Yen Forecast

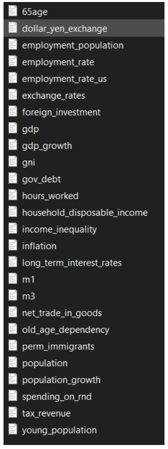
Introduction

Japan for decades has stood on the global stage with the third largest economy, albeit far behind the USA and China. However this year Germany has overtaken Japan in GDP, and the current value of the Japanese Yen (JPY) is plummeting. This is likely due to many factors, including aging population, rural depopulation, and numerous other socioeconomic factors.

Using fiscal data such as the historical value of the Yen and other metrics such as M2, real M2, M1, exchange rates, can we predict the real value of the yen over time, to create an arbitrage opportunity?

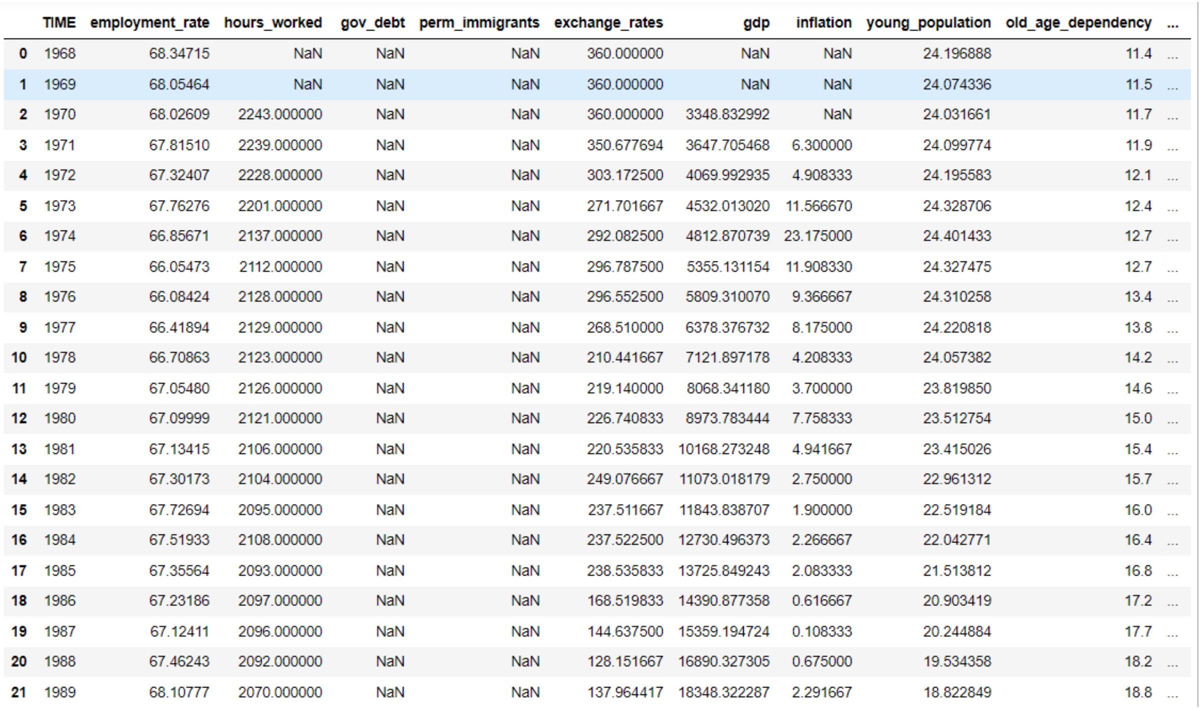
Methods part 1

I began my collecting as much data as I could that I thought could be relevant. This included 26 datasets consisting of multiple different factors such as percentage of population of 65, gdp growth, gni, m1, and many more.



