Stock Market Analysis with RNN

Frode, Ria, Teis

Table of contents

Our goal

ML goal: Predict return of stocks with RNN.

Main goal: *Can we create a portfolio that generates a better return than LightGBM?*

Time series data

	permno	month	ret_excess	mktcap_lag	sic2	macro_dp	macro_dy	macro_ep	macro_svar	macro_bm	...	characteristic_retvol	characteristic_std_dolvol
0	10026	2000-01-01	-0.071173	184.848500	20.0	-4.423936	-4.476181	-3.346471	0.005206	0.154654	...	0.009387	0.177490
1	10026	2000-02-01	-0.040248	172.450125	20.0	-4.402228	-4.422541	-3.307460	0.003000	0.167056	...	-0.379047	0.142339
2	10026	2000-03-01	0.073266	166.250937	20.0	-4.493159	-4.400835	-3.381429	0.006678	0.149974	...	-0.424079	0.290580
3	10026	2000-04-01	-0.202713	178.477500	20.0	-4.463033	-4.494313	-3.343822	0.007942	0.152600	...	-0.152970	0.436264
4	10026	2000-05-01	-0.071667	140.409375	20.0	-4.442030	-4.464188	-3.315378	0.005185	0.155669	...	-0.292083	0.652242
...	...	...	...	...	...	...	...	...	...	...	...	...	...
335407	92807	2020-08-01	-0.027975	48.959328	60.0	-4.080892	-4.013173	-3.569975	0.000743	0.235975	...	-0.196950	0.693906
335408	92807	2020-09-01	0.007068	47.594609	60.0	-4.045576	-4.085594	-3.533379	0.004907	0.241482	...	-0.634702	0.643793
335409	92807	2020-10-01	-0.008997	47.935789	60.0	-4.020767	-4.048823	-3.519301	0.003661	0.253146	...	-0.669062	0.522848
335410	92807	2020-11-01	0.188306	47.082840	60.0	-4.126172	-4.024025	-3.635623	0.002492	0.226352	...	-0.009408	0.716852
335411	92807	2020-12-01	-0.045832	55.953520	60.0	-4.165889	-4.129440	-3.686452	0.000678	0.219195	...	0.168611	0.352018

Time series data – preprocessing

Only stocks between January 2000 and December 2020.

Standardise data with StandardScaler (for neural network only).

Added extra features for fun:

- Oil price
- LIBOR rates

Final data set:

- 1331 stocks
- 252 time stamps
- 175 features (92 micro, 18 macro, 65 sectors)

Building the model – feature selection

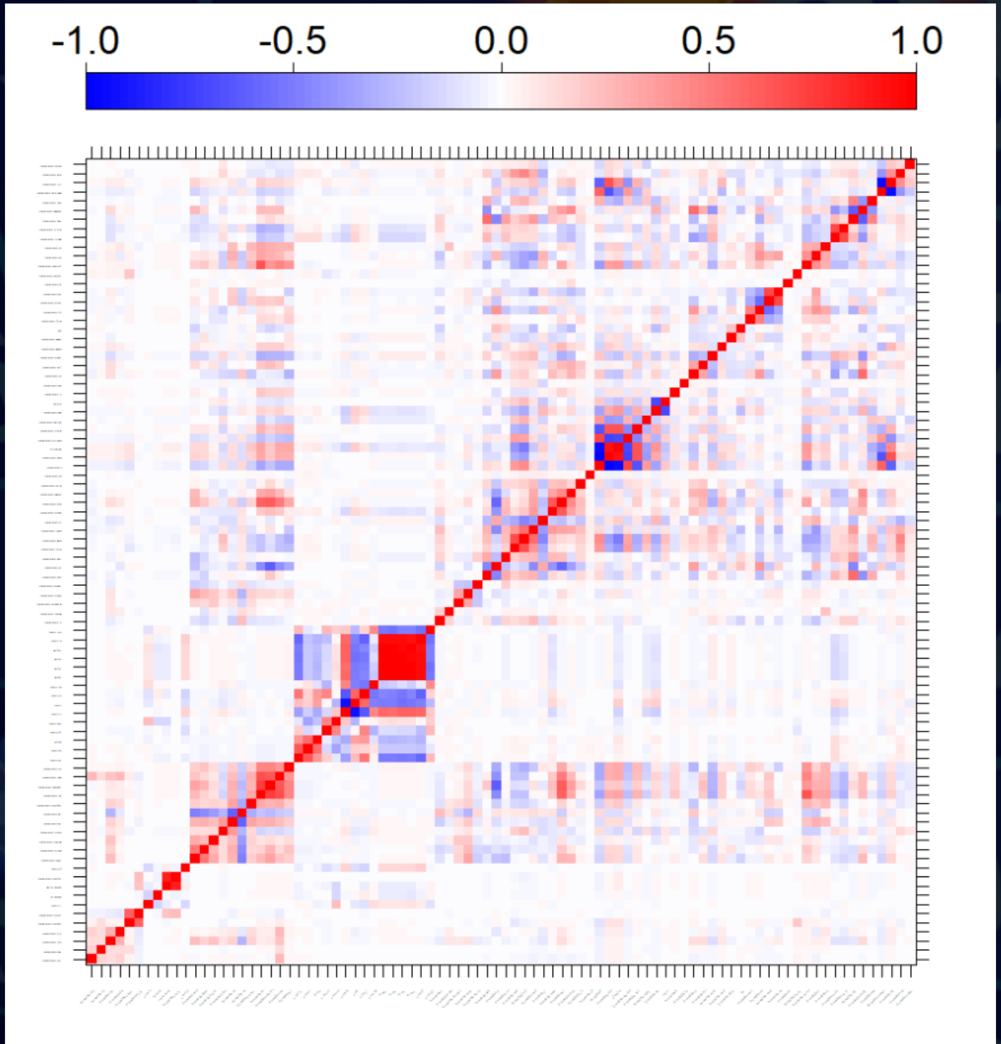
Time series data:

- Feature importance may be time dependent

Removing the features which are the least correlated to the target on average.

- End with 159 features

Alternative: do yearly feature selection.



LightGBM – model architecture

Base model which we want to beat.

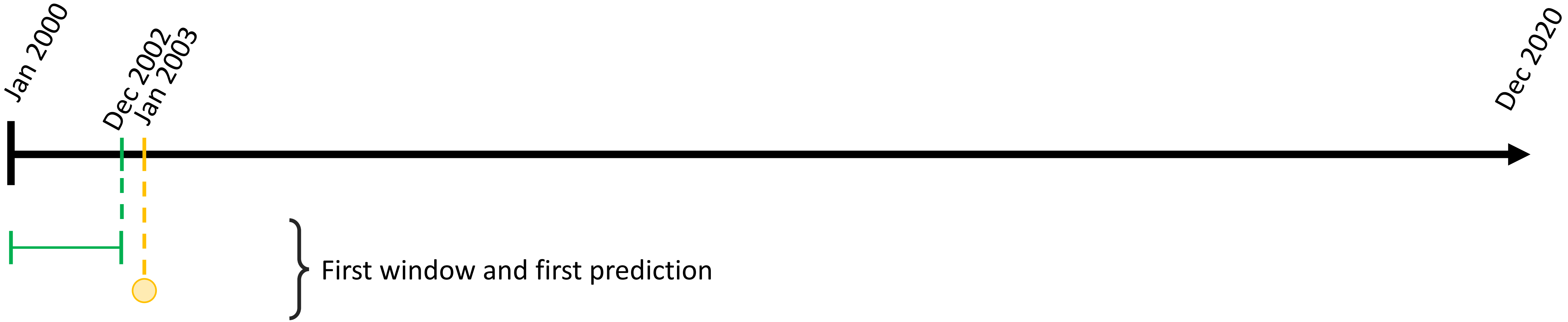
Hyperparameters:

- $num_{leaves} = 70$
- $learning\ rate = 0.005$
- $n_{estimators} = 120$
- $\min_{child\_samples} = 10$
- Early stopping: 30

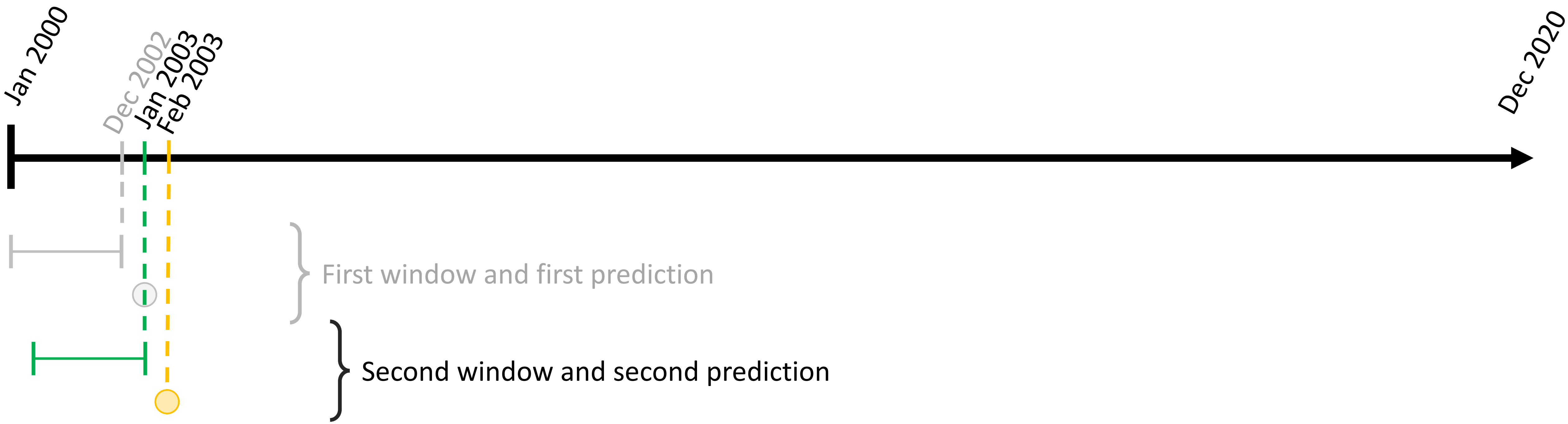
LightGBM Setup

LightGBM – rolling window on time series data

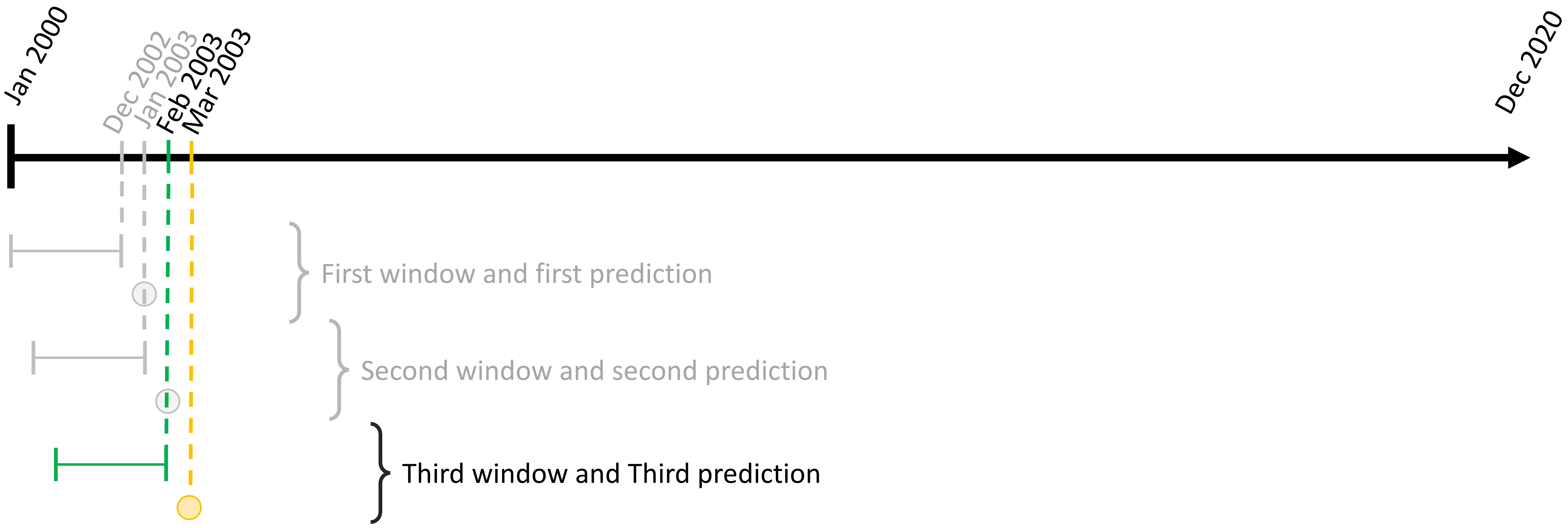
LightGBM – rolling window on time series data



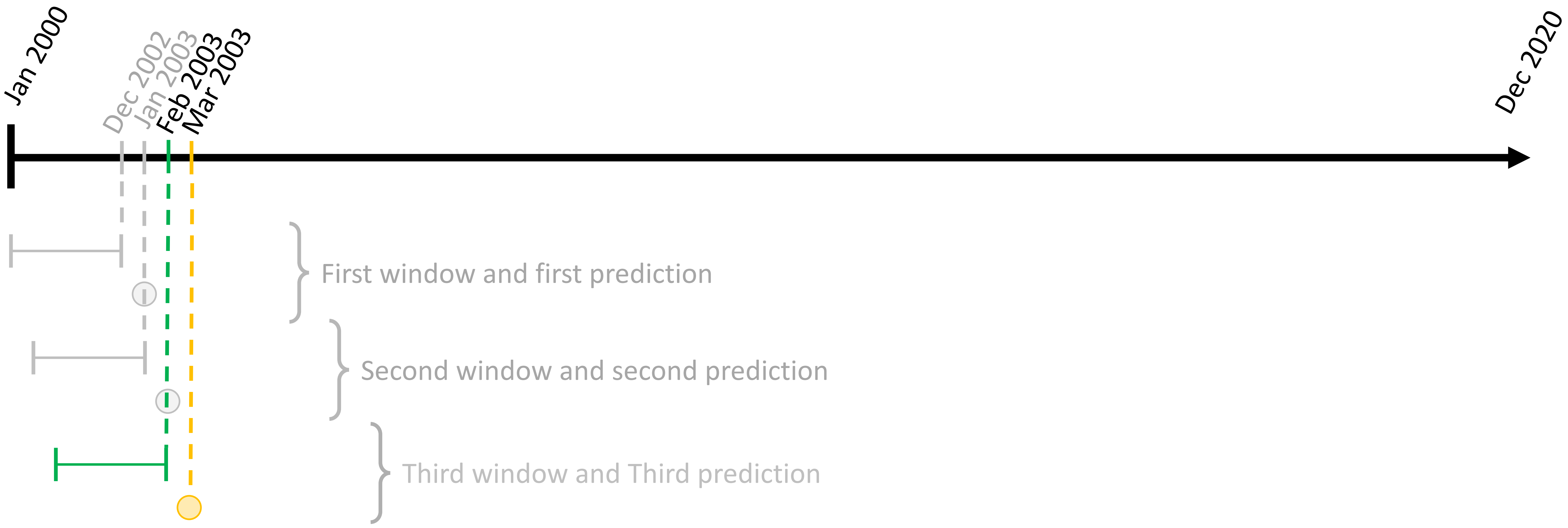
LightGBM – rolling window on time series data



