Daksh Mukhra - z5163002
Yvonne Thich - z5206070

August 1, 2021

# 1 Introduction

Recent technological advancements and the proliferation of social networking have transformed the media landscape and how sports companies are engaging with their fans. In light of the pandemic, there has been a stronger push by sporting executives towards creating deeper relationships between players and fans through digital engagement. In this project, we shall delve into "America's favourite pastime", Major League Baseball (MLB).

As MLB is heavily data driven, it is a great opportunity to apply machine learning to yield insights into what constitutes digital popularity. More information on this challenge can be found in MLB Player Digital Engagement Forecasting on Kaggle (Google Cloud/ Major League Baseball, 2021).

The purpose of this challenge is to generate insights into what factors influence online activity to predict four measures of fan engagement using baseball player digital content. Although previous work has identified some catalysts to baseball fan engagement, this challenge delves into richer datasets at a more granular level in an attempt to find strong correlations to digital engagement on and off the field.

Our approach taken to address this problem is as follows. We shall conduct data exploration to gain high level insights into factors which pique supporter interest and then build a set of time series models to forecast next day player engagement. The models used in this study are Light GBM, XGBoost, Vector Autoregression and Autoregressive Neural Network.

# 2 Data Exploration

## 2.1 Data Pre-processing

| Files | Description |
|---|---|
| train | A training set containing data on MLB players active at some point since 2018 in nested JSON fields. Predictions are only scored for those players active in 2021, but previous seasons' players are included to provide more data exploration and modelling purposes. The nested JSON fields are: nextDayPlayerEngagement, games, rosters, playerBoxScores, teamBoxScores, transactions, standings, awards, events, playerTwitterFollowers, teamTwitterFollowers |
| awards | A collection of awards given out prior to 01/01/2018 (the first date in train.csv) |
| players | A library containing high level information about all MLB players in this dataset |
| seasons | Information about start and end dates of all seasons in this dataset |
| teams | A library containing high level information about all MLB teams |

*Descriptions taken from MLB Player Digital Engagement Forecasting from Kaggle*

Prior to any data exploration, we unpacked the nested JSON files in train.csv and merged all the fields into a master table. A noteworthy field is *nextDayPlayerEngagement* as it contains *engagementMetricsDate, playerId, target1, target2, target3* and *target4*. Our aim is to predict the target predictions for each *playerId* on given engagement metric dates. Our dataset has a total of 2,506,176 observations and 72 features.

## 2.2 Targets

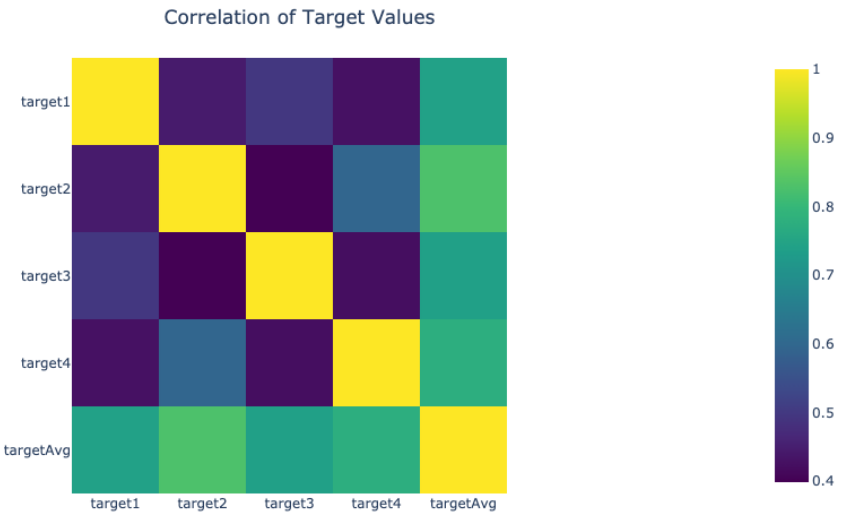



Figure 2.1: Heatmap of Target Values

As the four targets of engagement are not explicitly specified, we shall first explore how each target variable varies from one another. Figure 1.a in the appendix shows significant seasonality, with all target values peaking in March and dipping in October. As MLB is generally scheduled from early April to early October, this observation implies that fan engagement is affected by on and off season periods. By analysing the Pearson correlation in Figure 2.1, we can see that the target variables have a medium positive correlation with each other. We can also note that Target 2 tends to move more closely with Target 4. However, as there are a high number of outlier values (refer to Figure 1.b in the appendix), the Pearson correlation outputs may be slightly distorted. Since all four targets have a strong positive correlation with the target average, we shall use the target average value in the rest of our analysis.

## 2.3 Top Player by Position

Looking at the breakdown of top 100 players by position (Figure 2.a in appendix), 30% of these players were Outfielders and approximately another 30% were pitchers, which shows that these players are more likely to score higher compared to the rest of the team players on engagement score metric. Although pitchers make up 30% of the top 100 players (Figure 2.a in the appendix), looking at the average target scores for the whole data set (Figure 2.b in the appendix), the score for pitchers is actually quite low and much behind several other positions. This indicates that only a few key pitchers have a relatively high engagement (Figure 2.c in the appendix). We can also see that the average score for outfielders remains high which is consistent with the top100 players. Although designated hitters have a high target average, this is skewed as only 6 designated hitters are present in the entire data set and it would be

unwise to draw conclusions from such a sample. From the findings above, it is clear that player position will play a part in predicting engagement metrics as players in positions such as outfielders and First base correlate with high engagement metrics.

### 2.4 Player Demographics

It is also noticed that based on the player's country of birth, they could have a high or low level of importance. For example, if a country, like Netherlands, only had 2 MLB players, then the level of engagement for those 2 players would be higher than any of the 100 USA players as there would be a higher density of fans per player.