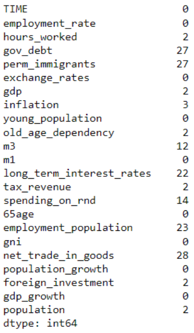
```
65age
dollar_yen_exchange
employment_population
employment_rate
employment_rate_us
exchange_rates
foreign_investment
gdp
gdp_growth
gni
gov_debt
hours_worked
household_disposable_income
income_inequality
inflation
long_term_interest_rates
m1
m3
net_trade_in_goods
old_age_dependency
perm_immigrants
population
population_growth
spending_on_rnd
tax_revenue
young_population
```

Unfortunately not all of the data came from the same place, but I was able to join all of the datasets by year.



	TIME	employment_rate	hours_worked	gov_debt	perm_immigrants	exchange_rates	gdp	inflation	young_population	old_age_dependency	...
0	1968	68.34715	NaN	NaN	NaN	360.000000	NaN	NaN	24.196888	11.4	...
1	1969	68.05464	NaN	NaN	NaN	360.000000	NaN	NaN	24.074336	11.5	...
2	1970	68.02609	2243.000000	NaN	NaN	360.000000	3348.832992	NaN	24.031661	11.7	...
3	1971	67.81510	2239.000000	NaN	NaN	350.677694	3647.705468	6.300000	24.099774	11.9	...
4	1972	67.32407	2228.000000	NaN	NaN	303.172500	4069.992935	4.908333	24.195583	12.1	...
5	1973	67.76276	2201.000000	NaN	NaN	271.701667	4532.013020	11.566670	24.328706	12.4	...
6	1974	66.85671	2137.000000	NaN	NaN	292.082500	4812.870739	23.175000	24.401433	12.7	...
7	1975	66.05473	2112.000000	NaN	NaN	296.787500	5355.131154	11.908330	24.327475	12.7	...
8	1976	66.08424	2128.000000	NaN	NaN	296.552500	5809.310070	9.366667	24.310258	13.4	...
9	1977	66.41894	2129.000000	NaN	NaN	268.510000	6378.376732	8.175000	24.220818	13.8	...
10	1978	66.70863	2123.000000	NaN	NaN	210.441667	7121.897178	4.208333	24.057382	14.2	...
11	1979	67.05480	2126.000000	NaN	NaN	219.140000	8068.341180	3.700000	23.819850	14.6	...
12	1980	67.09999	2121.000000	NaN	NaN	226.740833	8973.783444	7.758333	23.512754	15.0	...
13	1981	67.13415	2106.000000	NaN	NaN	220.535833	10168.273248	4.941667	23.415026	15.4	...
14	1982	67.30173	2104.000000	NaN	NaN	249.076667	11073.018179	2.750000	22.961312	15.7	...
15	1983	67.72694	2095.000000	NaN	NaN	237.511667	11843.838707	1.900000	22.519184	16.0	...
16	1984	67.51933	2108.000000	NaN	NaN	237.522500	12730.496373	2.266667	22.042771	16.4	...
17	1985	67.35564	2093.000000	NaN	NaN	238.535833	13725.849243	2.083333	21.513812	16.8	...
18	1986	67.23186	2097.000000	NaN	NaN	168.519833	14390.877358	0.616667	20.903419	17.2	...
19	1987	67.12411	2096.000000	NaN	NaN	144.637500	15359.194724	0.108333	20.244884	17.7	...
20	1988	67.46243	2092.000000	NaN	NaN	128.151667	16890.327305	0.675000	19.534358	18.2	...
21	1989	68.10777	2070.000000	NaN	NaN	137.964417	18348.322287	2.291667	18.822849	18.8	...

The next problem that arose was the sheer number of null values. Some variables had 28 rows of null values, which presents a problem when we only have 55 years (rows) of data.

