DataRobot

Time Series Modeling

DataRobot

Agenda

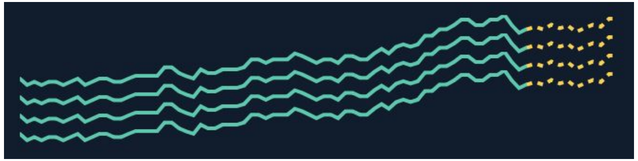
- Kicking off time aware modeling in DataRobot
- Exploring model families
- Evaluation and insights
- Methods to improve results
- Time aware modeling with the Python API

Attributes of Single vs. Multiseries Modeling

Single series models

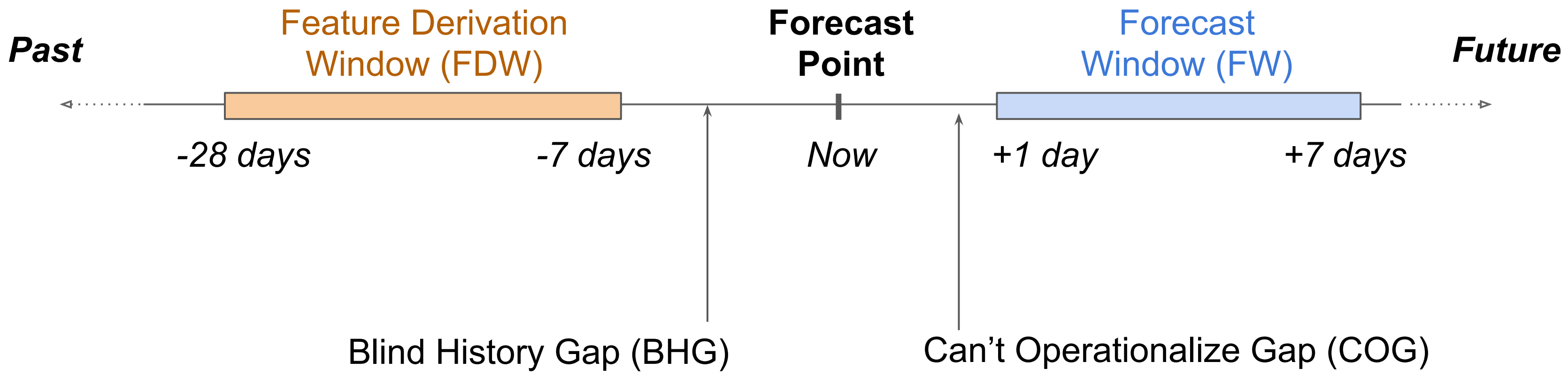
- Pros
    - All aspects of the model are estimated for the exact series of interest
- Cons
    - Feature selection is limited to just what the individual series has encountered
    - Scale - if you have >100 products, the number of models is prohibitive

Multiseries models



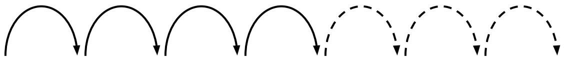
- Pros
    - The model often learns features that single series models miss
    - Scale - in DataRobot we can model 100K series per project
    - We can also add cross-series features under advanced settings
- Cons
    - The best result is selected for the aggregate, not for any one series

General Time Series Framework



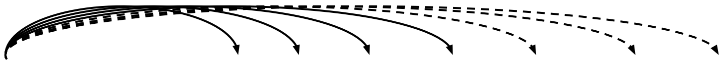
Time Series Modeling Families

Integrated Models



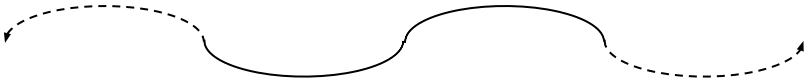
ARIMA, ETS, RNNs, etc.

Per Forecast Distance Models



Xgboost, elastic-net, etc.

Trends and Decomposition Models



Fourier models, linear trends,
decompositions, etc.