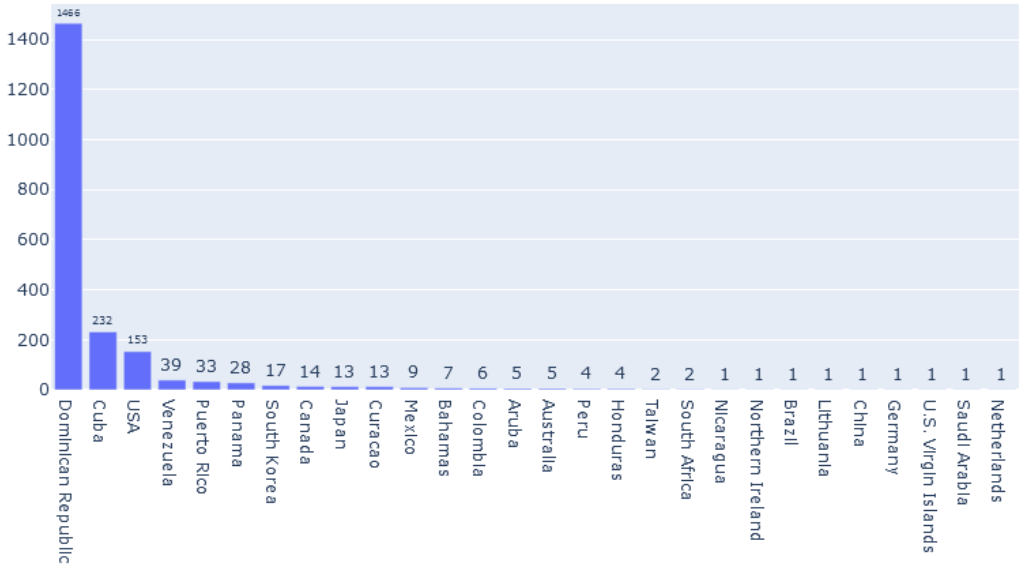



Figure 2.2: Number of unique occurences of countries of birth in the dataset

## 3 Evaluation Metric

Our models shall be evaluated on the mean column-wise mean absolute error (MCMAE). Mean absolute error (MAE) measures the errors between paired observations expressing the same event (Wikipedia 2021). A mean absolute error is calculated for each of the four target variables and the score is the average of those four MAE values.

$$MCMAE = \frac{\sum_{i=1}^{4}\sum_{j=1}^{n}|y_{ij} - \hat{y}_{ij}|}{4n}$$

where $\hat{y}_{ij}$ is the prediction for the $i^{th}$ target, $y_{ij}$ is the true value and $n$ is the number of observations.

# 4 IMPLEMENTATION OF MODELS

## 4.1 VECTOR AUTOREGRESSION

Vector Autoregression (VAR) is a simple bi-directional model used in multivariate forecasting. The predictions are made using ordinary least squares under the assumption that a future value is a result of historical patterns of itself and other features, with all variables treated as endogenous (Hushani 2018, pg1). Thus VAR captures the intertwined dynamics of time series data which aligns to the purpose of this project. VAR makes the predictions as follows:

$$\hat{\mathbf{y}}(\mathbf{t}) = \beta_{\mathbf{0}} + \sum \beta_{\mathbf{i}} \mathbf{y}(\mathbf{t} - \mathbf{i}) + \mathbf{e}(\mathbf{t})$$

where $\hat{y}$ is the forecast at time $t$, $\beta$s are the model coefficients and $e(t)$ is the noise component (Hushani 2018, pg2).

As follower count and measures of social media engagement are bi-directionally linked and number of player Twitter followers has a stronger positive correlation with engagement (refer to Section 3.c in the appendix), we shall use the VAR model to further explore its ability to predict our four measures of baseball player engagement. It is interesting to note that despite being a popular time series forecasting method, the VAR model was not a common approach for this challenge.

### 4.1.1 PRE-PROCESSING

| | target1_x | target2_x | target3_x | target4_x | numberOfFollowers_x |
|---|---|---|---|---|---|
| target1_y | 1.0000 | 0.0000 | 0.0206 | 0.0027 | 0.0 |
| target2_y | 0.0003 | 1.0000 | 0.00010 | 0.0000 | 0.0 |
| target3_y | 0.0218 | 0.0000 | 1.0000 | 0.0000 | 0.0 |
| target4_y | 0.0000 | 0.0166 | 0.0018 | 1.0000 | 0.0 |
| numberOfFollowers_y | 0.0000 | 0.0000 | 0.0000 | 0.0002 | 1.0 |

Table 4.1: Matrix of p-values checking the Granger Causality of all possible combinations of the time series

For this model, we used data, represented by the target engagement scores and total number of player Twitter followers between January 2018 and April 2021, a total of 1187 days. From Table 4.1, as the p-values of the response variables and the predictors all fall below the significance level of 0.05, we can deduce that all variables are statistically significant to our predictions. To avoid data leakage, the model was trained on data up until 31 March 2021 and tested on the April 2021 data. Section 4.e in the appendix details the removal of seasonal fluctuations from our data to promote stationarity and improve the accuracy of our prediction model. The motive behind this is that if a time series has stationary behaviour over a given period, then it can be assumed that the time series will behave similarly at a later point in time (Monigatti 2021).

### 4.1.2 Criticisms

One of the greatest criticisms of VAR is that the model is atheoretical. As every variable is assumed to influence every other variable in the system, there is a limitation to our feature selection (Hyndman 2018, Section 11.2). For example, whilst the number of home runs scored by a player increases digital engagement, fan's social media activity does not Granger causes the number of home runs scored, and thus cannot be included in the model. This may potentially lead to less valuable insights into what factors influence digital engagement. Another shortcoming of the VAR model is the 'curse of dimensionality'. Our model has 1925 parameters not including the constant terms and error variance-covariances (refer to Section 6.a in the appendix). As many parameters have to be estimated, this increases the risk of overfitting the model, in turn generating more inaccurate predictions out of sample (Hyndman 2018, Section 11.2).

## 4.2 Gradient Boosting Decision Tree Models

GDBTs are popular machine learning algorithms, particularly in regards to machine learning competitions like those hosted on Kaggle (Mwiti, 2021). This is predominantly due to the efficiency, accuracy and interpretability which it achieves in many different machine learning tasks. Traditional implementations of GBDTs include algorithms like XGBoost and AdaBoost, with the former being used here in the challenge. However, these algorithms often become computationally complex with a large number of features and instances, making it very time consuming when it comes to large datasets.

GBDTs face challenges in regards to large datasets, being forced to make a choice between accuracy and efficiency. Later models like CatBoost and LightGBM have found different ways to address these issues.

### 4.2.1 Extreme Gradient Boosting

XGBoost or Extreme Gradient Boosting is a linear model that uses parallel computations to do tree learning, making it another adept option for machine learning. XGBoost uses internal buffers on each of the parallel threads, making for more hardware efficient use.

With the large feature space in the MLB Competition, XGBoost's parallelisation of expansion for tree learning makes running the model much more efficient in terms of time and space compared to some other models like Random Forest or VAR.

### 4.2.2 Light Gradient Boosting Machine

LightGBM or the Light Gradient Boosting Machine is a gradient boosting framework that uses tree based learning algorithms. Despite most decision tree algorithms growing trees by level (depth-wise), LGBM grows leaf-wise or best-first.

To address the issues with GBDT, two techniques have been implemented in LightGBM:

1. **Gradient-based One-Side Sampling (GOSS):**
   This is where instances with large gradients are kept, in order to maintain accuracy, while those with smaller gradients are dropped randomly.