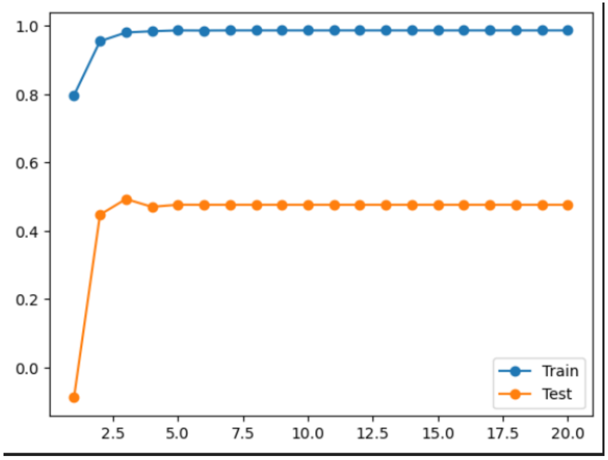


```
TIME                       0
employment_rate            0
hours_worked               2
gov_debt                  27
perm_immigrants           27
exchange_rates             0
gdp                        2
inflation                  3
young_population           0
old_age_dependency         2
m3                        12
m1                         0
long_term_interest_rates  22
tax_revenue                2
spending_on_rnd           14
65age                      0
employment_population     23
gni                        0
net_trade_in_goods        28
population_growth          0
foreign_investment         2
gdp_growth                 0
population                 2
dtype: int64
```

I knew this would be an issue but I wanted to see which variables were important to keep so for now I just removed all of the null values and proceeded.

The initial plan was to use a random forest model, I was particularly interested in the decision tree based models and their hyperparameter tuning. Random Forests are also known for their high accuracy. I was aware of the difficulties of creating a predictor model especially on a target variable like exchange rates, so I wanted to be able to test and adjust the model as much as I could. That being said we will quickly see a pivot from the random forest.

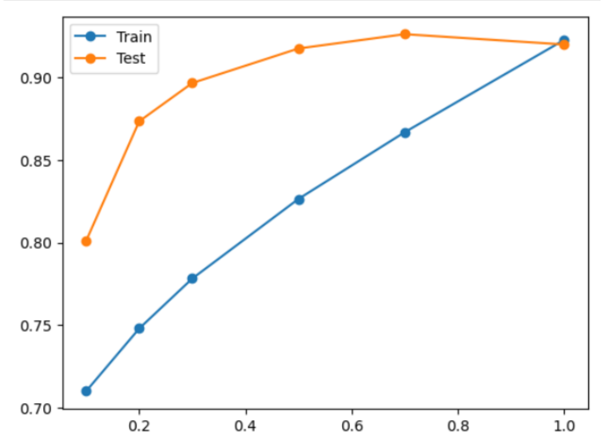
As I was testing my hyperparameters I quickly found a large disparity between the training and testing scores.



	MSE	Train R2	Test R2
Scores	51.36	0.98	0.46

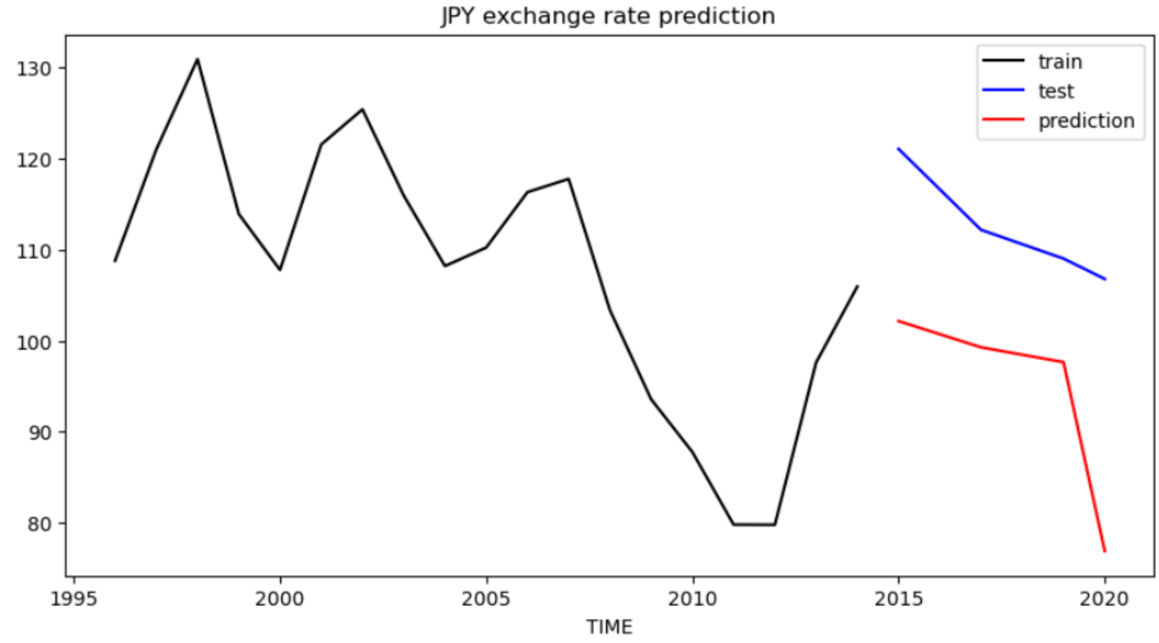
In addition, we can see the train r2 value is far too high. This is a sign of major overfitting, so I decided to look into different models. The model I decided on was an extreme gradient boosting tree.