LightGBM – rolling window on time series data

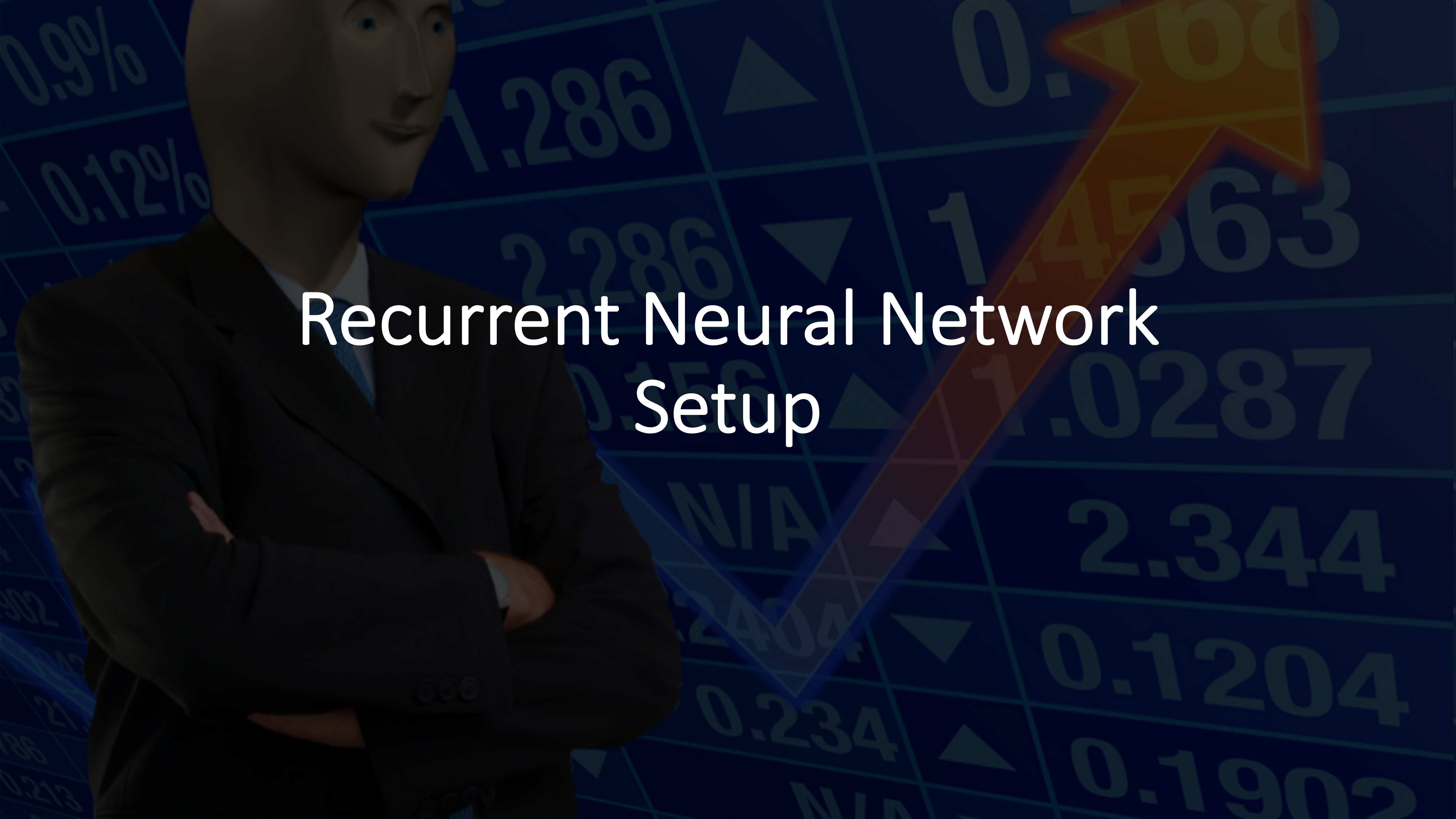


LightGBM – rolling window on time series data

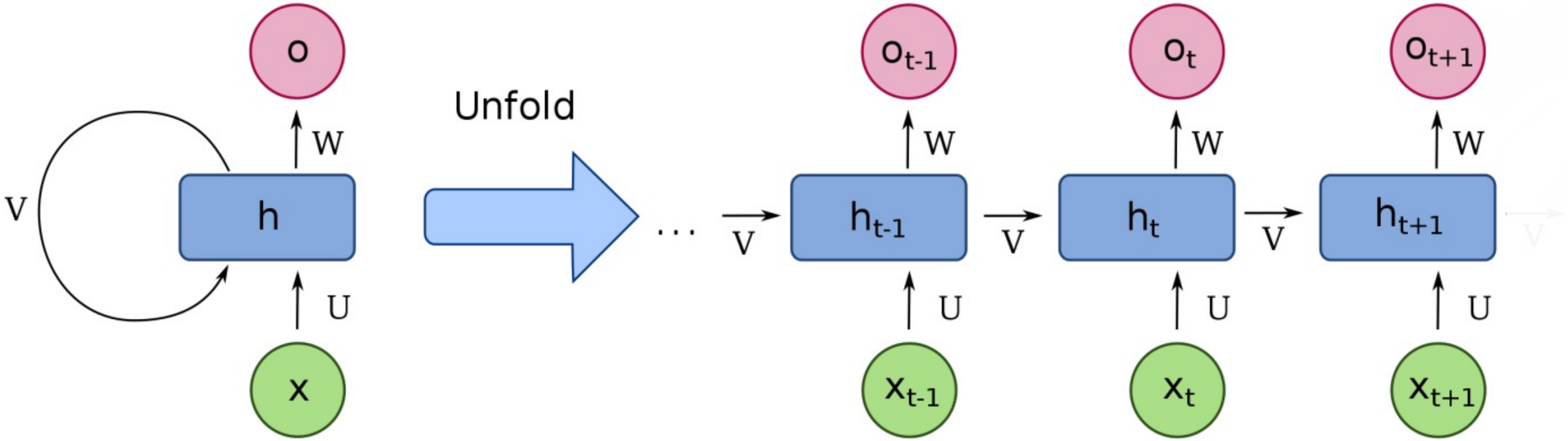


... and so on

Recurrent Neural Network Setup

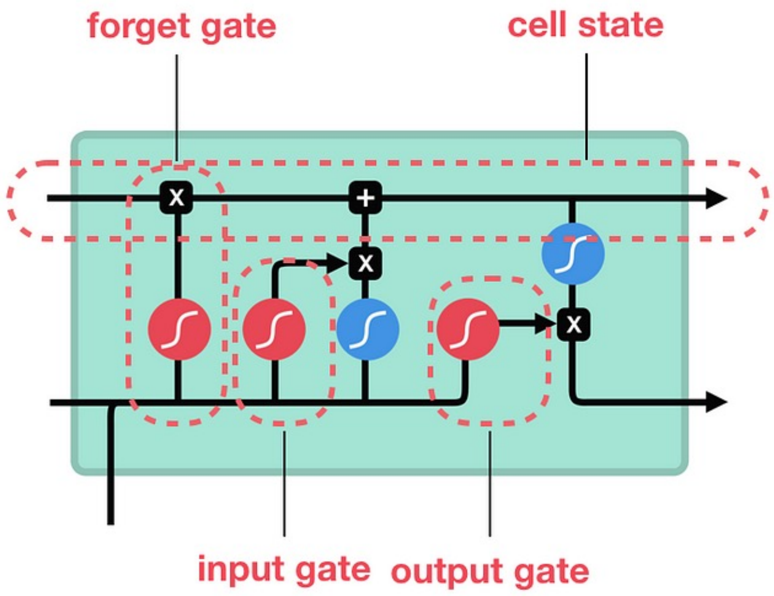


Recurrent Neural Network

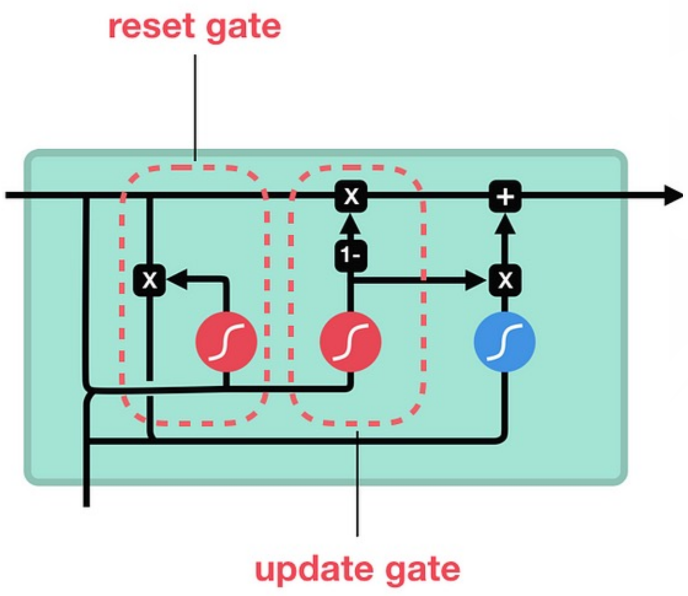


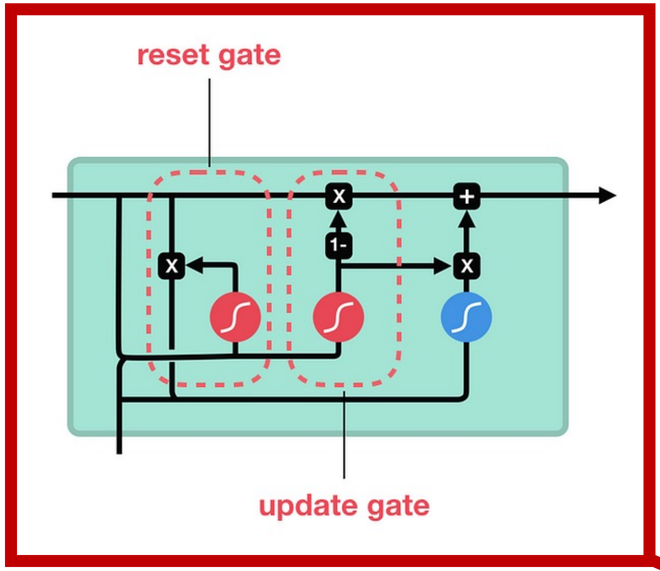
LSTM vs. GRU

LSTM

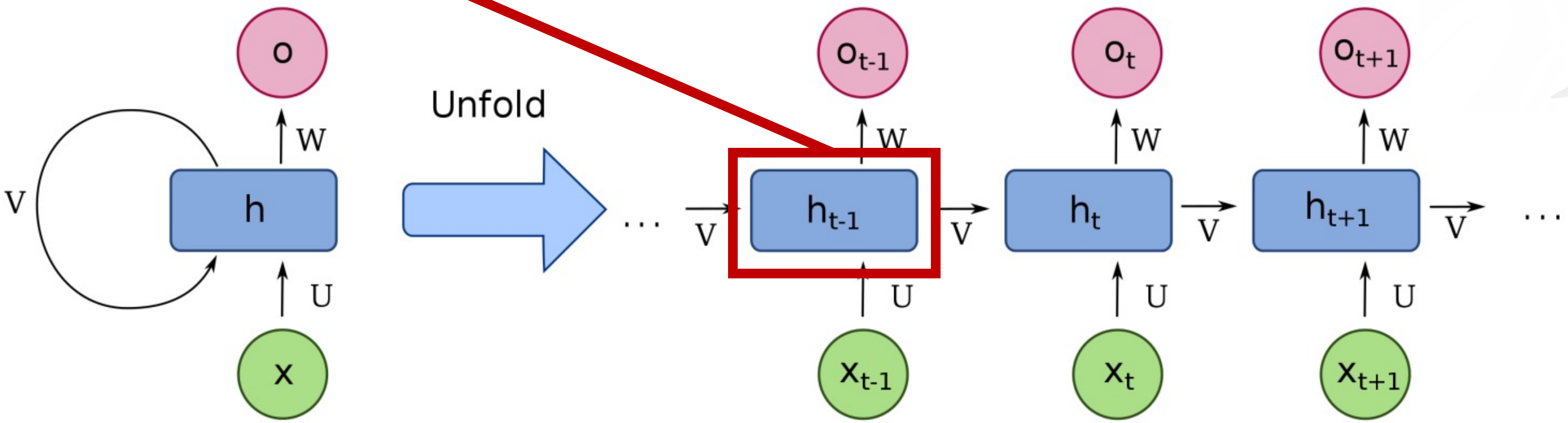


GRU

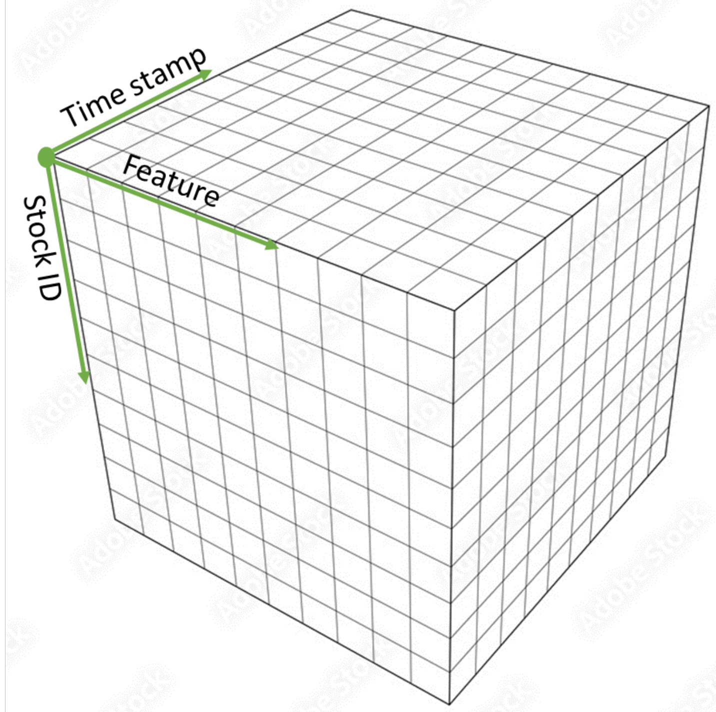


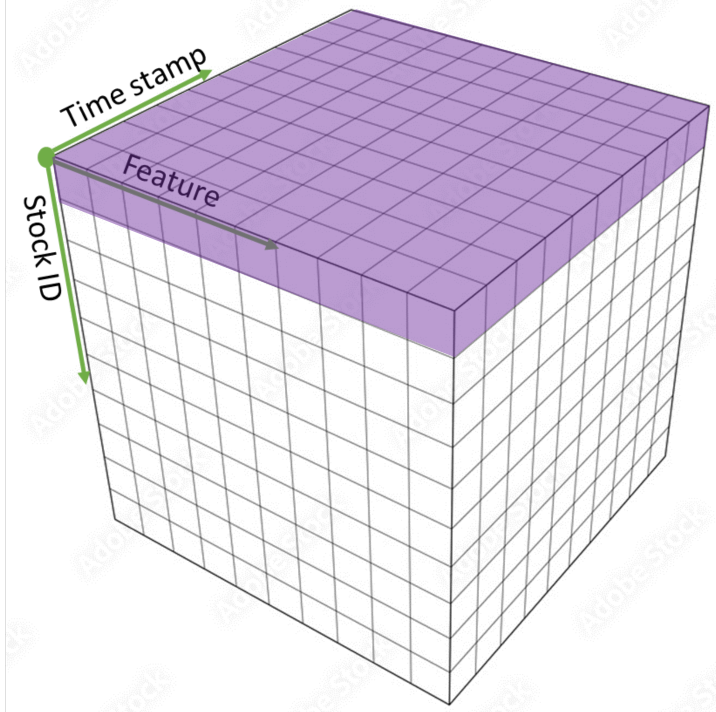


GRU structure



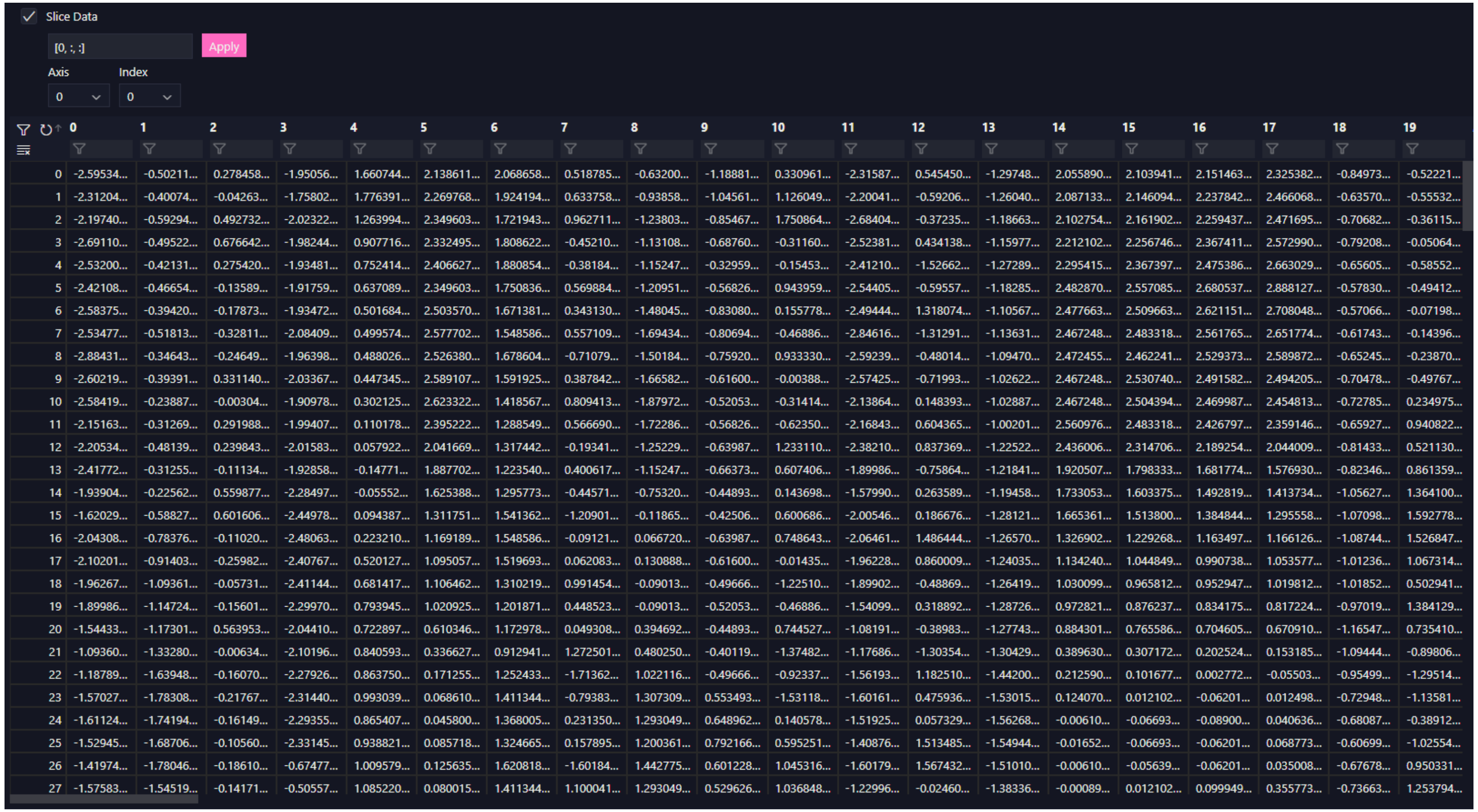
Data format for training the model


Time stamp
Feature
Stock ID



$$ncol = 159 \ \ \text{(features)}$$

$$nrow = 12\,m * 21\,yrs = 252 \ \ \text{(time stamps)}$$

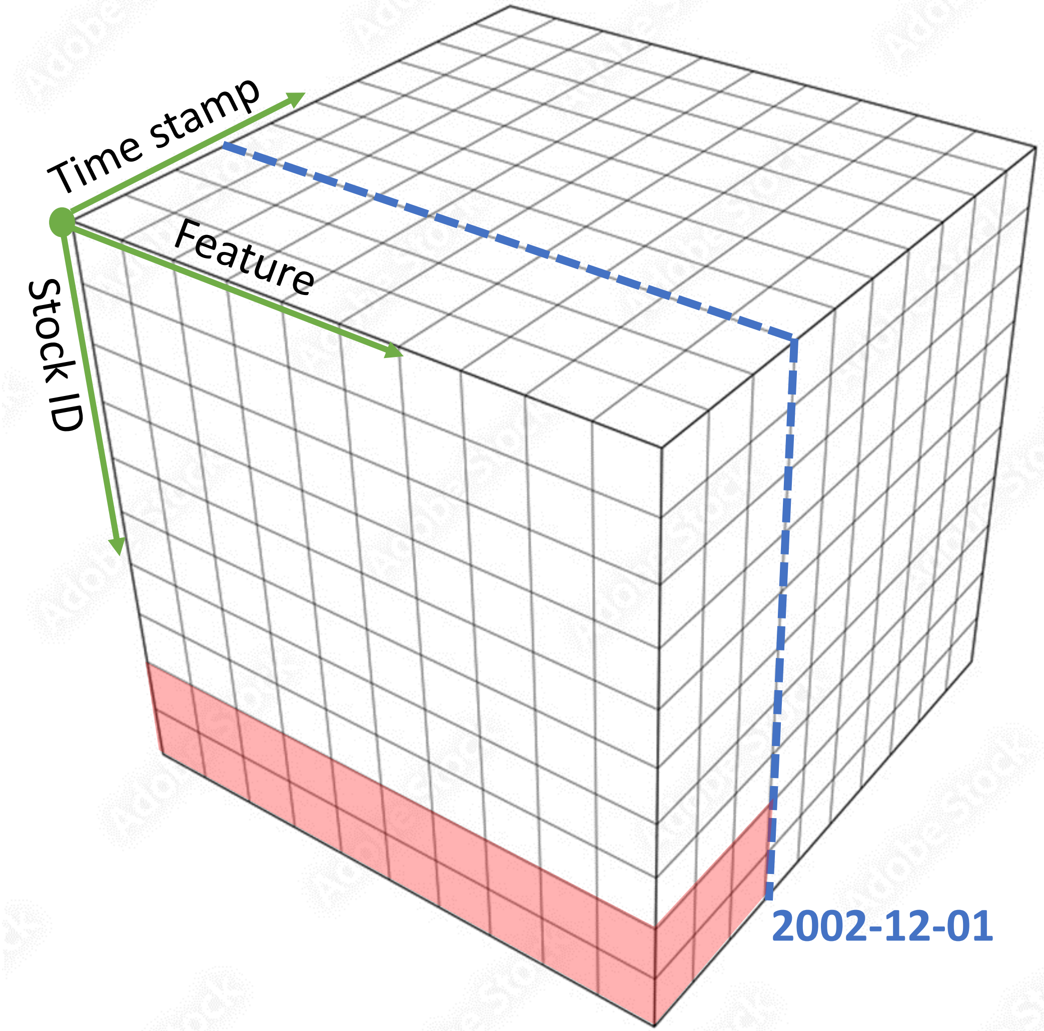


	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	-2.59534...	-0.50211...	0.278458...	-1.95056...	1.660744...	2.138611...	2.068658...	0.518785...	-0.63200...	-1.18881...	0.330961...	-2.31587...	0.545450...	-1.29748...	2.055890...	2.103941...	2.151463...	2.325382...	-0.84973...	-0.52221...
1	-2.31204...	-0.40074...	-0.04263...	-1.75802...	1.776391...	2.269768...	1.924194...	0.633758...	-0.93858...	-1.04561...	1.126049...	-2.20041...	-0.59206...	-1.26040...	2.087133...	2.146094...	2.237842...	2.466068...	-0.63570...	-0.55532...
2	-2.19740...	-0.59294...	0.492732...	-2.02322...	1.263994...	2.349603...	1.721943...	0.962711...	-1.23803...	-0.85467...	1.750864...	-2.68404...	-0.37235...	-1.18663...	2.102754...	2.161902...	2.259437...	2.471695...	-0.70682...	-0.36115...
3	-2.69110...	-0.49522...	0.676642...	-1.98244...	0.907716...	2.332495...	1.808622...	-0.45210...	-1.13108...	-0.68760...	-0.31160...	-2.52381...	0.434138...	-1.15977...	2.212102...	2.256746...	2.367411...	2.572990...	-0.79208...	-0.05064...
4	-2.53200...	-0.42131...	0.275420...	-1.93481...	0.752414...	2.406627...	1.880854...	-0.38184...	-1.15247...	-0.32959...	-0.15453...	-2.41210...	-1.52662...	-1.27289...	2.295415...	2.367397...	2.475386...	2.663029...	-0.65605...	-0.58552...
5	-2.42108...	-0.46654...	-0.13589...	-1.91759...	0.637089...	2.349603...	1.750836...	0.569884...	-1.20951...	-0.56826...	0.943959...	-2.54405...	-0.59557...	-1.18285...	2.482870...	2.557085...	2.680537...	2.888127...	-0.57830...	-0.49412...
6	-2.58375...	-0.39420...	-0.17873...	-1.93472...	0.501684...	2.503570...	1.671381...	0.343130...	-1.48045...	-0.83080...	0.155778...	-2.49444...	1.318074...	-1.10567...	2.477663...	2.509663...	2.621151...	2.708048...	-0.57066...	-0.07198...
7	-2.53477...	-0.51813...	-0.32811...	-2.08409...	0.499574...	2.577702...	1.548586...	0.557109...	-1.69434...	-0.80694...	-0.46886...	-2.84616...	-1.31291...	-1.13631...	2.467248...	2.483318...	2.561765...	2.651774...	-0.61743...	-0.14396...