2. **Exclusive Feature Bundling (EFB):**
   To address sparse feature spaces, exclusive features, features that rarely take nonzero values simultaneously, are bundled together.

The LightGBM model has a large number of parameters, which can be found in the documentation.

LightGBM was considered the standard approach for this challenge by most, and achieved good results with even a basic feature space. Wee deviated from the normal lightGBM approach by expanding the feature set and using optuna to tune the parameters for the model as opposed to using a lags based method where there was the risk of over-fitting and data leakage.

#### 4.2.3 GBDT Feature Engineering

The dataset possessed a large amount of data of varied types. In order to use LightGBM effectively the data had to be processed and presented in the form of a float, int or a boolean and due to the large feature space of the dataset, a large amount of feature selection was also required. in order to reduce the overall time and space complexity as well as to prevent overfitting. To do this, we used the Pearson correlation score to measure non-categorical, numerical variables, label encoding for non-numerical, non-categorical variables as well as processing for other features in order to translate them into numerical variables that were capable of being read by the model implementation. This processing included:

- Reducing the large numerical variables like *numberOfFollowers* into more reasonable proportional floats
- Label encoding for string variables to measure the number of unique times it pops up in the dataset
- Further expansion of some features like *mlbDebutDate* to find out things like how long the player has been in the league, how old the player is etc.

### 4.3 Auto Regressive Neural Network

The multilayer perceptron neural network is a supervised learning technique where information only travels forward in the network. The inputs to each node are combined using a weighted linear combination and the result is then modified by a nonlinear function before being outputted to another node. These weights are optimized using a backpropagation algorithm to minimize a loss function. An Autoregressive Neural Network (ARNN) builds upon this by passing lagged values of the time series data into a network and using it to predict future values. We shall be using this model to predict social media engagement scores.

#### 4.3.1 PRE-PROCESSING

As the data provided is temporal, it brings into play issues that can skew predictions such as seasonality (non-stationary data). To counter this, the first difference was taken for each target, effectively detrending the data (appendix Figure 4.a before, 4.c after). We then proceeded to create 3 data sets to test 3 variations of the (ARNN):

- ARNN with 5 lags of each target as features
- ARNN with 20 lags as features
- ARNN with 5 lags and additional player box score metrics as features.

To motivate the selection of 5 lags, a plot (see Figure 3.a in the appendix) was constructed showing the autocorrelation distribution for player target averages across different lag values. Here, lag means are reasonably high until lag 5 but then reduces significantly afterwards. 20 lags were selected as reference of comparison. Player box scores such as *hits, homeRuns* (see Figure 2.d in appendix) and *strikeOutsPitches , total Bases and plate appearances*, have all shown to be strongly correlated with engagement (see Table 3.a in appendix). This correlation is expected as better player performance will usually result in more commentary and digital engagement surrounding that performance. We shall use the ARNN model to further explore these features' predictive power. After constructing each data set, each was split into train and tests, with the use of time series split to avoid data leakage issues.

#### 4.3.2 CONSTRUCTION OF NEURAL NETWORKS

All three models consist of an input layer containing the nodes that represent the different features for each model, 3 hidden layers and an output layer representing the 4 targets. Regularization in the form of dropout layers and Batch normalization was used to counter overfitting, with the ReLU activation function and MAE for loss. Each model was implemented using Keras, a high level framework based on Tensorflow and was trained using K-fold cross validation.

#### 4.3.3 HYPER-PARAMETER TUNING

Tuning the parameters is necessary in order to find the optimal model. This was done in a three stage process where we tuned the number of nodes in hidden layers, the learning rate and dropout rate of the network. Each hyperparameter was tuned in a method similar to GridSearchCV, where we optimised for the loss of the network. It was found that the best combination of hyper parameters were 80 hidden nodes, with a learning rate of 0.001 a drop out rate of 0.2.

#### 4.3.4 CRITICISMS

Although the set of ARNN's produced comparable results, one of the biggest problems is that there is no quantitative way to measure one feature as having a greater predictive power then another. As a neural network is inherently Blackbox in nature, there are issues with interpretability.

# 5 RESULTS

## 5.1 VAR RESULTS

Various method validation techniques were applied to determine the best lag to fit the VAR model. The prediction using the best lag was then compared with the test data set to obtain a MCMAE score. Table 7.a in the appendix summarises the results. Table 5.1 below illustrates the results of the final VAR model.

| VAR Results | |
|---|---|
| Target | Mean MAE |
| target 1 | 1.461469808915275 |
| target 2 | 6.155860832555195 |
| target 3 | 1.8644215073337407 |
| target 4 | 2.2729689118121925 |
| MCMAE | 2.938680265154101 |

Table 5.1

The maximum lag order for our model was 100 lags. This number was selected by considering the trade off between having more degrees of freedom, which could increase the risk of overfitting, and specification errors from using too few lags (Bosede Ngozi 2018). The lag order of 77 was chosen based on the minimum Akaike Information Criterion (Table 5.a in the appendix) as that produced the best MCMAE score (refer to Table 8.a in appendix). After data preprocessing and tuning, our final VAR model achieved a MCMAE score of 2.94.

By analysing the weights of our final model, we can deduce that the lags of a target value is the most significant in forecasting digital engagement for that specific target, and that more recent lags are favoured. It is interesting to note that *target4*'s earlier scores play a bigger role in predicting next day *target2* values. This aligns with our findings in Section 2.2. Another interesting note is that whilst the number of player Twitter followers does not heavily influence target predictions, it is more meaningful than using other target scores.

## 5.2 GBDT RESULTS

GDBT algorithms seemingly reached better results than the VAR model. This could be a direct result of the number of features that GDBT algorithms can make use of before becoming computationally unwieldy or as a result of the forecasting of this dataset.

To test the models, we used k-fold cross-validation using the implementation from Sklearn. As this is a time-series forecasting challenge, if we used cross-validation as a form of hyper parameter tuning, there is the possibility of overfitting (Cerqueira, V., Torgo, L. and Mozetič, I., 2020), so hyper parameter tuning was done using Optuna to minimise the MAE of the models. K-fold cross validation is used here to assess the results of the hyper parameter tuning results

from Optuna in an attempt to further tune the parameters to get a better result.

| LightGBM Results | |
|---|---|
| Target | Mean MAE |
| target 1 | 0.7624581589322104 |
| target 2 | 2.8817678527825126 |
| target 3 | 0.884181540970667 |
| target 4 | 1.1617746169383174 |
| MCMAE | 1.42254554 |

Table 5.2

| XGBoost Results | |
|---|---|
| Target | Mean MAE |
| target 1 | 0.9269265555141777 |
| target 2 | 2.5753980264780862 |
| target 3 | 1.2614307652565837 |
| target 4 | 1.2356987949187161 |
| MCMAE | 1.49986354 |

Table 5.3

From the cross validation results in Table 5.2 and 5.3, LightGBM seems to have a slightly more accurate model, but this may be due to the fact that there is no native objective function for XGBoost to calculate MAE and RMSE or a separate implementation of MAE has to be used.