I chose this model for a few different reasons. Our main goal is to reduce overfitting, and one of the best ways to do that is by reducing model complexity. The individual trees in an xgboost model are not built to their full depth which helps reduce overfitting. Additionally the gradient boosting models improve their accuracy with each tree trained so we will likely see improvements in accuracy.



	MSE	Train R2	Test R2
Scores	6.79	0.90	0.93

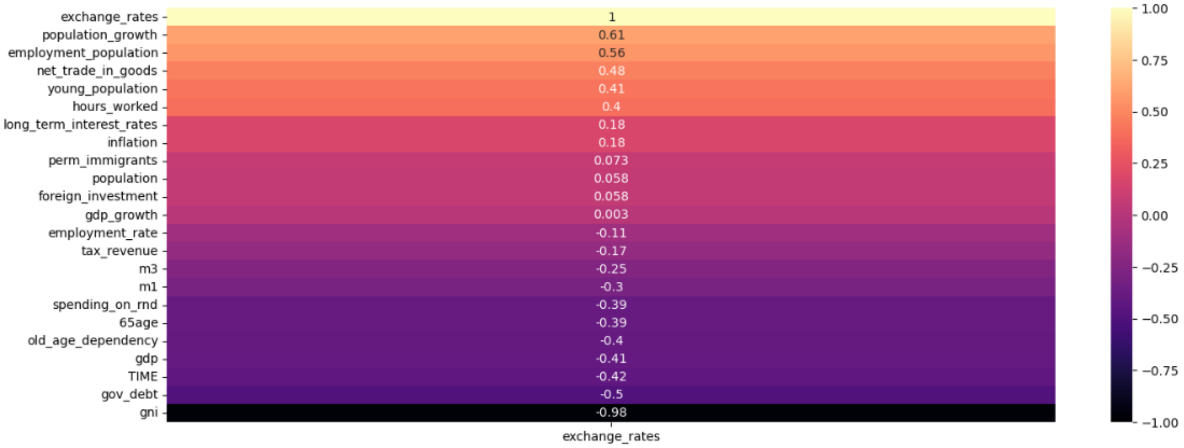
And we do, we have brought both the training and test r2 values much closer together, and the MSE is near zero. However, these results are still signs of overfitting. While we were able to improve the results slightly by switching models, the real problem lies within the datasets themselves!