8	-2.88431...	-0.34643...	-0.24649...	-1.96398...	0.488026...	2.526380...	1.678604...	-0.71079...	-1.50184...	-0.75920...	0.933330...	-2.59239...	-0.48014...	-1.09470...	2.472455...	2.462241...	2.529373...	2.589872...	-0.65245...	-0.23870...
9	-2.60219...	-0.39391...	0.331140...	-2.03367...	0.447345...	2.589107...	1.591925...	0.387842...	-1.66582...	-0.61600...	-0.00388...	-2.57425...	-0.71993...	-1.02622...	2.467248...	2.530740...	2.491582...	2.494205...	-0.70478...	-0.49767...
10	-2.58419...	-0.23887...	-0.00304...	-1.90978...	0.302125...	2.623322...	1.418567...	0.809413...	-1.87972...	-0.52053...	-0.31414...	-2.13864...	0.148393...	-1.02887...	2.467248...	2.504394...	2.469987...	2.454813...	-0.72785...	0.234975...
11	-2.15163...	-0.31269...	0.291988...	-1.99407...	0.110178...	2.395222...	1.288549...	0.566690...	-1.72286...	-0.56826...	-0.62350...	-2.16843...	0.604365...	-1.00201...	2.560976...	2.483318...	2.426797...	2.359146...	-0.65927...	0.940822...
12	-2.20534...	-0.48139...	0.239843...	-2.01583...	0.057922...	2.041669...	1.317442...	-0.19341...	-1.25229...	-0.63987...	1.233110...	-2.38210...	0.837369...	-1.22522...	2.436006...	2.314706...	2.189254...	2.044009...	-0.81433...	0.521130...
13	-2.41772...	-0.31255...	-0.11134...	-1.92858...	-0.14771...	1.887702...	1.223540...	0.400617...	-1.15247...	-0.66373...	0.607406...	-1.89986...	-0.75864...	-1.21841...	1.920507...	1.798333...	1.681774...	1.576930...	-0.82346...	0.861359...
14	-1.93904...	-0.22562...	0.559877...	-2.28497...	-0.05552...	1.625388...	1.295773...	-0.44571...	-0.75320...	-0.44893...	0.143698...	-1.57990...	0.263589...	-1.19458...	1.733053...	1.603375...	1.492819...	1.413734...	-1.05627...	1.364100...
15	-1.62029...	-0.58827...	0.601606...	-2.44978...	0.094387...	1.311751...	1.541362...	-1.20901...	-0.11865...	-0.42506...	0.600686...	-2.00546...	0.186676...	-1.28121...	1.665361...	1.513800...	1.384844...	1.295558...	-1.07098...	1.592778...
16	-2.04308...	-0.78376...	-0.11020...	-2.48063...	0.223210...	1.169189...	1.548586...	-0.09121...	0.066720...	-0.63987...	0.748643...	-2.06461...	1.486444...	-1.26570...	1.326902...	1.229268...	1.163497...	1.166126...	-1.08744...	1.526847...
17	-2.10201...	-0.91403...	-0.25982...	-2.40767...	0.520127...	1.095057...	1.519693...	0.062083...	0.130888...	-0.61600...	-0.01435...	-1.96228...	0.860009...	-1.24035...	1.134240...	1.044849...	0.990738...	1.053577...	-1.01236...	1.067314...
18	-1.96267...	-1.09361...	-0.05731...	-2.41144...	0.681417...	1.106462...	1.310219...	0.991454...	-0.09013...	-0.49666...	-1.22510...	-1.89902...	-0.48869...	-1.26419...	1.030099...	0.965812...	0.952947...	1.019812...	-1.01852...	0.502941...