## 5.3 Autoregressive Neural Network Results

| ARNN K-Fold Cross Validation Results | | | |
|---|---|---|---|
| Model | Lags | Folds | MCMAE |
| Autoregressive Neural Network | 5 | 10 | 1.6043152928352356 |
| Autoregressive Neural Network | 20 | 10 | 1.6995096802711487 |
| Autoregressive Neural Network with Player box features | 5 | 10 | 3.3845490169525147 |

Table 5.4

To obtain the results, we used k-fold cross-validation to train the models and found the final MCMAE score through predicting and calculating the loss on a held out test set. The Various

ARNN's obtained better results then the VAR model with the ARNN's with just lags performing slightly worse than XGBoost and LightGBM. The set of ARNN's performing better than the VAR model can be attributed to the nature of the learning technique itself. Specifically, the universal approximation theorem (Horniket al., 1989; Cybenko, 1989) states that a feedforward network with a linear output layer and at least one hidden layer with any "squashing" activation function can approximate any Borel measurable function, provided that the network is given enough hidden units. Thus our network may be closer to mapping an unseen non-linear relationship between the targets and Lagged target values as the VAR model is only linear in nature. Compared to the GDBT algorithms, the results here are worse due to the less extensive feature set used and a less robust hyper parameter tuning methodology. The large discrepancy between the results of the ARNN with player box scores as extra features and those without can be directly attributed to the difference in data set size used to train each respective model. From the set of ARNNs, it is clear that the recent lags of a target value is significant in forecasting digital engagement.

## 6 Conclusions, Learnings and Further Works

In this project, we did extensive data exploration and analysis to determine a feature set which is meaningful to the prediction of next day digital engagement. Given the richness of our dataset, gradient boosting (LightGBM, XGBoost) and deep learning models (ARNN) performed better than the stochastic process model (VAR), with LightGBM performing on top. From our findings, factors which are most important to forecasting baseball fan engagement includes the lagged values of the targets, position name, the number of times a player appears in the dataset and the number of player and team Twitter followers.

A key takeaway is that the choice of features is not as important as the methods used to process and implement the features in the model. Although this may introduce additional bias depending on our encoding method, experimentation in the gradient boosting models seems to support this hypothesis.

For future work, we could further explore the effects of encoding, particularly for our other models. An area which we could analyse is the effectiveness of categorising events as in or off season and how that influences our results. Further research on hyper parameter tuning methods other than Optuna may also improve the accuracy score of our predictions. Additionally, collecting more data, in particular, relating to popular trends or viral videos from Youtube or Tiktok, the player's influence, by using natural language processing to determine whether they are advocates for social issues, as well as data about the fans themselves may generate richer insights into our problem. We could also experiment with a Long Short Term Memory Recurrent Neural Network (RNN). The main advantage of RNN over artificial neutral networks is that RNN can model sequence of data, like time series, so that each sample can be assumed to be dependent on previous ones.

# 7 BIBLIOGRAPHY

Hushani, P., 2018, Using Autoregressive Modelling and Machine Learning for Stock Market Prediction and Trading

Monigatti, L., 2021, Intro to Time Series Forecasting. Kaggle. February. Available from: https://www.kaggle.com/iamleonie/intro-to-time-series-forecasting [Accessed 20/07/2021]

Bosede Ngozi, A., 2018, Time Series Analysis(Lecture 2): Choosing Optimal Lags in EViews. Blogspot. 12 February. Available from: http://cruncheconometrix.blogspot.com/2018/02/time-series-analysis-lecture-2-choosing.html [Accessed 20/07/2021]

Hyndman, R.J., Athanasopoulos, G., 2018, Forecasting: principles and practice, 2nd edition, OTexts: Melbourne, Australia. Available from: https://otexts.com/fpp2/VAR.html [Accessed 21/07/2021]

Engineering Statistics Handbook, 6.4.4.2 Stationarity. Available from: https://www.itl.nist.gov/div898/handbook/pmc/section4/pmc442.htm. [Accessed 21/07/2021]

Statsmodels v0.13.0.dev0 (+542), 2021, statsmodels.tsa.stattools.adfuller. Available from: https://www.statsmodels.org/dev/generated/statsmodels.tsa.stattools.adfuller.html. [Accessed 19/07/2021]

Hauser, M., 2018, Vector autoregressions, VAR Chapter 2. Available from: http://statmath.wu.ac.at/ hauser/LVs/FinEtricsQF/FEtrics_Chp2.pdf. [Accessed 22/07/2021]

Wikipedia, 2021, Mean absolute error. Available from: https://en.wikipedia.org/wiki/Mean_absolute_error. [Accessed 10/07/2021]

Shi, H., 2007. Best-first decision tree learning (Doctoral dissertation, The University of Waikato). [Accessed 24/07/2021]

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.Y., 2017. Lightgbm: A highly efficient gradient boosting decision tree. Advances in neural information processing systems, 30, pp.3146-3154. [Accessed 24/07/2021]

Al Daoud, E., 2019. Comparison between XGBoost, LightGBM and CatBoost using a home credit dataset. International Journal of Computer and Information Engineering, 13(1), pp.6-10. [Accessed 24/07/2021]

Mwiti, D. (2021). Gradient Boosted Decision Trees [Guide] - a Conceptual Explanation. [online] neptune.ai. Available at: https://neptune.ai/blog/gradient-boosted-decision-trees-guide [Accessed 25 Jul. 2021].

Kai-wen, Z. and Chao, Y., 2017. Research of short-term load forecasting based on Gradient Boosting Decision Tree (GBDT). Guizhou Electric Power Technology, 2, pp.82-84. ARNN -

references

Cerqueira, V., Torgo, L. and Mozetič, I., 2020. Evaluating time series forecasting models: An empirical study on performance estimation methods. Machine Learning, 109(11), pp.1997-2028. [Accessed 20/07/2021]

Brownlee, J. (2018). How to Grid Search Deep Learning Models for Time Series Forecasting. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/how-to-grid-search-deep-learning-models-for-time-series-forecasting/ [Accessed 22/07/2021]

Otexts.com. 2021. 11.3 Neural network models | Forecasting: Principles and Practice (2nd ed). [online] Available at:
<https://otexts.com/fpp2/nnetar.html> [Accessed 21/07/2021]