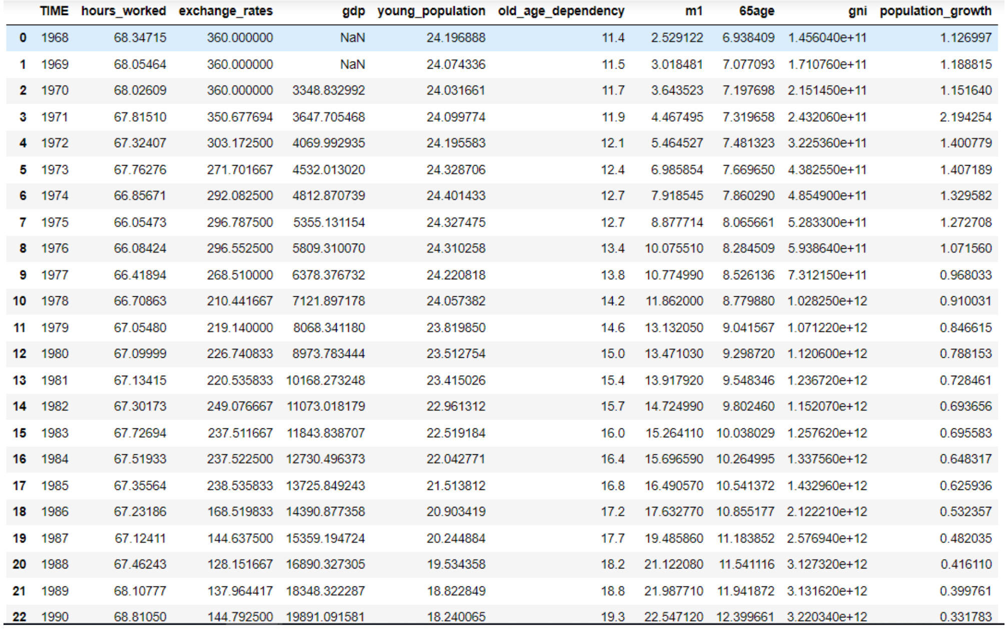
As you can see our prediction from the actual value (test) is pretty far off!

Methods part 2

First we must look at how each predictor impacted our prediction. We can use some feature importance graphs to visualize this:

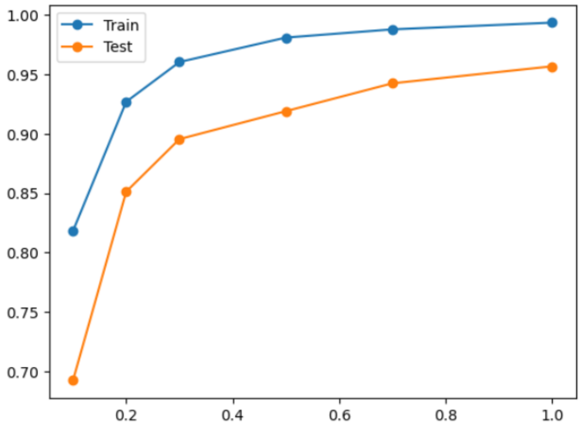


As we can see there are lots of variables that have very little impact on the prediction itself. We can remove both these variables that have little importance and the variables that have many null values. The first will reduce our model complexity even further, to reduce overfitting. The second will add more rows and give our model more data to train on!



	TIME	hours_worked	exchange_rates	gdp	young_population	old_age_dependency	m1	65age	gni	population_growth
0	1968	68.34715	360.000000	NaN	24.196888	11.4	2.529122	6.938409	1.456040e+11	1.126997
1	1969	68.05464	360.000000	NaN	24.074336	11.5	3.018481	7.077093	1.710760e+11	1.188815
2	1970	68.02609	360.000000	3348.832992	24.031661	11.7	3.643523	7.197698	2.151450e+11	1.151640
3	1971	67.81510	350.677694	3647.705468	24.099774	11.9	4.467495	7.319658	2.432060e+11	2.194254
4	1972	67.32407	303.172500	4069.992935	24.195583	12.1	5.464527	7.481323	3.225360e+11	1.400779
5	1973	67.76276	271.701667	4532.013020	24.328706	12.4	6.985854	7.669650	4.382550e+11	1.407189
6	1974	66.85671	292.082500	4812.870739	24.401433	12.7	7.918545	7.860290	4.854900e+11	1.329582
7	1975	66.05473	296.787500	5355.131154	24.327475	12.7	8.877714	8.065661	5.283300e+11	1.272708
8	1976	66.08424	296.552500	5809.310070	24.310258	13.4	10.075510	8.284509	5.938640e+11	1.071560
9	1977	66.41894	268.510000	6378.376732	24.220818	13.8	10.774990	8.526136	7.312150e+11	0.968033
10	1978	66.70863	210.441667	7121.897178	24.057382	14.2	11.862000	8.779880	1.028250e+12	0.910031
11	1979	67.05480	219.140000	8068.341180	23.819850	14.6	13.132050	9.041567	1.071220e+12	0.846615
12	1980	67.09999	226.740833	8973.783444	23.512754	15.0	13.471030	9.298720	1.120600e+12	0.788153
13	1981	67.13415	220.535833	10168.273248	23.415026	15.4	13.917920	9.548346	1.236720e+12	0.728461
14	1982	67.30173	249.076667	11073.018179	22.961312	15.7	14.724990	9.802460	1.152070e+12	0.693656
15	1983	67.72694	237.511667	11843.838707	22.519184	16.0	15.264110	10.038029	1.257620e+12	0.695583
16	1984	67.51933	237.522500	12730.496373	22.042771	16.4	15.696590	10.264995	1.337560e+12	0.648317
17	1985	67.35564	238.535833	13725.849243	21.513812	16.8	16.490570	10.541372	1.432960e+12	0.625936
18	1986	67.23186	168.519833	14390.877358	20.903419	17.2	17.632770	10.855177	2.122210e+12	0.532357
19	1987	67.12411	144.637500	15359.194724	20.244884	17.7	19.485860	11.183852	2.576940e+12	0.482035
20	1988	67.46243	128.151667	16890.327305	19.534358	18.2	21.122080	11.541116	3.127320e+12	0.416110
21	1989	68.10777	137.964417	18348.322287	18.822849	18.8	21.987710	11.941872	3.131620e+12	0.399761
22	1990	68.81050	144.792500	19891.091581	18.240065	19.3	22.547120	12.399661	3.220340e+12	0.331783