19	-1.89986...	-1.14724...	-0.15601...	-2.29970...	0.793945...	1.020925...	1.201871...	0.448523...	-0.09013...	-0.52053...	-0.46886...	-1.54099...	0.318892...	-1.28726...	0.972821...	0.876237...	0.834175...	0.817224...	-0.97019...	1.384129...
20	-1.54433...	-1.17301...	0.563953...	-2.04410...	0.722897...	0.610346...	1.172978...	0.049308...	0.394692...	-0.44893...	0.744527...	-1.08191...	-0.38983...	-1.27743...	0.884301...	0.765586...	0.704605...	0.670910...	-1.16547...	0.735410...
21	-1.09360...	-1.33280...	-0.00634...	-2.10196...	0.840593...	0.336627...	0.912941...	1.272501...	0.480250...	-0.40119...	-1.37482...	-1.17686...	-1.30354...	-1.30429...	0.389630...	0.307172...	0.202524...	0.153185...	-1.09444...	-0.89806...
22	-1.18789...	-1.63948...	-0.16070...	-2.27926...	0.863750...	0.171255...	1.252433...	-1.71362...	1.022116...	-0.49666...	-0.92337...	-1.56193...	1.182510...	-1.44200...	0.212590...	0.101677...	0.002772...	-0.05503...	-0.95499...	-1.29514...
23	-1.57027...	-1.78308...	-0.21767...	-2.31440...	0.993039...	0.068610...	1.411344...	-0.79383...	1.307309...	0.553493...	-1.53118...	-1.60161...	0.475936...	-1.53015...	0.124070...	0.012102...	-0.06201...	0.012498...	-0.72948...	-1.13581...
24	-1.61124...	-1.74194...	-0.16149...	-2.29355...	0.865407...	0.045800...	1.368005...	0.231350...	1.293049...	0.648962...	0.140578...	-1.51925...	0.057329...	-1.56268...	-0.00610...	-0.06693...	-0.08900...	0.040636...	-0.68087...	-0.38912...
25	-1.52945...	-1.68706...	-0.10560...	-2.33145...	0.938821...	0.085718...	1.324665...	0.157895...	1.200361...	0.792166...	0.595251...	-1.40876...	1.513485...	-1.54944...	-0.01652...	-0.06693...	-0.06201...	0.068773...	-0.60699...	-1.02554...
26	-1.41974...	-1.78046...	-0.18610...	-0.67477...	1.009579...	0.125635...	1.620818...	-1.60184...	1.442775...	0.601228...	1.045316...	-1.60179...	1.567432...	-1.51010...	-0.00610...	-0.05639...	-0.06201...	0.035008...	-0.67678...	0.950331...
27	-1.57583...	-1.54519...	-0.14171...	-0.50557...	1.085220...	0.080015...	1.411344...	1.100041...	1.293049...	0.529626...	1.036848...	-1.22996...	-0.02460...	-1.38336...	-0.00089...	0.012102...	0.099949...	0.355773...	-0.73663...	1.253794...

Building the model – cross validation

Time series data:

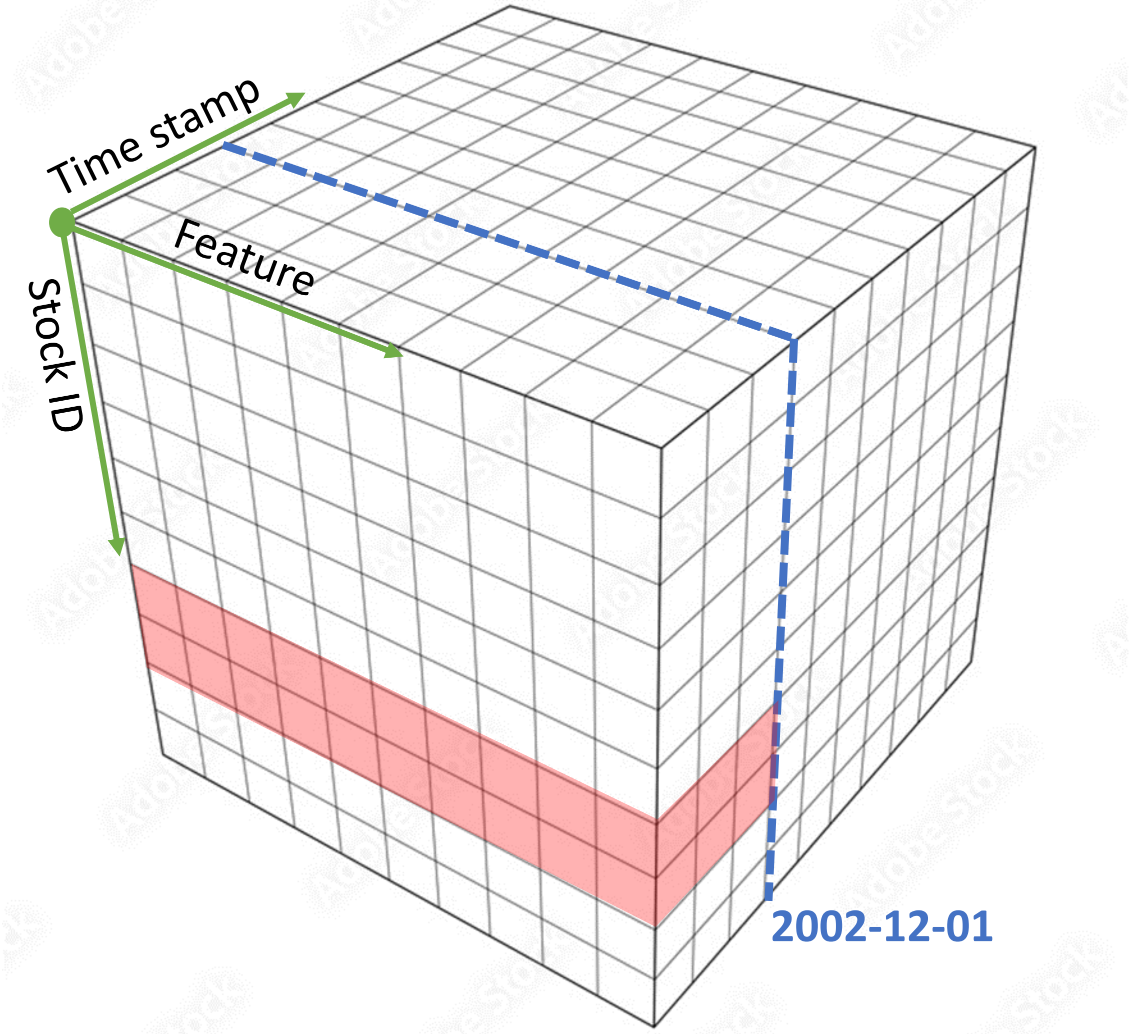
- Avoid looking into the future



Building the model – cross validation

Time series data:

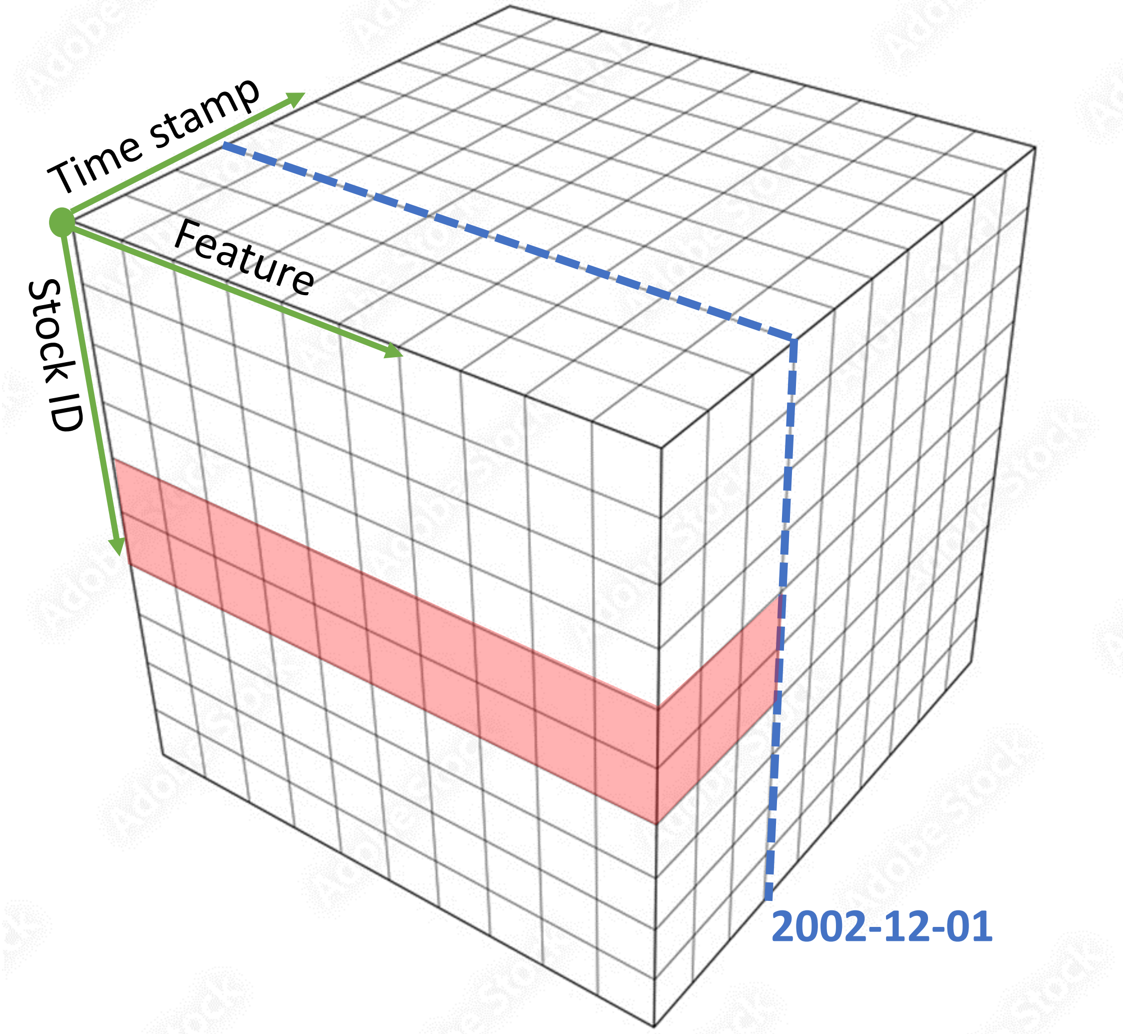
○ Avoid looking into the future



Building the model – cross validation

Time series data:

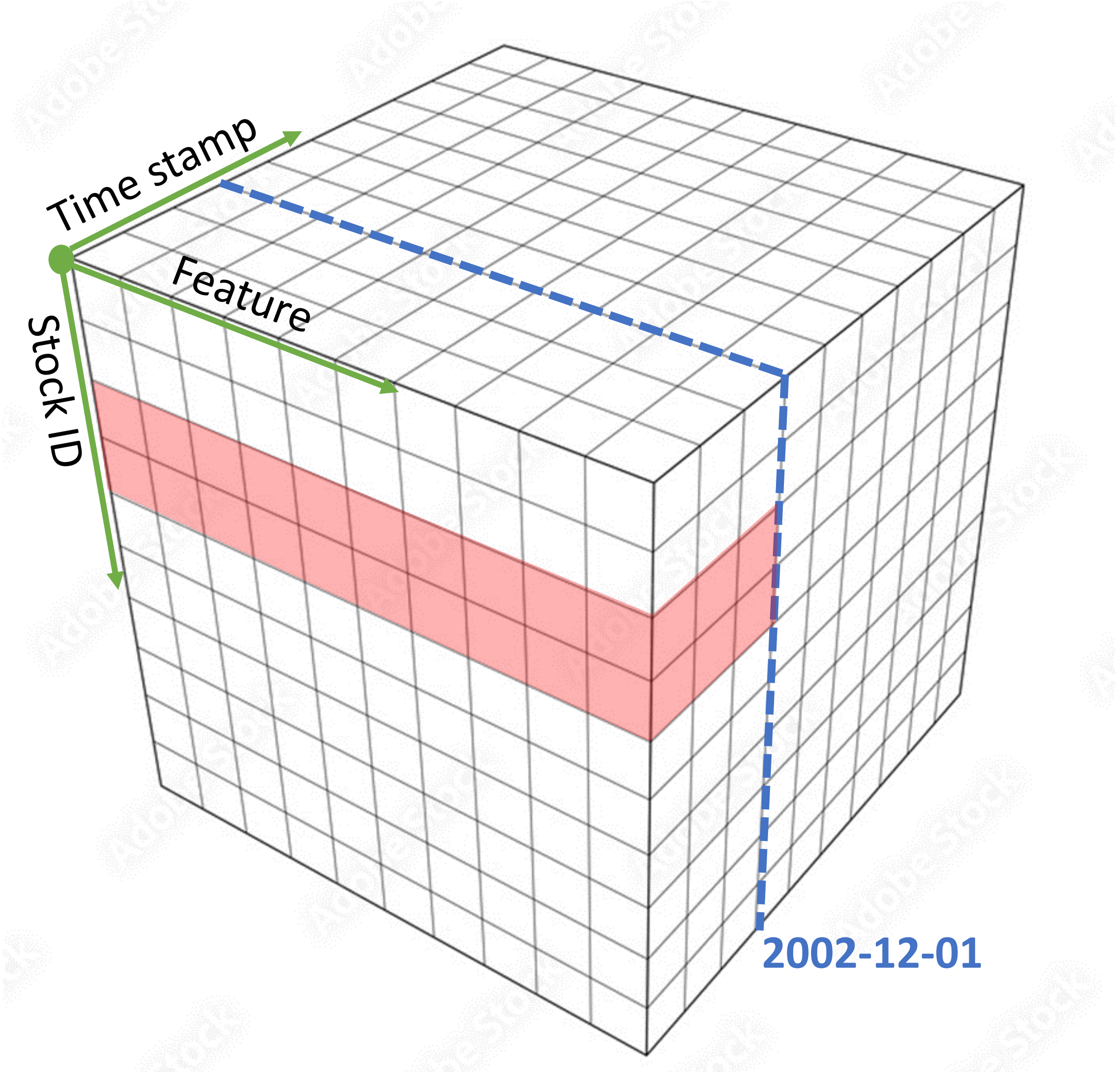
- Avoid looking into the future



Building the model – cross validation

Time series data:

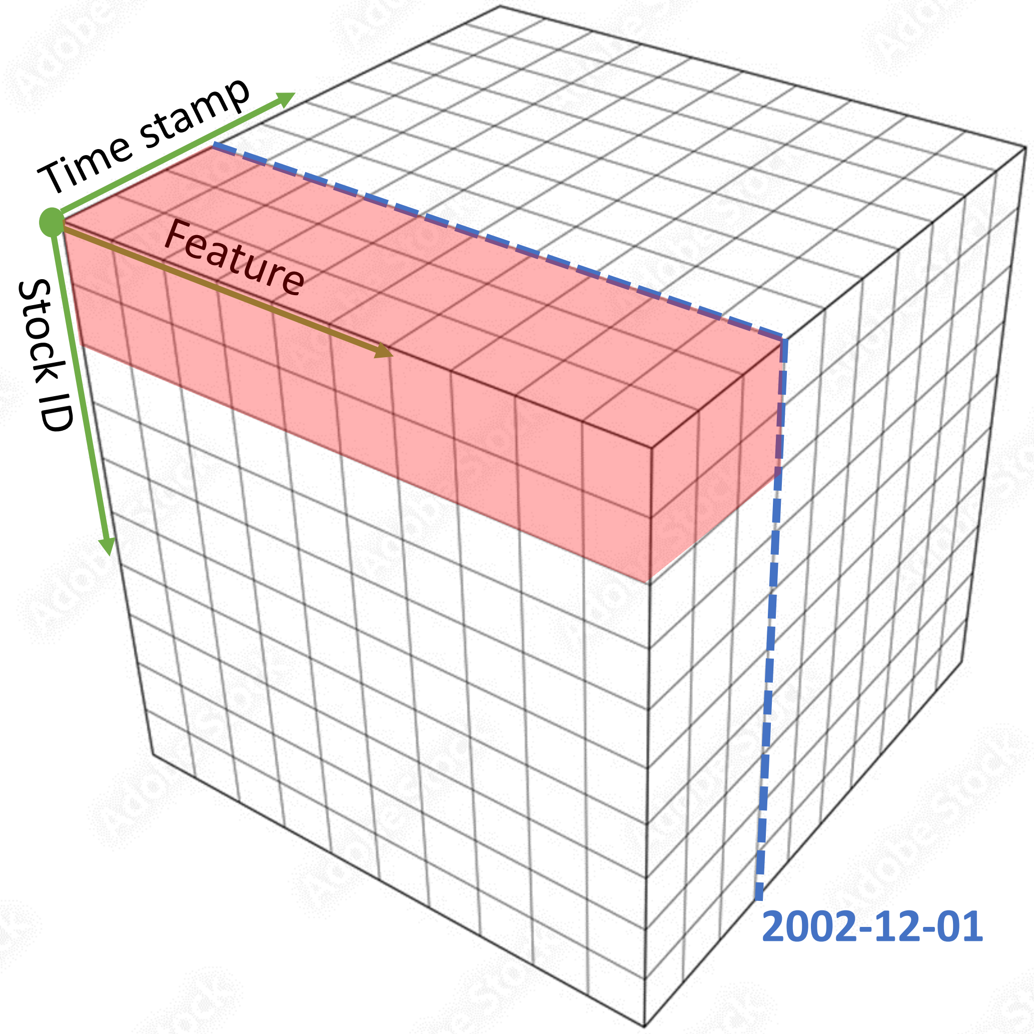
- Avoid looking into the future



Building the model – cross validation

Time series data:

- Avoid looking into the future



Building the model – hyperparameters

GRU:

- Number of inputs = 128

Used for the NN structure on top of GRU:

Four different methods:

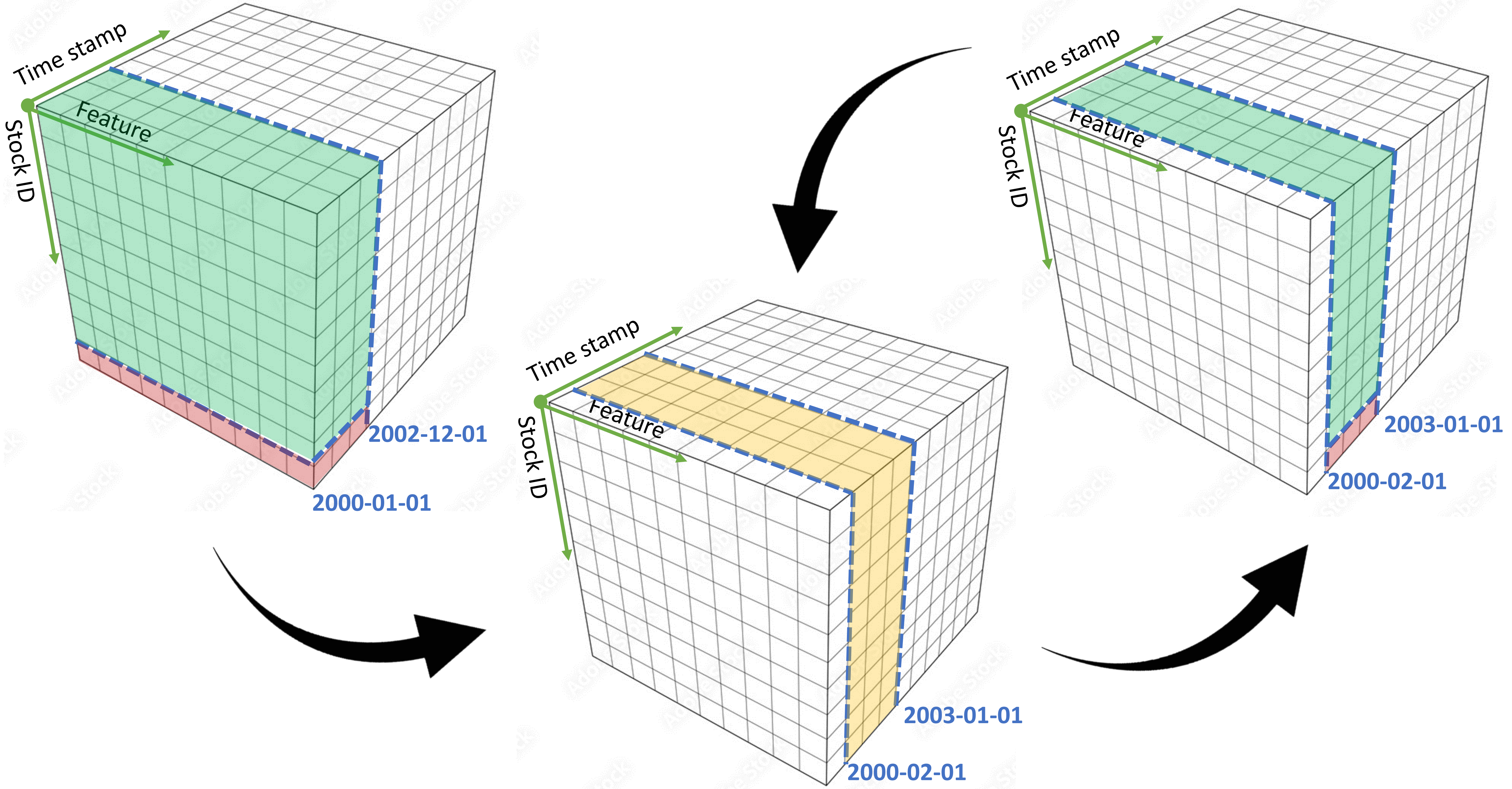
- Bayesian optimizer (CV) → (29, 28, 3)
- Hyperband (CV) → (26, 42)
- Rule of thumb (pyramid) → (38, 11, 3)
- No additional dense layers → Prediction

Training the model

Combine rolling window with 3D tensor in time series data.

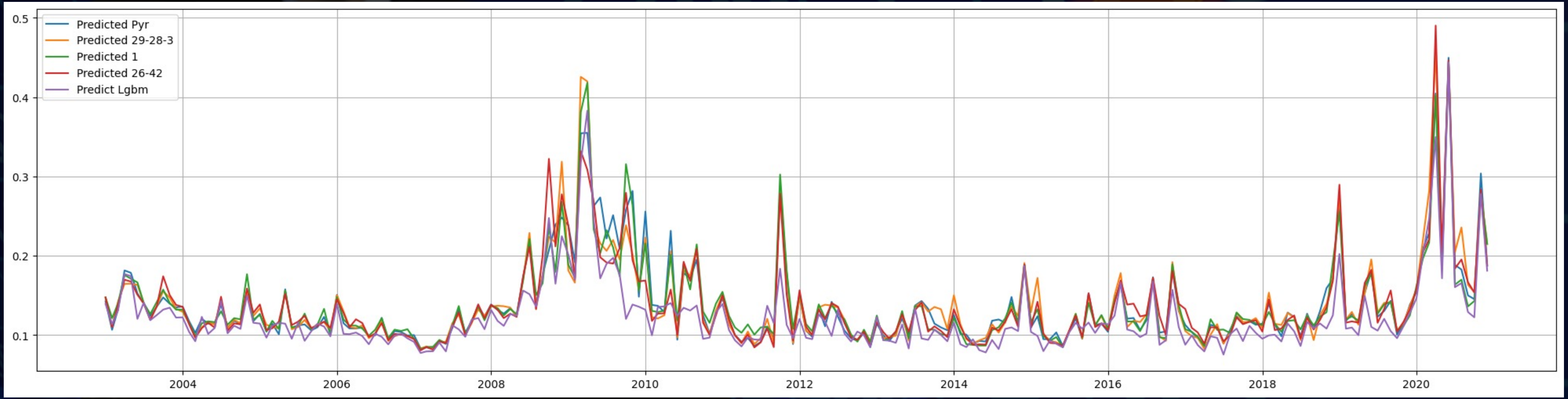
Method:

1) Train the model on 36 months with validation → Get parameters
2) Predict excess return of the 37th month → Get predictions
3) Shift the window one month forward and go to 1)



Time stamp
Feature
Stock ID
2002-12-01
2000-01-01
Time stamp
Feature
Stock ID
2003-01-01
2000-02-01
Time stamp
Feature
Stock ID
2003-01-01
2000-02-01

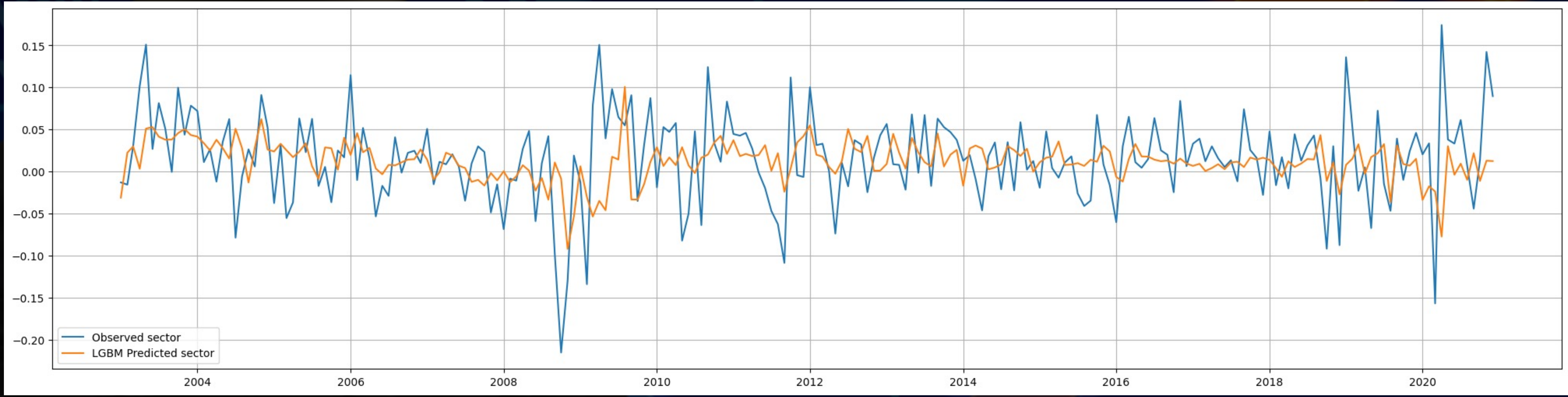
Predictions – rmse

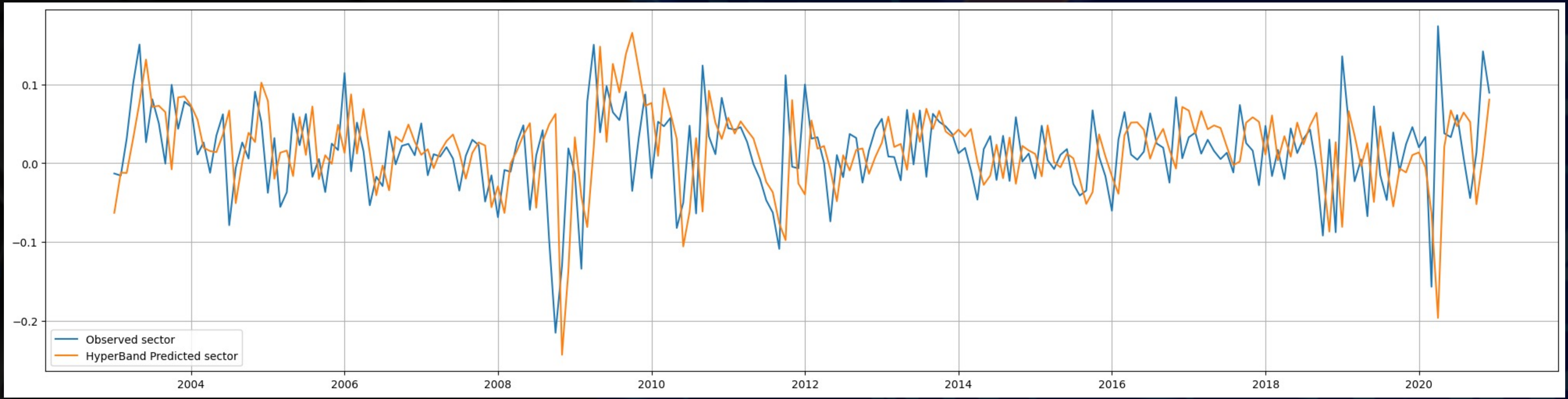


Predictions – which model is best?