The forecast distance approach duplicates each row per forecast distance

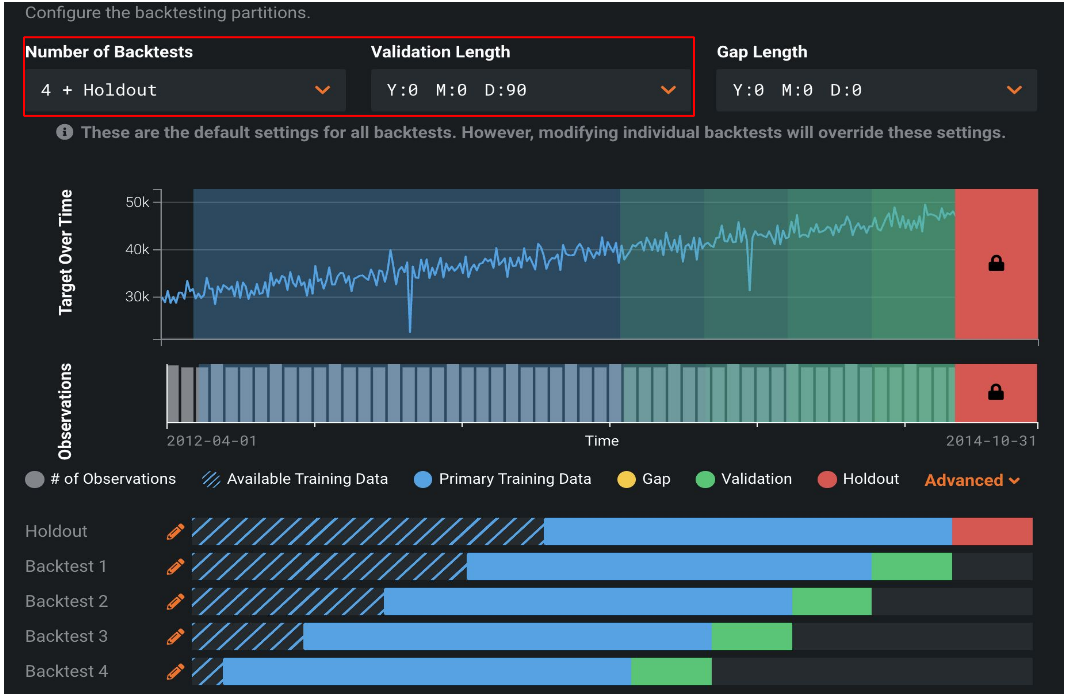
Target | "Anchors" | Lag features

Date	Store	Marketing	Store_size	Sales	Forecast_Point	Forecast_Distance	7_Day_Mean	14_Day_Min	...
2017-12-07	Baltimore		14,000	53,338	2017-10-03	65	25146	5621	...
2018-01-24	Columbus	In Store...	17,000	33,402	2017-12-10	45	4562	1254	...
2018-08-31	Baltimore	Summer Campaign...	14,000	27,766	2018-05-01	122	2654	656	...
2019-04-17	Lancaster		13,500	62,779	2018-10-08	191	25698	6569	...
2019-04-18	Savannah	July Sale..	16,000	17,771	2019-04-08	10	6542	859	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
2020-04-30	Richmond		9,000	41,537	2019-04-30	365	36958	6527	...

Sampling may occur to fit the expanded dataset in memory!