We ended up with only 8 predictors afterwards (from 25 before!) and 51 rows. We can then proceed with our hyperparameter tuning and test our model:

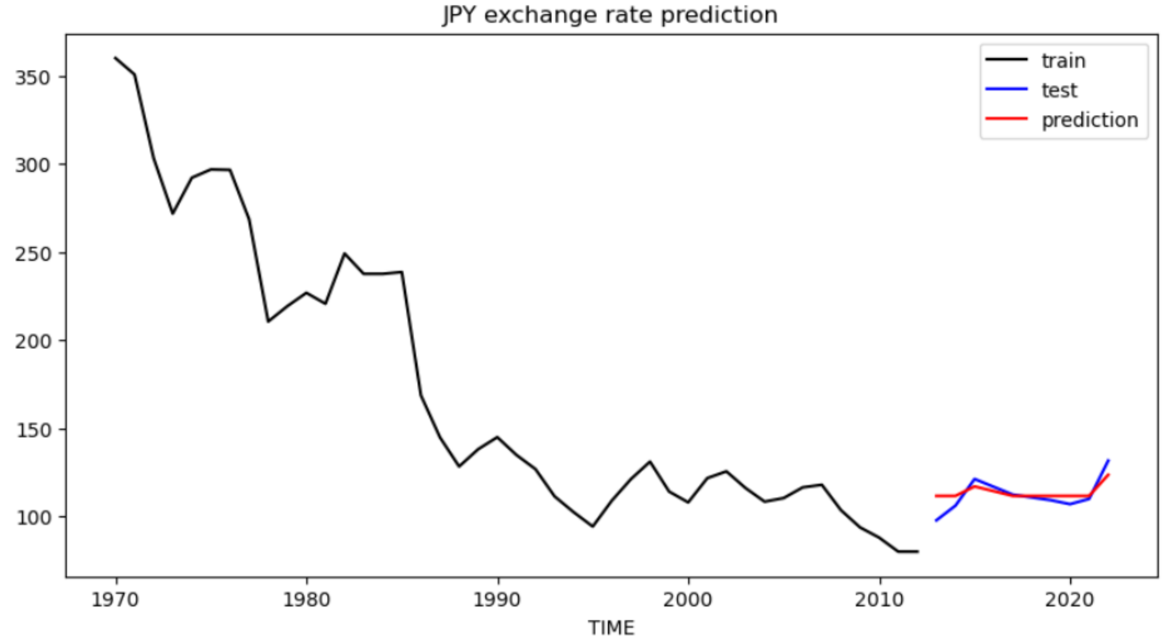


	MSE	Train R2	Test R2
Scores	1685.63	0.88	0.79

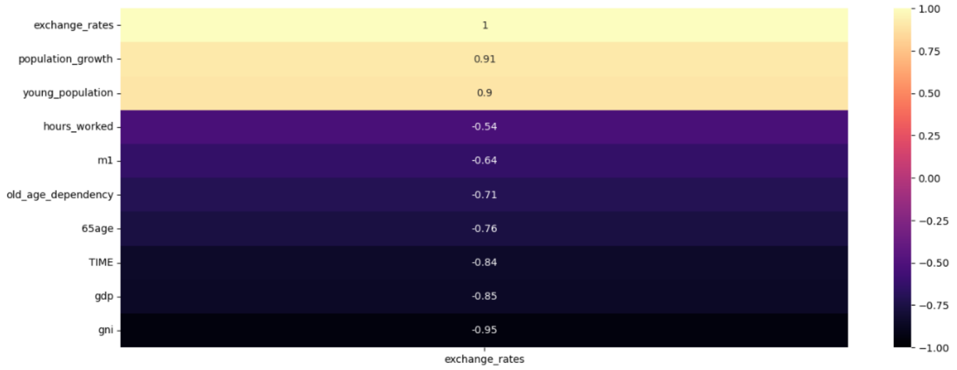
Our results become much more realistic!

Results

Our second xgboost model with our cleaned data performed much better than our first random forest model, and has far less overfitting.



We can see there are far less predictors, but the only ones that remain have far more coorelation!

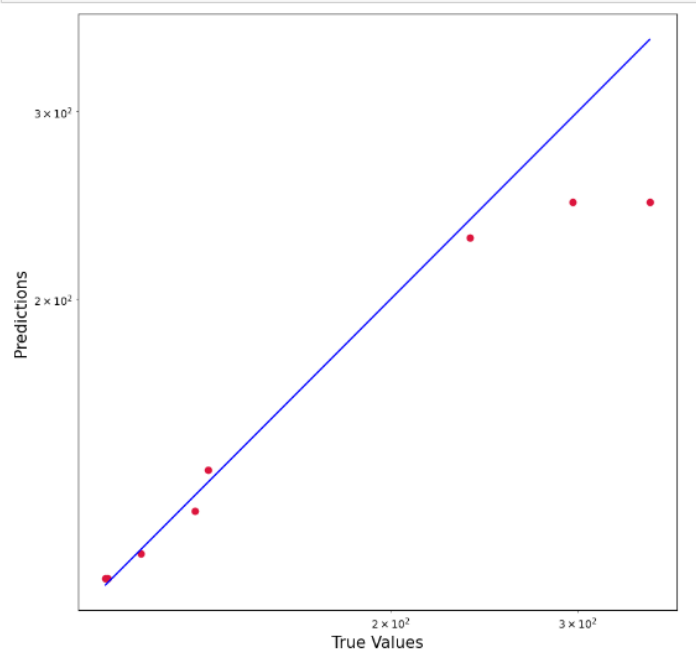


	MSE	Train R2	Test R2
Scores	1685.63	0.88	0.79

Discussion/Reflection

We were able to significantly improve the performance of our forecasting model through a few different methods. Even with hyperparameter tuning our initial random forest model had problems with overfitting, so in order to reduce complexity we switched to an extreme boosting gradient tree model. In addition to this and the hyperparamter tuning, we also cleaned much of our data. We initially had about 27 rows and 25 predictor variables, but after some feature importance analysis we decided to reduce the predictors to 8, and which in turn increased the rows to 51. These two combined provided a much better prediction. However there are still some issues:

Plotting the predictions from the actual values, the predictions do get notably worse the farther it gets from the training data



While I the fact that the prediction gets worse the farther ahead into the future it goes is completely reasonable, I think it is still telling of how difficult it is to predict a target variable like exchange rate. Even with as many as 25 predictors (which I had expected to have more importance) there simply was not a lot of coorelation between the variables and their patterns. In addition, we had as little as 55 years worth of data to use, which is likely not enough to have an accurate forecasting model.