Triebe, O., Laptev, N. and Rajagopal, R. (2019). AR-Net: A simple Auto-Regressive Neural Network for time-series. arXiv:1911.12436 [cs, stat]. [online] Available at:
https://arxiv.org/abs/1911.12436.[Accessed 15/07/2021]

citeseerx.ist.psu.edu. (n.d.). Download Limit Exceeded. [online] Available at:
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.48.2583rep=rep1type=pdf [Accessed 23/07/2021]

Nielsen, M.A. (2019). Neural Networks and Deep Learning. [online] Neuralnetworksanddeeplearning.com. Available at: http://neuralnetworksanddeeplearning.com/chap4.html. [Accessed 24/07/2021]

TensorFlow. (n.d.). Train and evaluate with Keras | TensorFlow Core. [online] Available at:
https://www.tensorflow.org/guide/keras/train_and_evaluate. [Accessed 17/07/2021]

# 8 Appendix

Figure 1.a

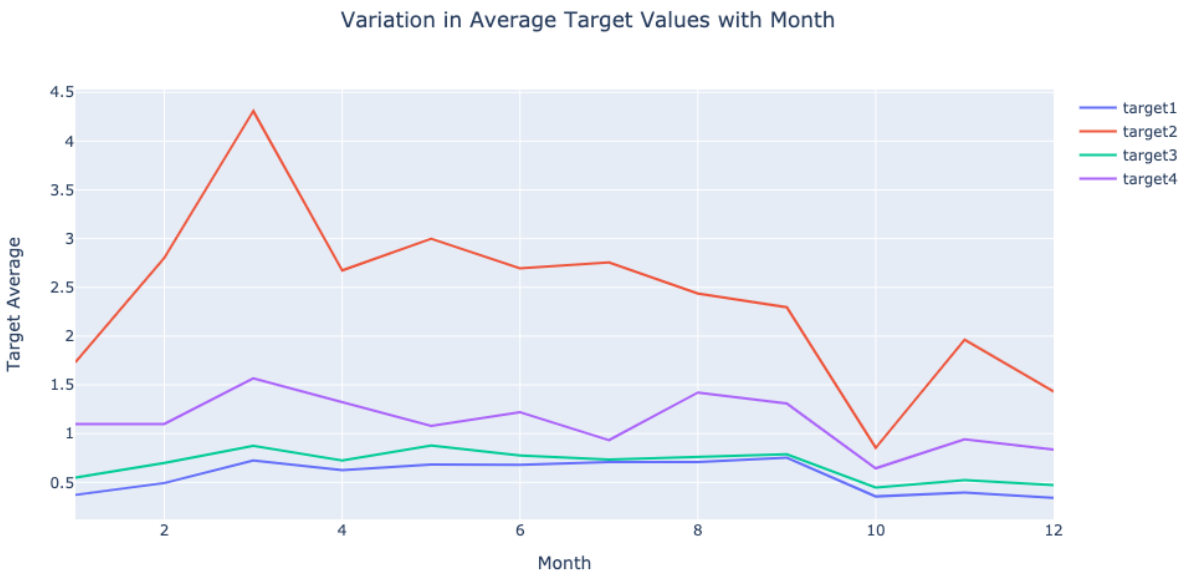


Figure 1.b

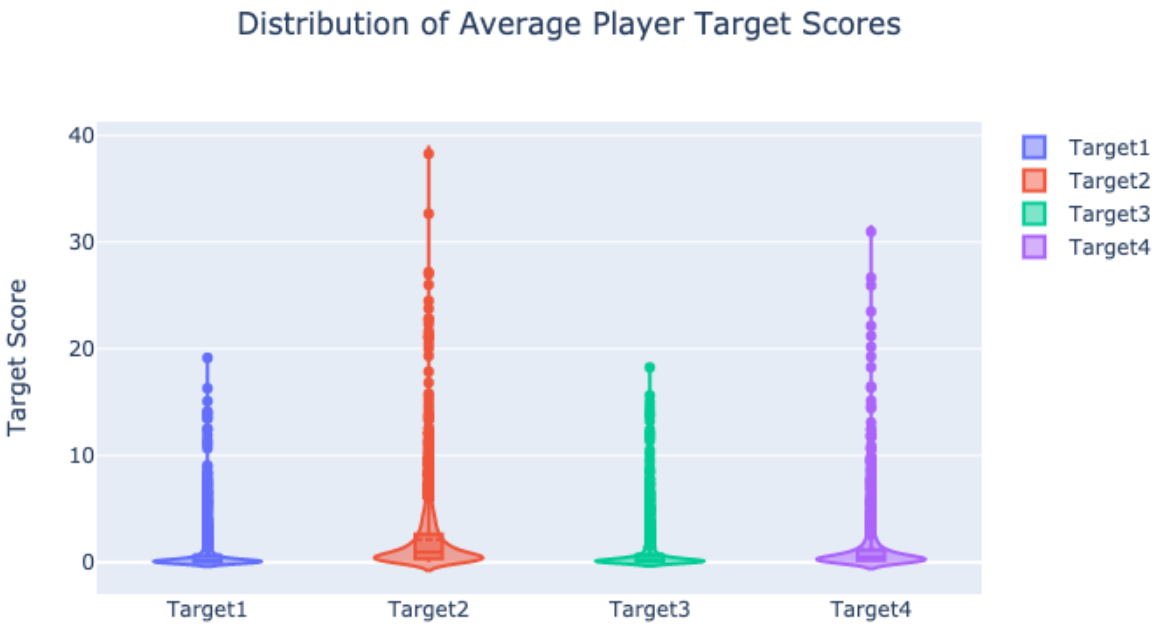



Figure 2.a

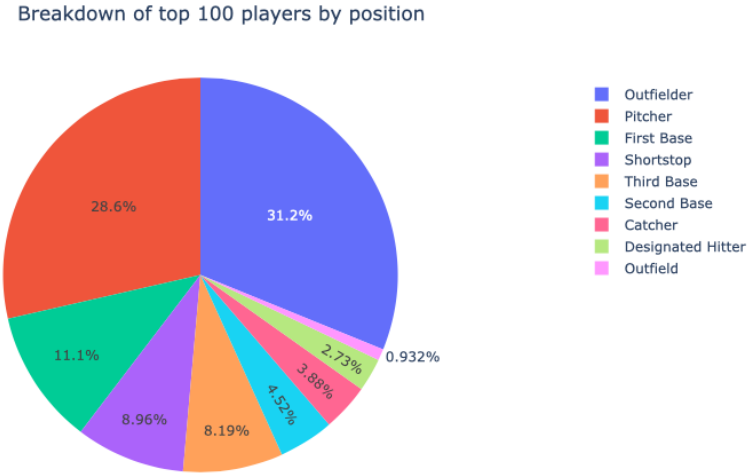


Figure 2.b

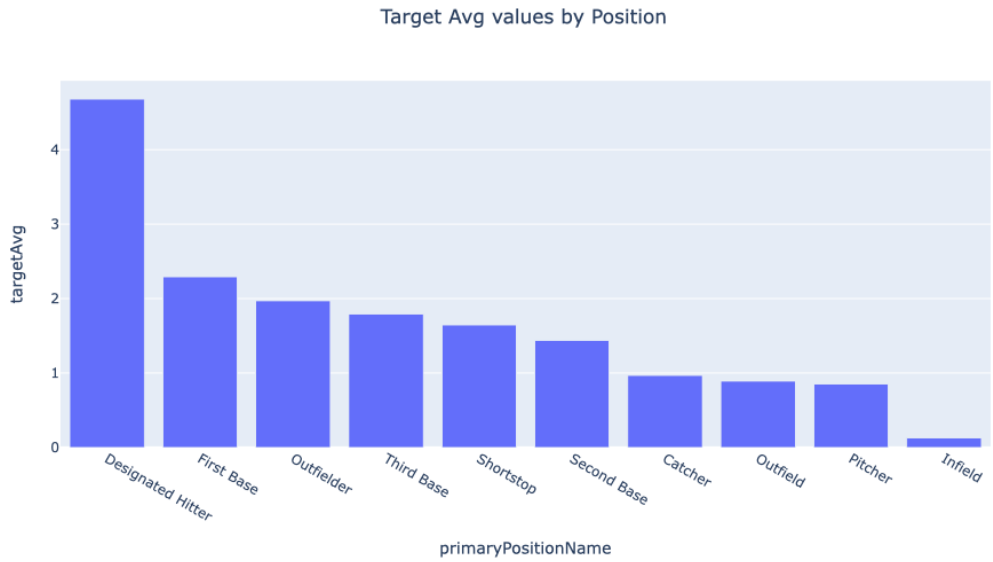



Figure 2.c

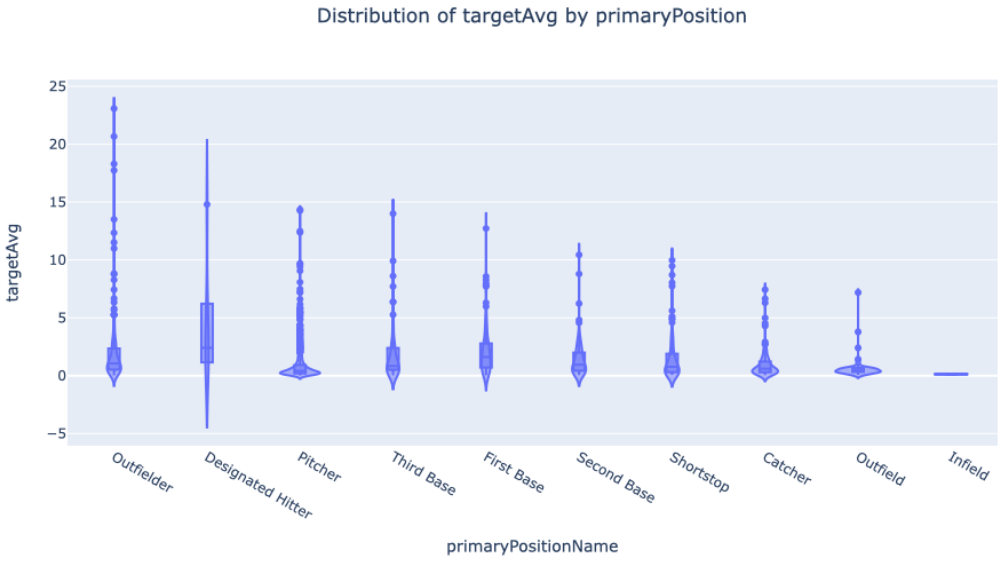


Figure 2.d

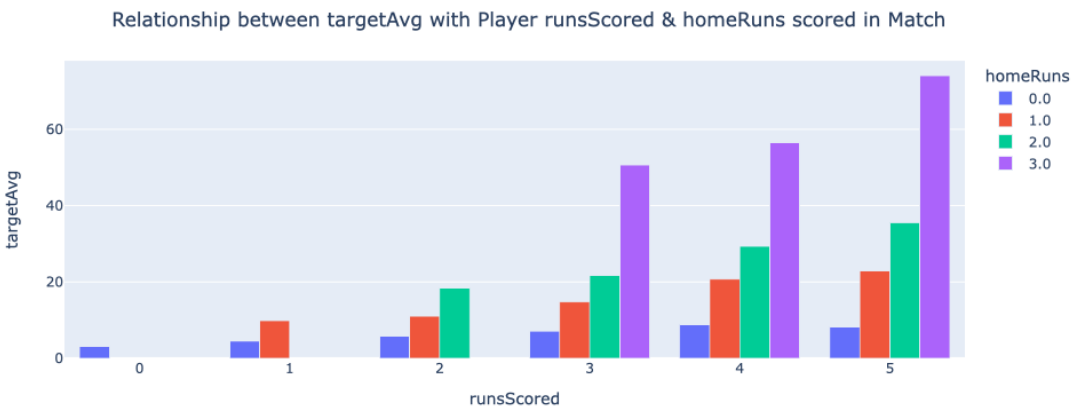
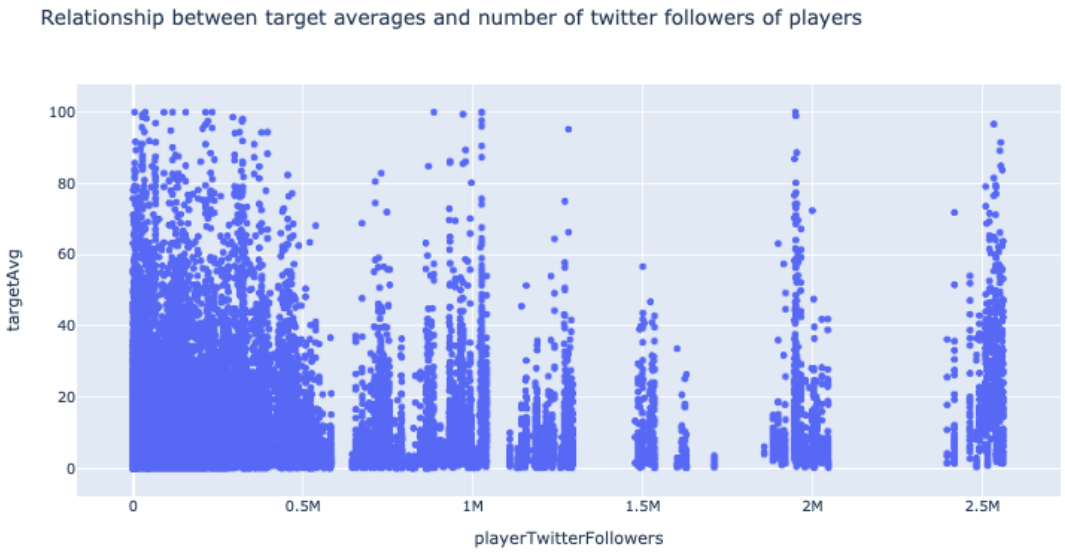