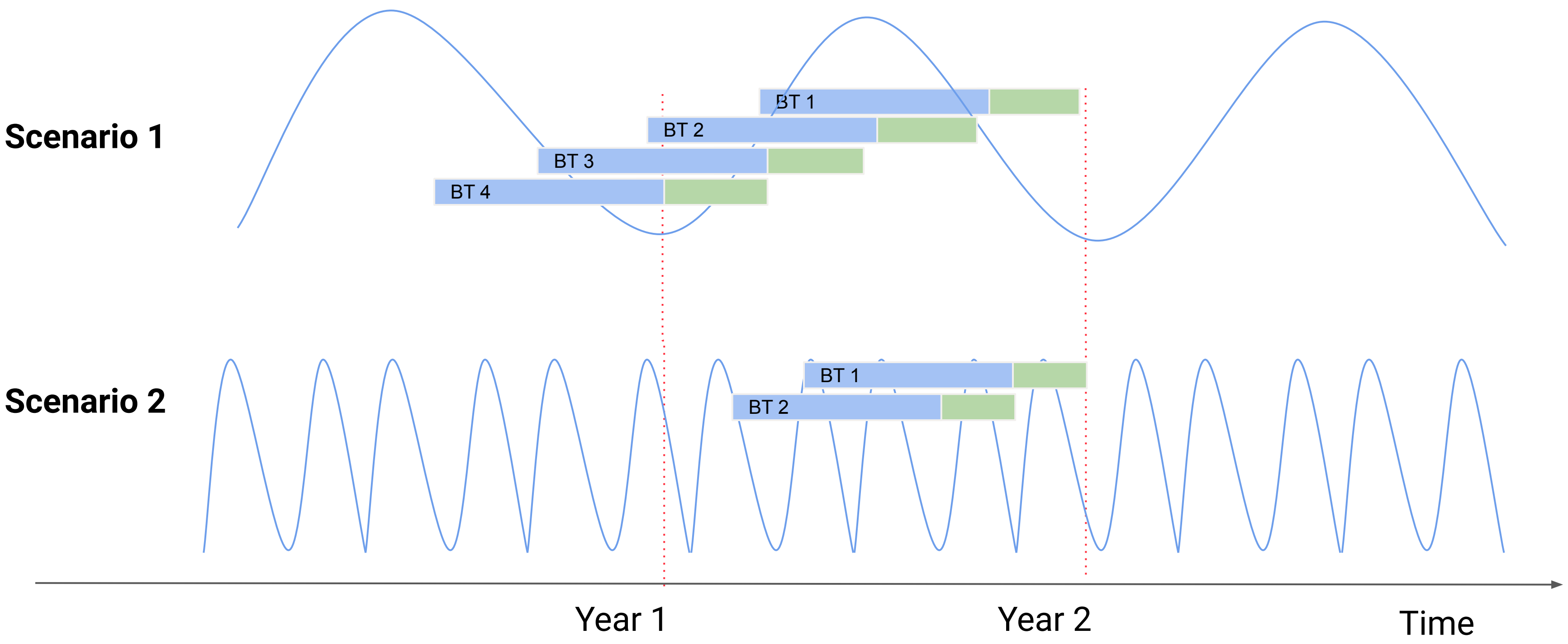
Backtests | Best Practices

- Set the length of an individual backtest to the time you expect the model to remain in production without retraining
- Set the length of an individual backtest greater than or equal to the max FD
- Set the range of all backtests to span at least one full cycle
- Set the OTV gap to the time it takes to deploy a model and start making predictions



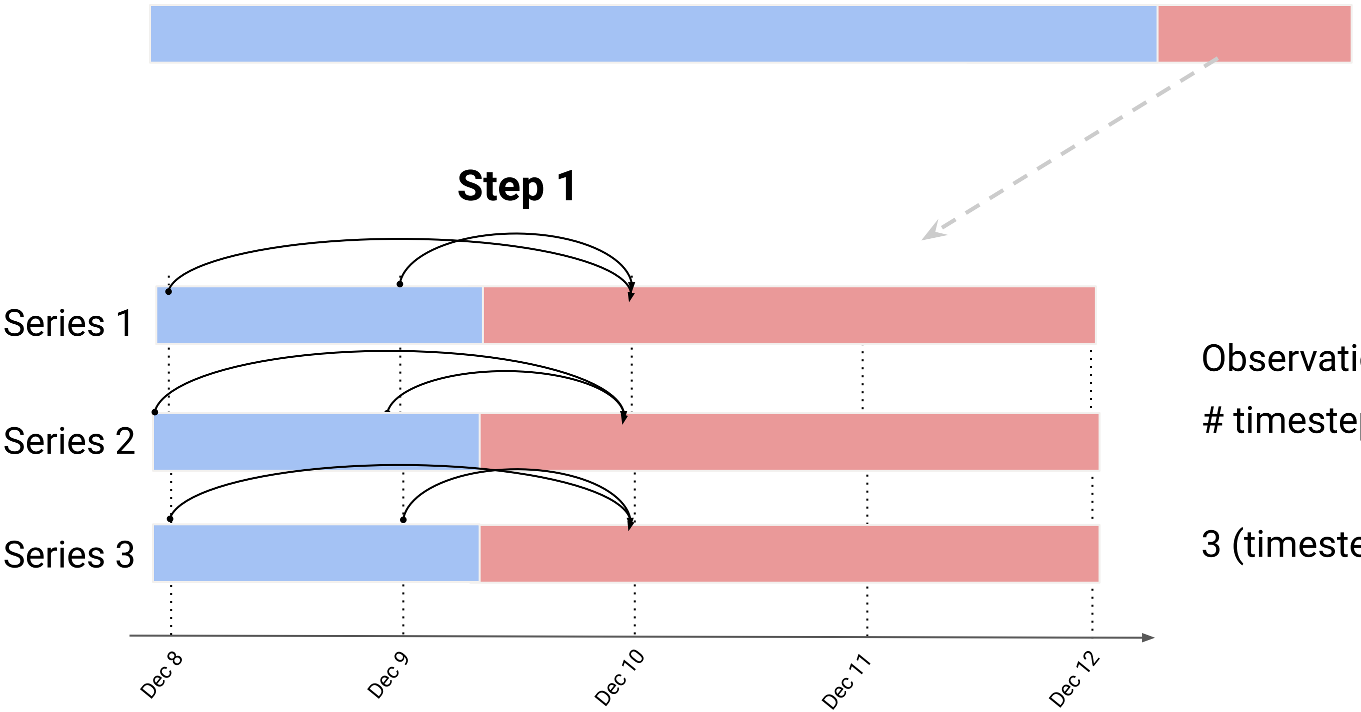
Start by choosing the length of an individual backtest, and then decide how many backtests to use