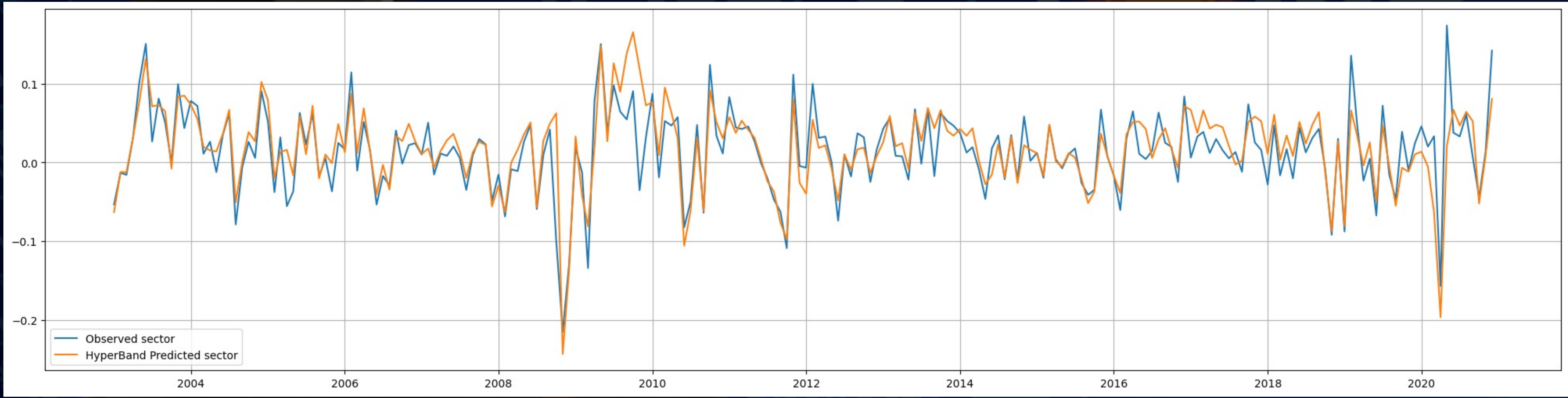
	RMSE entire period	RMSE good years	RSME bad years	
29-28-3 (Bayesian)	0.15181	0.12115	0.19092	
26-42 (Hyperband)	0.14871	0.12041	0.18526	Best NN
38-11-3 (Pyramid)	0.14940	0.11990	0.18726	
None (Only GRU)	0.14931	0.12042	0.18650	
LightGBM	0.13250	0.10922	0.16301	Best overall

Final predictions – grouped by sector

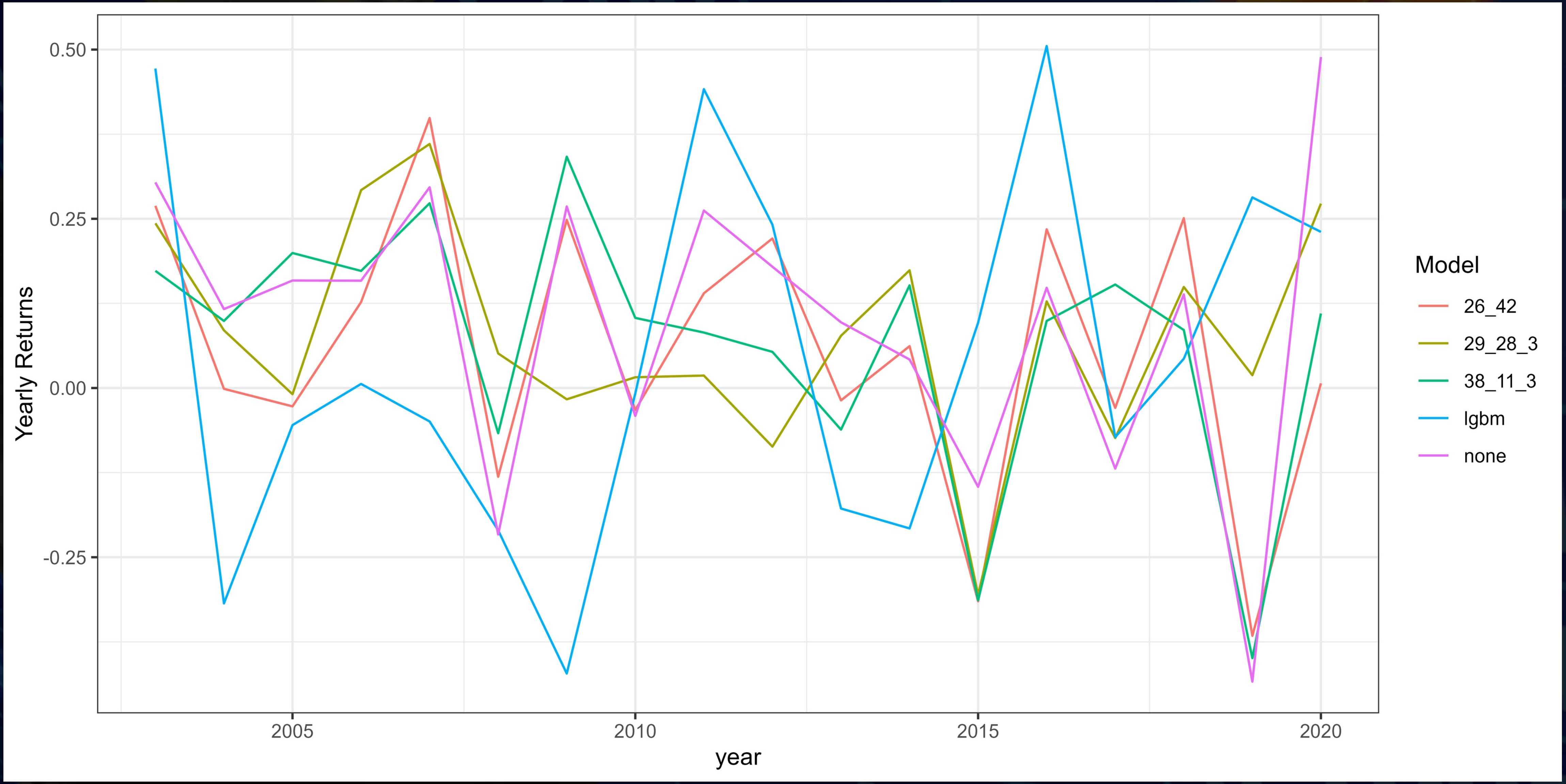




Final predictions – let us cheat and see what happens



Portfolios – can we create a profit?



Portfolios – can we create a profit?

$$Sharpe\ ratio = \frac{Avg.\ yearly\ return\ - riskfree}{volatility}$$

	None	29-28-3	26-42	38-11-3	LightGBM
Avg. yearly return	9,46%	7,75%	5,76%	6,98%	4,43%
Max return	48,9%	36,05%	39,88%	34,18%	50,53%
Min return	-43,38%	-30,56%	-36,61%	-39,91%	-42,18%
Volatility	21,96%	15,96%	20,19%	18,39%	27,8%
Sharpe ratio	0,43	0,49	0,29	0,38	0,16

What is next?

- Try LSTM vs. GRU
- Try XGBoost for time series
- Do yearly feature selection
- Do yearly hyperparameter optimization

Appendix

Feature selection

Removed the following 17 covariates based on Spearman correlation. These were all less than 0.25% correlated to the target variable.

```
characteristic_pchsaleinv, characteristic_hire, characteristic_sgr, characteristic_cashpr,
characteristic_chinv, characteristic_securedind, characteristic_chpmia, characteristic_absacc,
characteristic_bm, characteristic_pchsale_pchrect,
characteristic_pricedelay, characteristic_pchquick, characteristic_mom12m,
characteristic_pchgm_pchsale, characteristic_realestate,
characteristic_pchsale_pchxsga, characteristic_lgr, macro_de, characteristic_cinvest
```

Model training

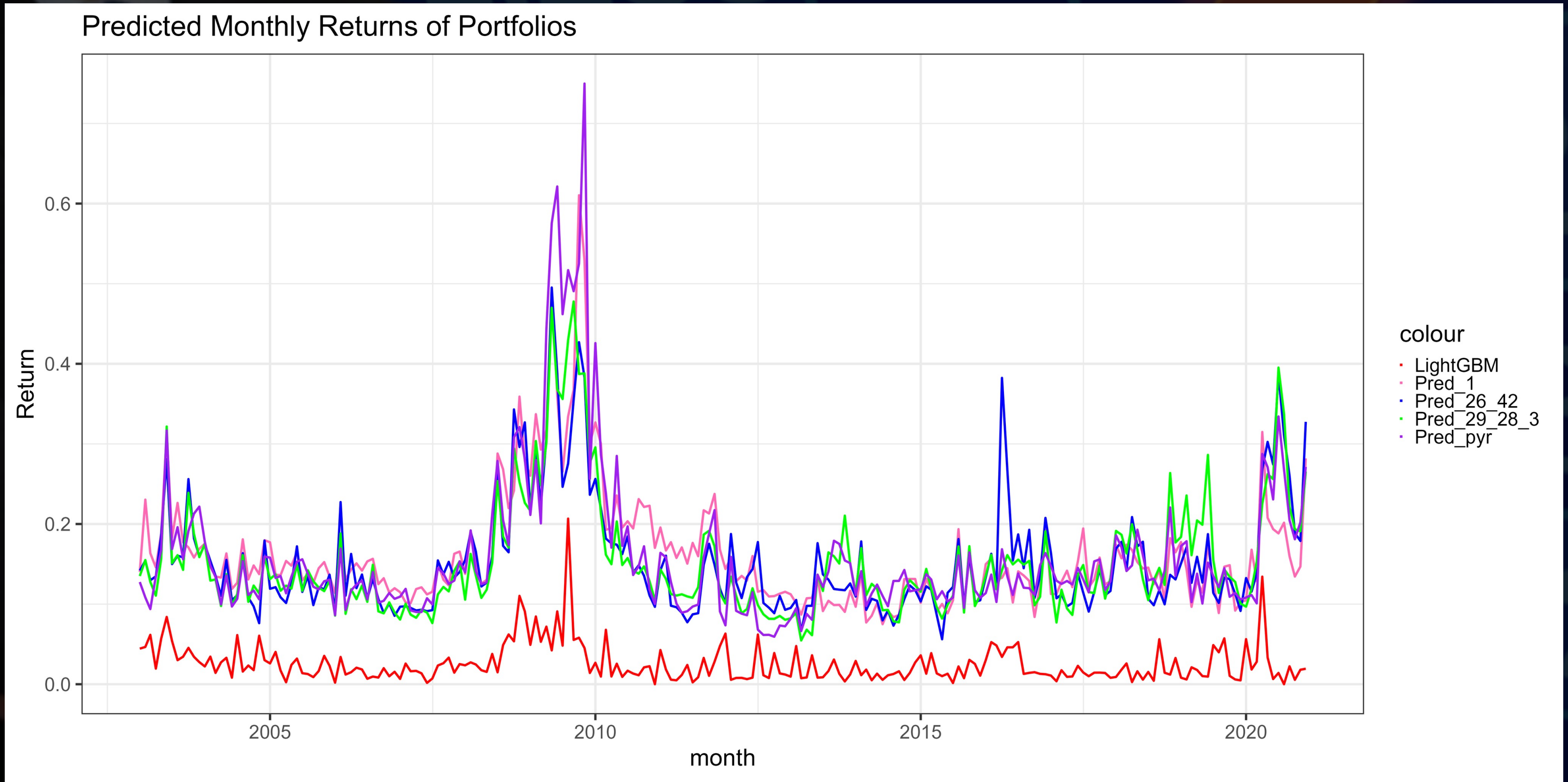
	Light GBM	None	29-28-3	26-42	Pyr
Input Features	All except the ones with least corr.	All except the ones with least corr.	All except the ones with least corr.	All except the ones with least corr.	All except the ones with least corr.
HP Optimization	Naïve approach	None	Bayesian	Hyperband	Pyramid rule of thumb
Hyperparameters	N-epochs: earlyStopping N-estimators: 120 Learning rate: 0.005 Num leaves: 70 Min child samples: 10	N-epochs: earlyStopping Batch size: 429 Optimizer: Adam Learning rate: 0.003 Num dense layers: 0 Neurons: 0	N-epochs: earlyStopping Batch size: 429 Optimizer: Adam Learning rate: 0.003 Num dense layers: 3 Neurons: 29-28-3	N-epochs: earlyStopping Batch size: 429 Optimizer: Adam Learning rate: 0.003 Num dense layers: 2 Neurons: 26-42	N-epochs: earlyStopping Batch size: 429 Optimizer: Adam Learning rate: 0.003 Num dense layers: 3 Neurons: 38-11-3
RMSE	0.1324995	0.1493138	0.1518063	0.1487109	0.1494005
Run time (HP optim. time + training time)	11 min	~200 min	~70 min + 200 min	~370 min + 200 min	~300 min
Comments	Often just predicts in the middle. Poor performance.	Very high volatility	Relatively easy to fit but mediocre performance.	Lowest RMSE out of the NN's. Only NN with two layers and not three, but took very long.	Seems to be pretty good even though HP are just chosen from a rule of thumb

Model training

In the lightGBM, we tried experimenting with ”cutting” the validation set out horizontally and vertically, i.e. either as a chunk of time or a chunk of stocks. It turned out to make a very big difference and we ended up choosing the stock index method. Here are some plots to show the difference.

