Background

Numerai Signals is the first step to achieving phase 2 of Numerai's master plan: monopolize data. The goal of Numerai Signals is for Numerai to source signals that are orthogonal to the signal that we already have.

Numerai recently released three new targets and four new scores to help data scientists train models which produce more valuable signals. If you haven't already, please read through the detailed descriptions of each target and score in the post above before continuing with this one.

We have just updated Signals Diagnostics to include these new scores (FNCv4, RIC, CORRv4, and ICv2), along with new metrics (mean, standard deviation, Sharpe, and max drawdown) to give you more granular insight into how your model performs on historical data. We believe this kind of feedback is important for guiding your signal research in the right direction.

In this post, we will walk you through how to use the new scores in the Diagnostics tool to help improve your Signals models. In particular, we will try to improve our FNCv4 (since it will soon replace CORR for payouts) and RIC (as a way to indirectly improve our TC).

Using the Diagnostics Tool to Improve your Signals Models

Let's start by submitting example predictions to Diagnostics to establish a baseline.Example predictions are created with Yahoo finance data, using a simple gradient boosting regressor trained on target_20d. For the rest of this walkthrough, we will be using the same model and features but switching out the target that we will be training on.

model.fit(train[feature_names], "target_20d")

As you can see, the FNCv4 Sharpe is 0.62 and the RIC Sharpe is 0.41.

[

1482×1096 110 KB

](https://forum.numer.ai/uploads/default/original/2X/f/f951e3e8eaee2162de15baf6258095179a0794f9.png)

Improving FNCv4

From the previous post we learned that FNCv4 will soon replace CORR for payouts. So let's try to improve our FNCv4 first.

Since FNCv4 is based on target_20d_factor_feat_neutral, we can train our model on this target. And since CORRv4 is also based on this same target, we expect our CORRv4 to improve.

model.fit(train[feature_names], "target_20d_factor_feat_neutral")

As expected, we see a big improvement in our FNCv4 (Sharpe increased from 0.6281 to 0.7242, max drawdown decreased from -.1547 to -0.0766) and CORRv4 (Sharpe increased from 0.5223 to 0.6037, max drawdown decreased from -0.15 to -0.0877).

[

1488×1098 109 KB

](https://forum.numer.ai/uploads/default/original/2X/5/596ea59c8750de9311d7b243ab4d19e99cf5829b.png)

Improving RIC

From the previous post, we also learned that each of these new scores are all dissimilar from one another while being decently correlated with TC. This means that improving in one or more of these scores may indirectly help improve our model's TC.

[

1206×462 125 KB

](https://forum.numer.ai/uploads/default/original/2X/d/de7482af2662dac2ec92c9f4d59c88d9acd65987.png)

In this example, let's try to improve RIC. Since we know that RIC is based on target_20d_factor_neutral, we can train our

model on this target instead just like in the example above.

model.fit(train[feature_names], “target_20d_factor_neutral”)

As expected, we see a major improvement in RIC Sharpe (from 0.4136 to 0.4770) but worse max drawdown in every score relative to the baseline. One reason why we may be getting worse max drawdowns is because target_20d used in the baseline is neutral to both factors and features whereas target_20d_factor_neutral is only neutral to factors and not features.

Perhaps surprisingly, FNCv4 Sharpe also improved against the baseline and even against the FNCv4 example above. The lesson learned here is that oftentimes, training on these other targets may help with the other scores in surprising ways even if they are not directly related.

[

1474×1096 109 KB

](https://forum.numer.ai/uploads/default/original/2X/0/0c027850090bac768aa3c7d722da0b56e4044b4a.png)

Ensembling

Is there a way to improve FNCv4 and RIC at the same time without making our drawdowns worse? One way we try to achieve this is to train on both of the new targets (target_20d_factor_neutral for FNCv4 and target_20d_facor_feat_neutral for RIC) and ensembling them together.

ffn_model = GradientBoostingRegressor(subsample=0.1) fn_model = GradientBoostingRegressor(subsample=0.1) ffn_model.fit(train[feature_names], train["target_20d_factor_feat_neutral"]) fn_model.fit(train[feature_names], train["target_20d_factor_neutral"])

val["ffn_preds"] = ffn_model.predict(val[feature_names]) val["fn_preds"] = fn_model.predict(val[feature_names])

rank per date for each prediction column so that we can combine safely

val[["ffn_preds", "fn_preds"]] = val.groupby('date').apply(lambda d: d[["ffn_preds", "fn_preds"]].rank(pct=True))

combine

val["ensemble_preds"] = sum([val["ffn_preds"], val["fn_preds"]]).rank(pct=True)

[

1482×1094 109 KB

](https://forum.numer.ai/uploads/default/original/2X/b/b3eecea687e188bb9ed601d664e325d743e7ae7c.png)

With this ensemble model, the RIC max drawdown goes down dramatically from -0.43 to -0.22 and both are RIC and FNCv4 Sharpes are significantly higher than the original example predictions.