Relationship between targetAvg with Player runsScored & homeRuns scored in Match
targetAvg
0
20
40
60
0
1
2
3
4
5
runsScored
homeRuns
0.0
1.0
2.0
3.0

Table 3.a

| | target1 | target2 | target3 | target4 | targetAvg |
|---|---|---|---|---|---|
| pitchesThrown | -0.02 | -0.05 | 0.08 | -0.03 | -0.00 |
| inningsPitchedAsFrac | -0.01 | -0.04 | 0.09 | -0.03 | 0.01 |
| strikeOutsPitching | 0.01 | -0.02 | 0.09 | -0.01 | 0.03 |
| pitchingGameScore | -0.09 | -0.15 | 0.00 | -0.11 | -0.11 |
| pitches100mph | 0.01 | 0.02 | 0.02 | 0.03 | 0.02 |
| plateAppearances | 0.19 | 0.28 | 0.09 | 0.18 | 0.23 |
| hits | 0.24 | 0.26 | 0.13 | 0.15 | 0.25 |
| totalBases | 0.34 | 0.31 | 0.19 | 0.19 | 0.32 |
| winPctTeam | 0.08 | 0.09 | 0.08 | 0.09 | 0.11 |
| leagueRankTeam | -0.10 | -0.11 | -0.09 | -0.11 | -0.13 |
| winStreakTeam | 0.03 | 0.04 | 0.04 | 0.04 | 0.05 |
| lossStreakTeam | -0.03 | -0.03 | -0.02 | -0.03 | -0.03 |
| toDateAllStars | 0.15 | 0.24 | 0.16 | 0.36 | 0.29 |
| playerTwitterFollowers | 0.17 | 0.28 | 0.18 | 0.37 | 0.32 |
| teamTwitterFollowers | 0.10 | 0.11 | 0.10 | 0.13 | 0.14 |

Figure 3.b




Section 3.c

From Table 3.a, players with more Twitter followers and in teams with greater Twitter followings tend to have higher levels of digital engagement. However, a caveat is that low correlation does not inherently mean low predictive power as the relationships may not be linear. By analysing Figure 3.b, there seems to be no clear trend between the number of player Twitter followers and target average.

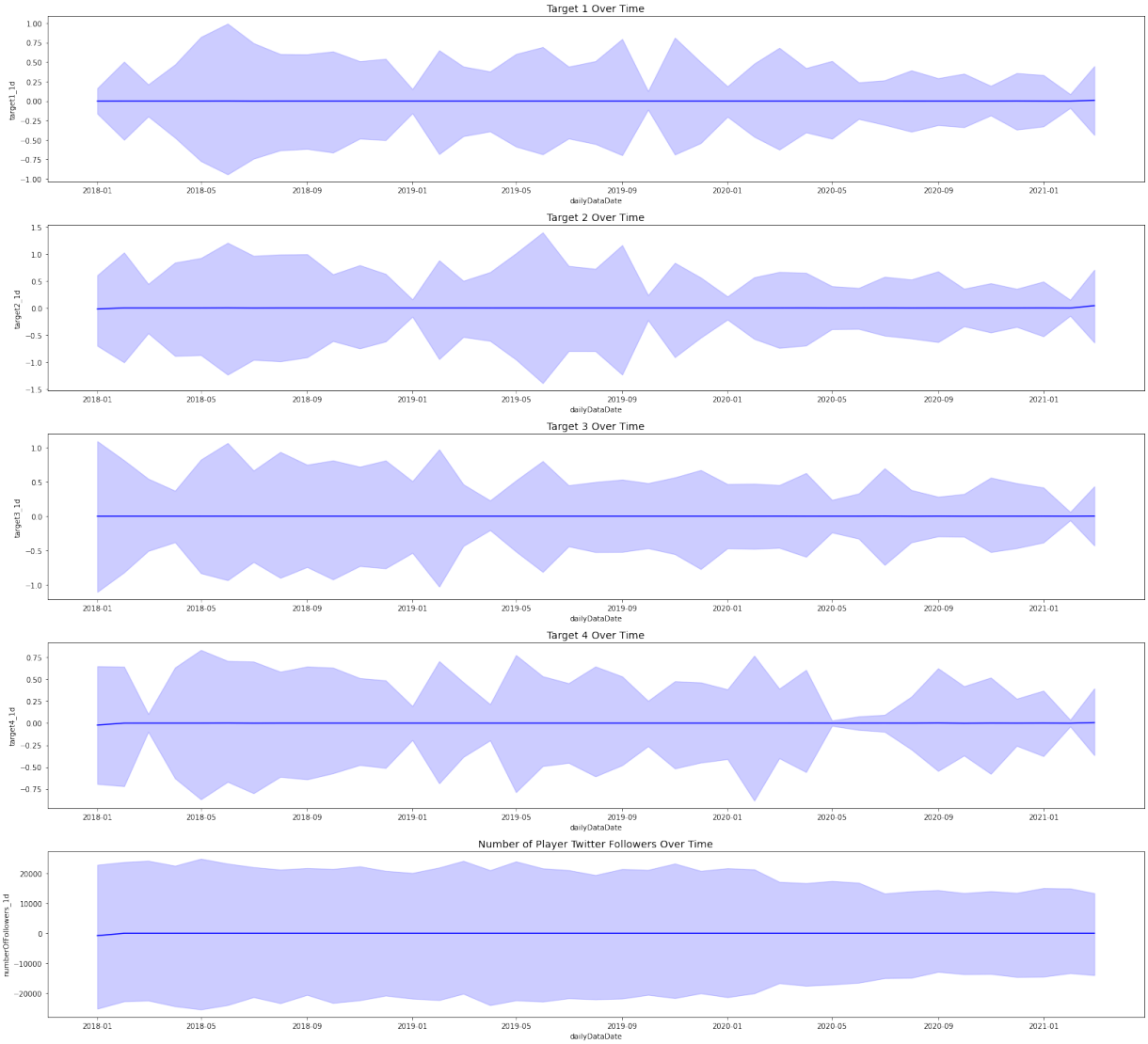
Figure 4.a


Target 1 Over Time
target1
dailyDataDate
Target 2 Over Time
target2
dailyDataDate
Target 3 Over Time
target3
dailyDataDate
Target 4 Over Time
target4
dailyDataDate
Number of Player Twitter Followers Over Time
numberOfFollowers
dailyDataDate