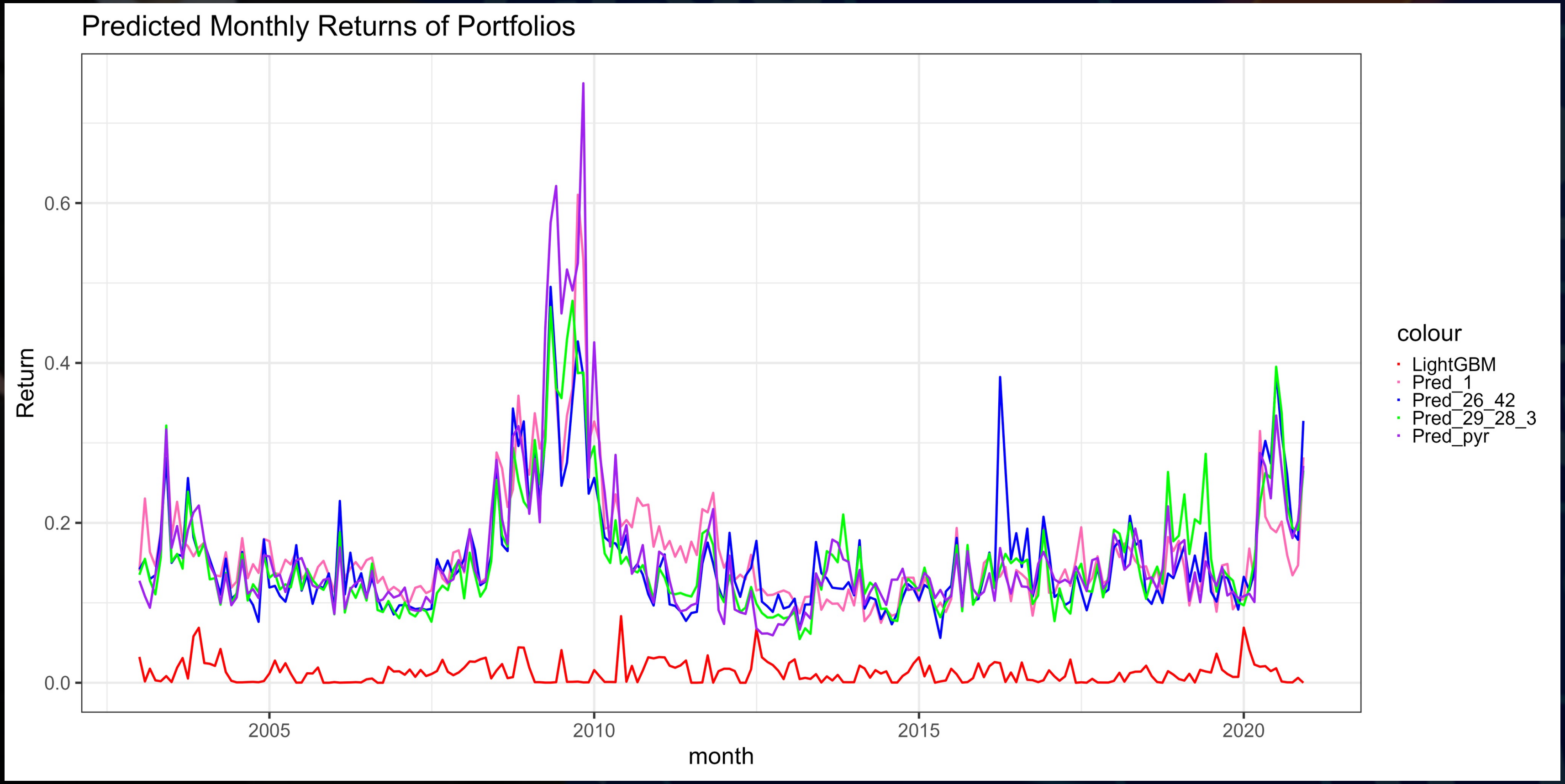
Model training

- Cutting by stock index for LightGBM, the red line, and it follows the trend approximately but does not predict anywhere near as high as the NN's. All NN's are cut by stock index always.



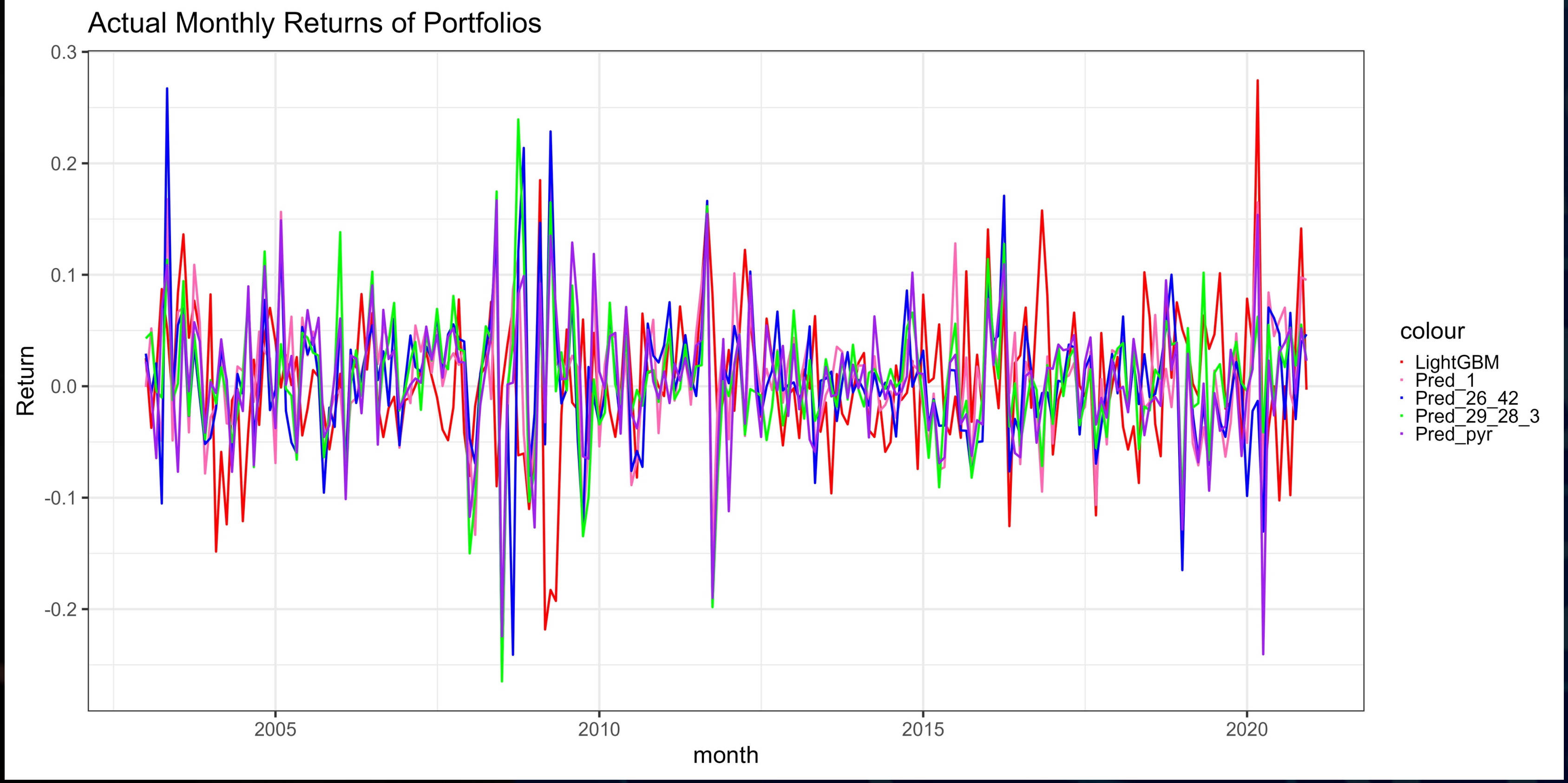
Model training

- Cutting by time index. The red line, lightGBM, flatlines a lot and does not reach any high or low peaks. All NN’s are cut by stock index always.



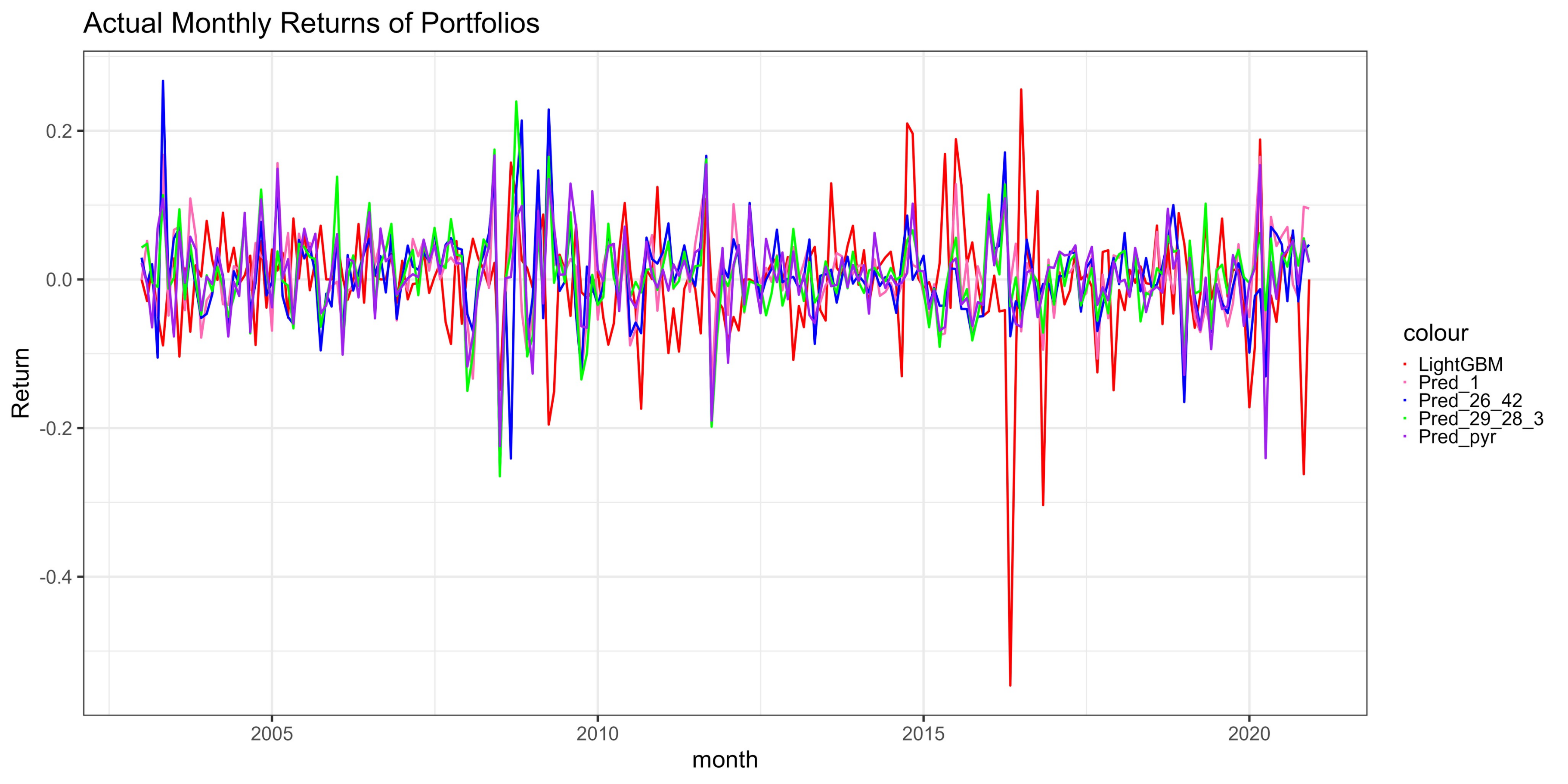
Model training

- Cutting by time index. No tendency, but still very different from cutting the other way.



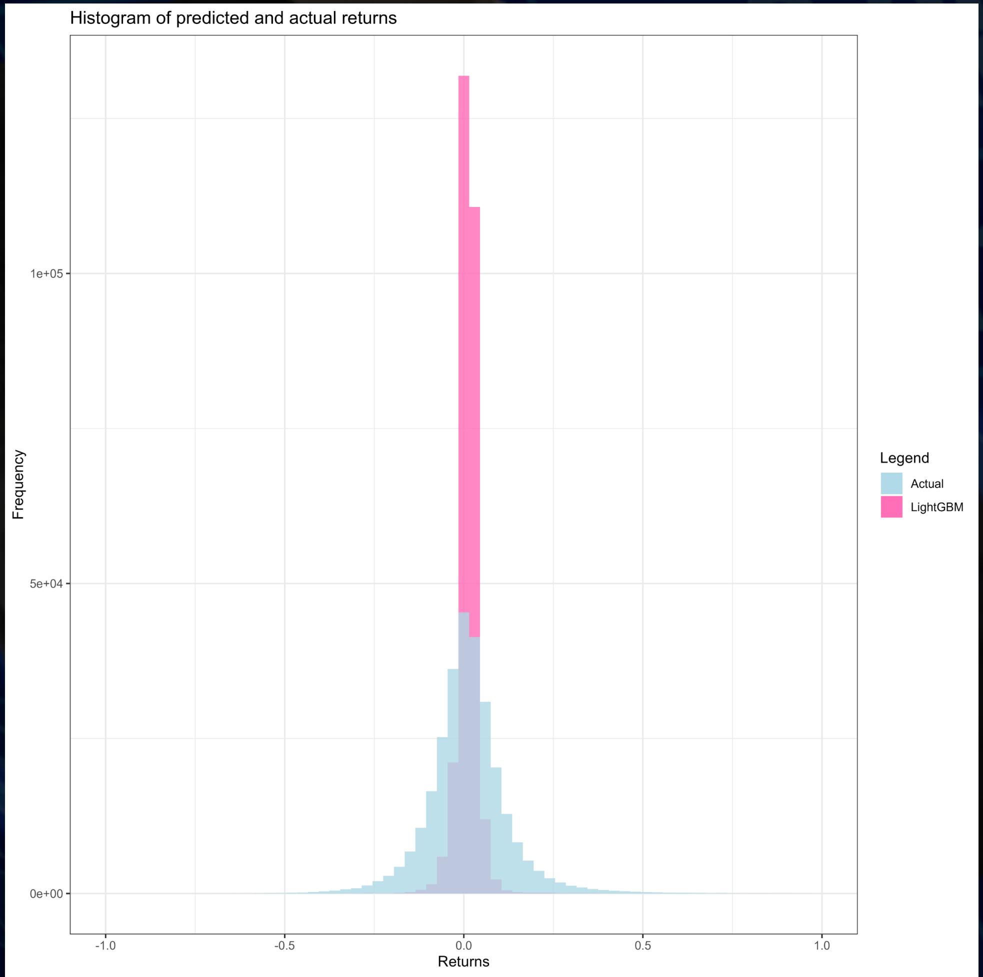
Model training

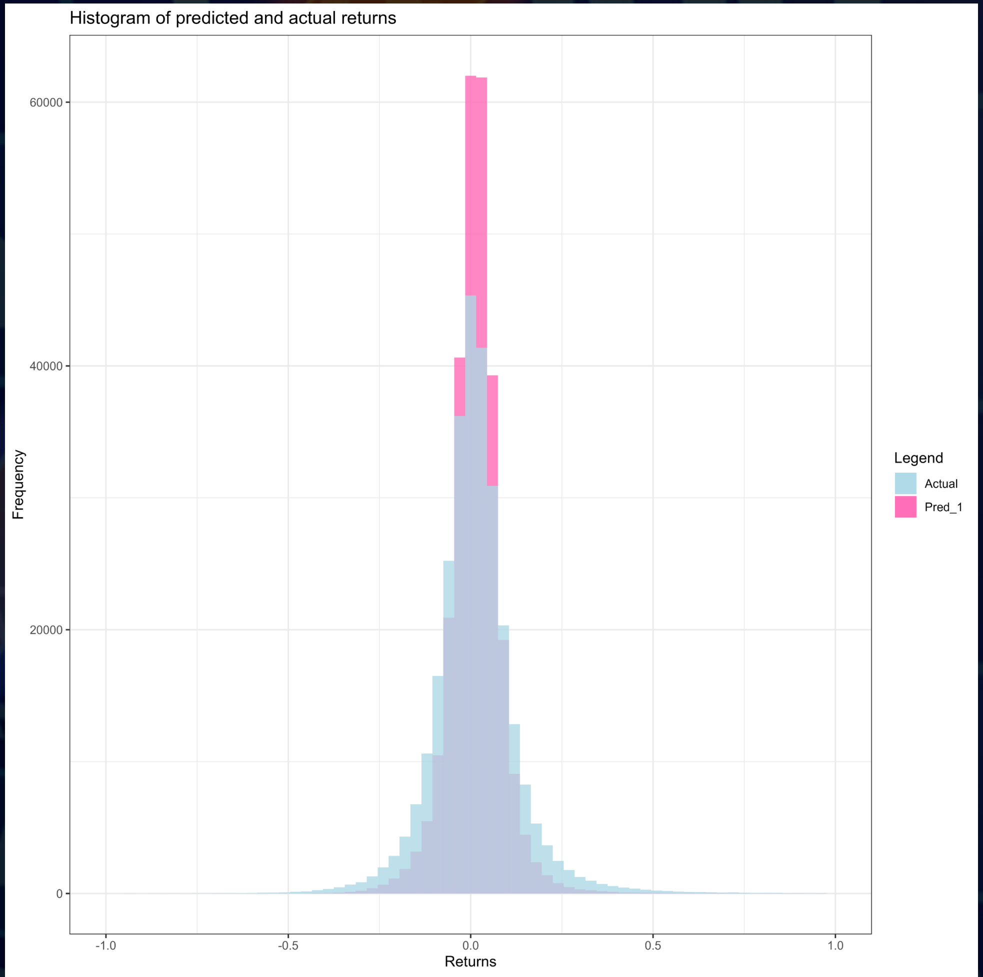
- Cutting by stock index.



Model evaluation

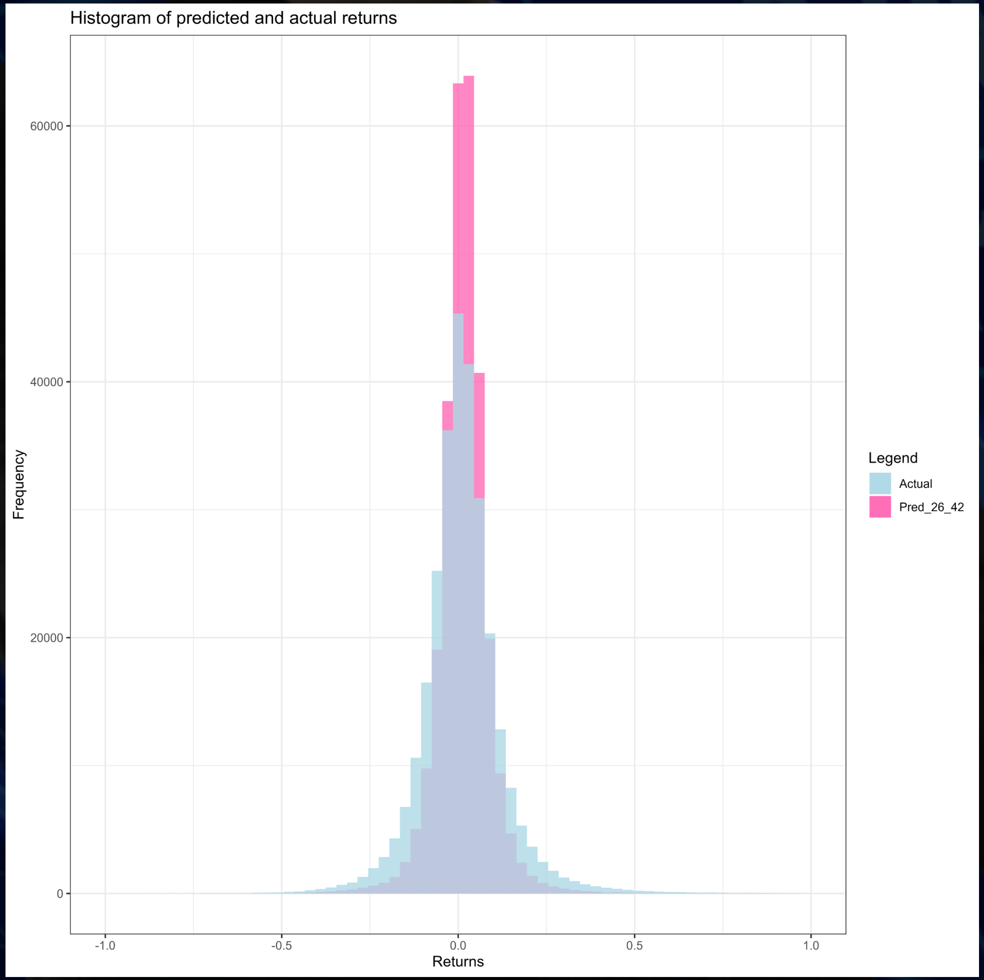
- We try to plot the frequency of each predicted value for each model.

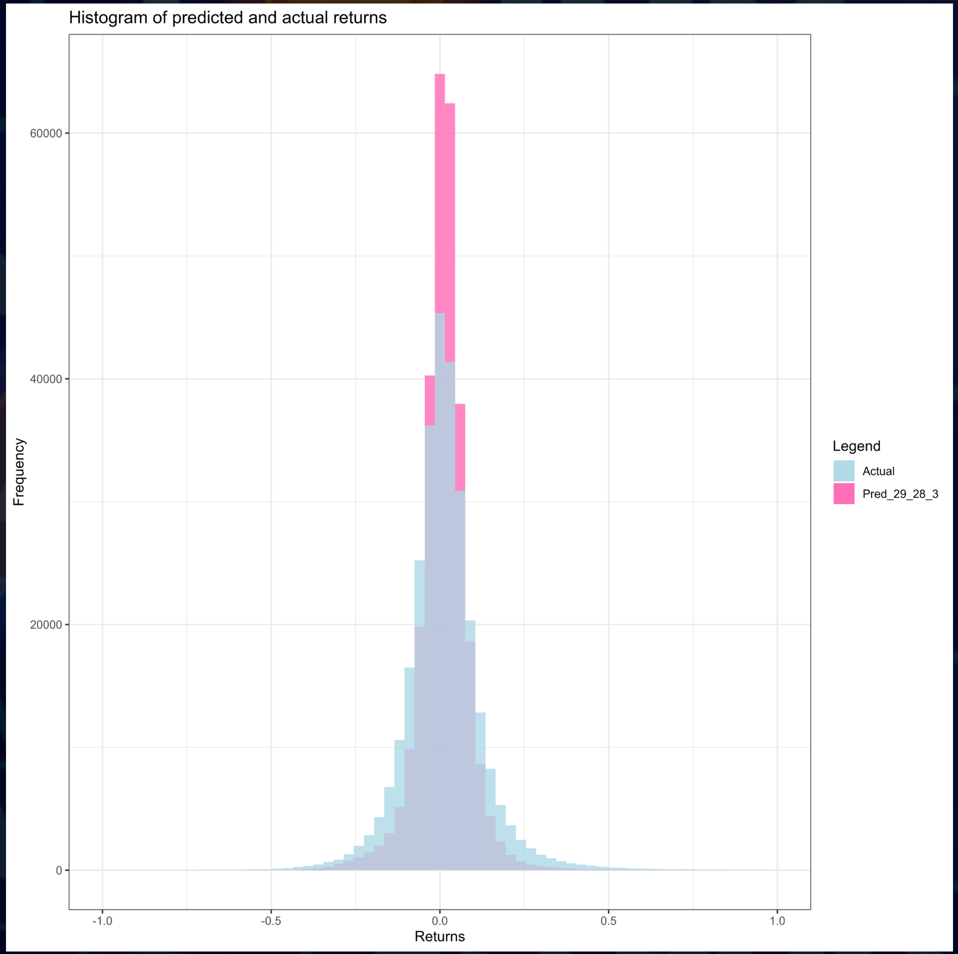




Model evaluation

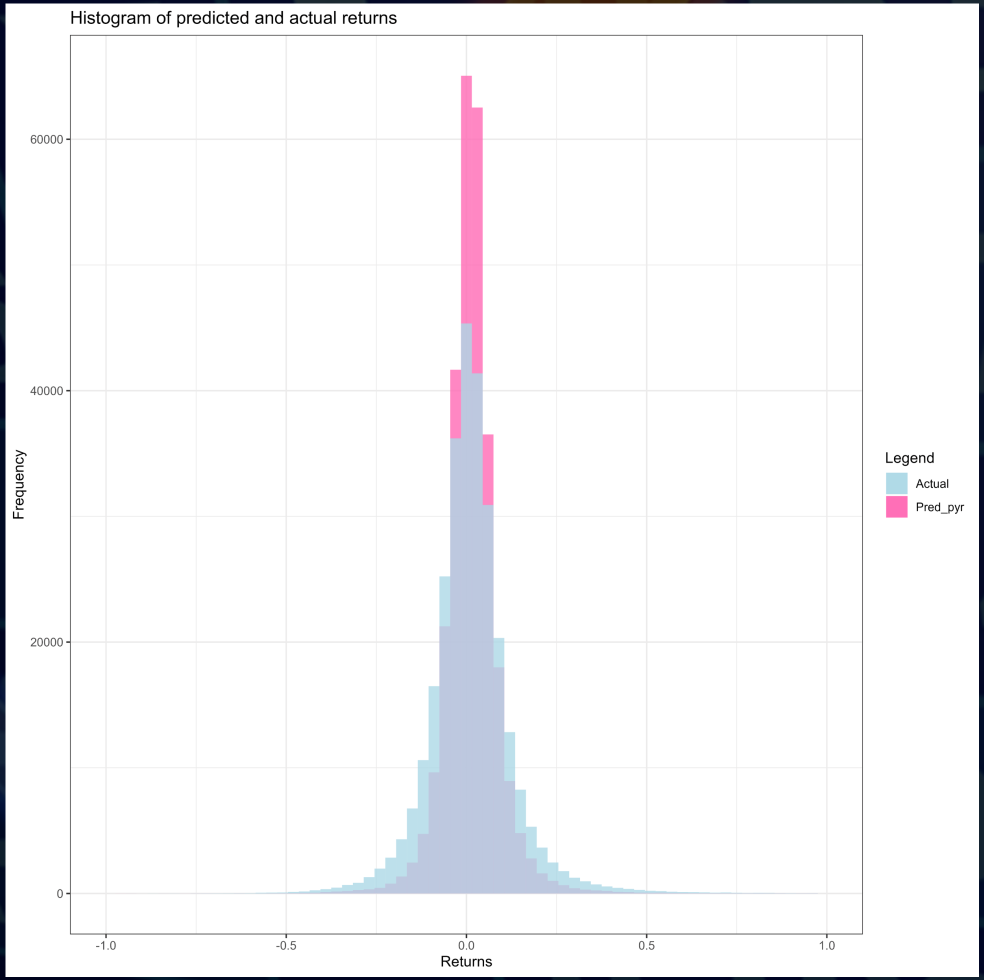
- We try to plot the frequency of each predicted value for each model





Model evaluation

- We try to plot the frequency of each predicted value for each model. All of the NN's look pretty much identical on these plots, but the lightGBM predicts 0 quite often compared to the rest. If we want to test the difference between the NN's, we need to look a numbers as the difference is not big enough to show in these plots. We have omitted a few extreme values from the histograms but these would not have made a difference in the overall conclusion



Model evaluation

- These are the residuals for each model