Section 4.b

```
   Augmented Dickey-Fuller Test on "target1"
   -----------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -37.9607
No. Lags Chosen       = 15
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "target2"
   -----------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -13.7466
No. Lags Chosen       = 49
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "target3"
   -----------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -19.7088
No. Lags Chosen       = 49
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "target4"
   -----------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -14.5564
No. Lags Chosen       = 50
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "numberOfFollowers"
   -----------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -18.4344
No. Lags Chosen       = 47
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.
```

Figure 4.c


Target 1 Over Time
target1_1d
dailyDataDate
Target 2 Over Time
target2_1d
dailyDataDate
Target 3 Over Time
target3_1d
dailyDataDate
Target 4 Over Time
target4_1d
dailyDataDate
Number of Player Twitter Followers Over Time
numberOfFollowers_1d
dailyDataDate

Section 4.d

```
   Augmented Dickey-Fuller Test on "target1_1d"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -40.8254
No. Lags Chosen       = 50
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "target2_1d"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -39.1977
No. Lags Chosen       = 50
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "target3_1d"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -41.1493
No. Lags Chosen       = 50
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "target4_1d"
   --------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -40.5353
No. Lags Chosen       = 50
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.


   Augmented Dickey-Fuller Test on "numberOfFollowers_1d"
   ------------------------------------------------------
Null Hypothesis: Data has unit root. Non-Stationary.
Significance Level    = 0.05
Test Statistic        = -38.0262
No. Lags Chosen       = 50
Critical value 1%     = -3.431
Critical value 5%     = -2.862
Critical value 10%    = -2.567
=> P-Value = 0.0. Rejecting Null Hypothesis.
=> Series is Stationary.
```

Figure 4.e

Stationarity involves a time series without trend and seasonality and with constant variance (Engineering Statistics Handbook). In our model, we checked for stationarity via two different approaches: visually and statistical testing. In Figure 4.a, while none of the features have clear trend lines or vast differences in variance, seasonality is evident in the target values. This non-stationarity was not recognised by the Augmented Dickey-Fuller (ADF) Test as shown in Section 4.b, with the p-values for each variables falling below the significance level of 0.05, thus rejecting the null hypothesis of containing a unit root (statsmodels 2021). To achieve stationarity, we shall take the first difference of our data. The output is displayed in Figure 4.c, where we can see that there is no clear trend, differences in variance and seasonality. The ADF test as shown in Section 4.d also reinforces the data's stationarity.

Table 5.a

VAR Order Selection (* highlights the minimums)

| | AIC | BIC | FPE | HQIC |
|---|---|---|---|---|
| **0** | 40.14 | 40.14 | 2.707e+17 | 40.14 |
| **1** | 38.74 | 38.75 | 6.676e+16 | 38.74 |
| **2** | 38.19 | 38.21 | 3.863e+16 | 38.20 |
| **3** | 37.82 | 37.84 | 2.668e+16 | 37.83 |
| **4** | 37.61 | 37.63 | 2.147e+16 | 37.61 |
| **5** | 37.47 | 37.51 | 1.881e+16 | 37.48 |
| **6** | 37.36 | 37.40 | 1.672e+16 | 37.37 |
| **7** | 37.29 | 37.34 | 1.567e+16 | 37.31 |
| **8** | 37.24 | 37.30 | 1.490e+16 | 37.26 |
| **9** | 37.19 | 37.26 | 1.423e+16 | 37.21 |
| **10** | 37.14 | 37.21 | 1.350e+16 | 37.16 |
| **11** | 37.11 | 37.19 | 1.306e+16 | 37.13 |
| **12** | 37.08 | 37.16 | 1.268e+16 | 37.11 |
| **13** | 37.06 | 37.15 | 1.244e+16 | 37.09 |
| **14** | 37.04 | 37.14 | 1.220e+16 | 37.07 |
| **15** | 37.02 | 37.13 | 1.199e+16 | 37.06 |
| **16** | 37.01 | 37.12 | 1.180e+16 | 37.04 |
| **17** | 37.00 | 37.11 | 1.166e+16 | 37.03 |
| **18** | 36.98 | 37.11 | 1.150e+16 | 37.02 |
| **19** | 36.97 | 37.10* | 1.138e+16 | 37.01 |
| **20** | 36.96 | 37.11 | 1.131e+16 | 37.01 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| **70** | 36.85 | 37.34 | 1.010e+16 | 37.01 |
| **71** | 36.85 | 37.35 | 1.011e+16 | 37.01 |
| **72** | 36.85 | 37.36 | 1.011e+16 | 37.01 |
| **73** | 36.85 | 37.36 | 1.010e+16 | 37.02 |
| **74** | 36.85 | 37.37 | 1.008e+16 | 37.02 |
| **75** | 36.85 | 37.37 | 1.007e+16 | 37.02 |
| **76** | 36.85 | 37.38 | 1.007e+16 | 37.02 |
| **77** | 36.85* | 37.39 | 1.007e+16* | 37.02 |
| **78** | 36.85 | 37.39 | 1.008e+16 | 37.02 |

Section 6.a

Given the equation:

$$\hat{\mathbf{y}}(\mathbf{t}) = \beta_{\mathbf{0}} + \sum \beta_{\mathbf{i}} \mathbf{y}(\mathbf{t} - \mathbf{i}) + \mathbf{e}(\mathbf{t})$$

where **(t)** is a vector of length 5, containing the predictions for *target1, target2, target3, target4* and *numberOfFollowers*. $VAR(p)$ in standard form has $pk^2$ many parameters excluding the constant terms and error variance-covariances (Hauser 2018), where $k$ is the length of the prediction vector. As our model was fitted on 77 lags, $p = 77$. Hence this gives us,

$$77 \times 5^2 = 1925$$

parameters.

Table 7.a

| **Method Validation** | **Max Lags** | **Best Lag** | **MCMAE Score** |
|---|---|---|---|
| 3 Fold Cross Validation | 20/ 50/ 100 | 4 | 6.818151463056363 |
| 5 Fold Cross Validation | 20/ 50/ 100 | 1 | 11.617829946194332 |
| 10 Fold Cross Validation | 20/ 50/ 100 | 2 | 9.330740431657674 |
| 20 Fold Cross Validation | 20/ 50/ 100 | 2 | 9.330740431657674 |
| Hold Out | 20 | 20 (based on min AIC and BIC) | 3.599367399554331 |
| Hold Out | 50 | 50 (based on min AIC and BIC) | 3.0195746708659232 |
| Hold Out | 100 | 77 (based on min AIC) | 2.938680265163667 |
| Hold Out | 100 | 19 (based on min BIC) | 3.596527655933557 |