Backtests | Example Configuration



Validation and Holdout | Closer Look

Each datetime in the holdout or validation has one prediction per forecast distance



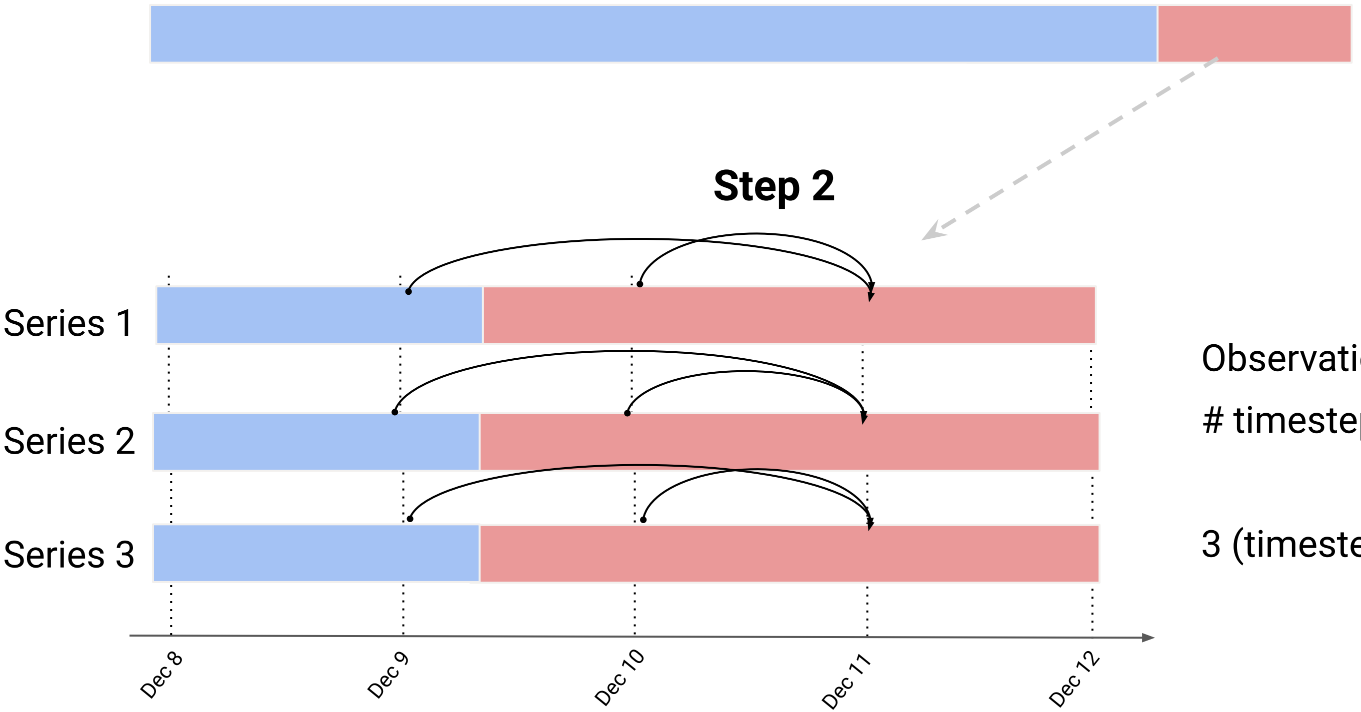
Observations in backtest =

timesteps x # FDs x # series

3 (timesteps) x 2 (FD) x 3 (series) = 18

Validation and Holdout | Closer Look

Each datetime in the holdout or validation has one prediction per forecast distance



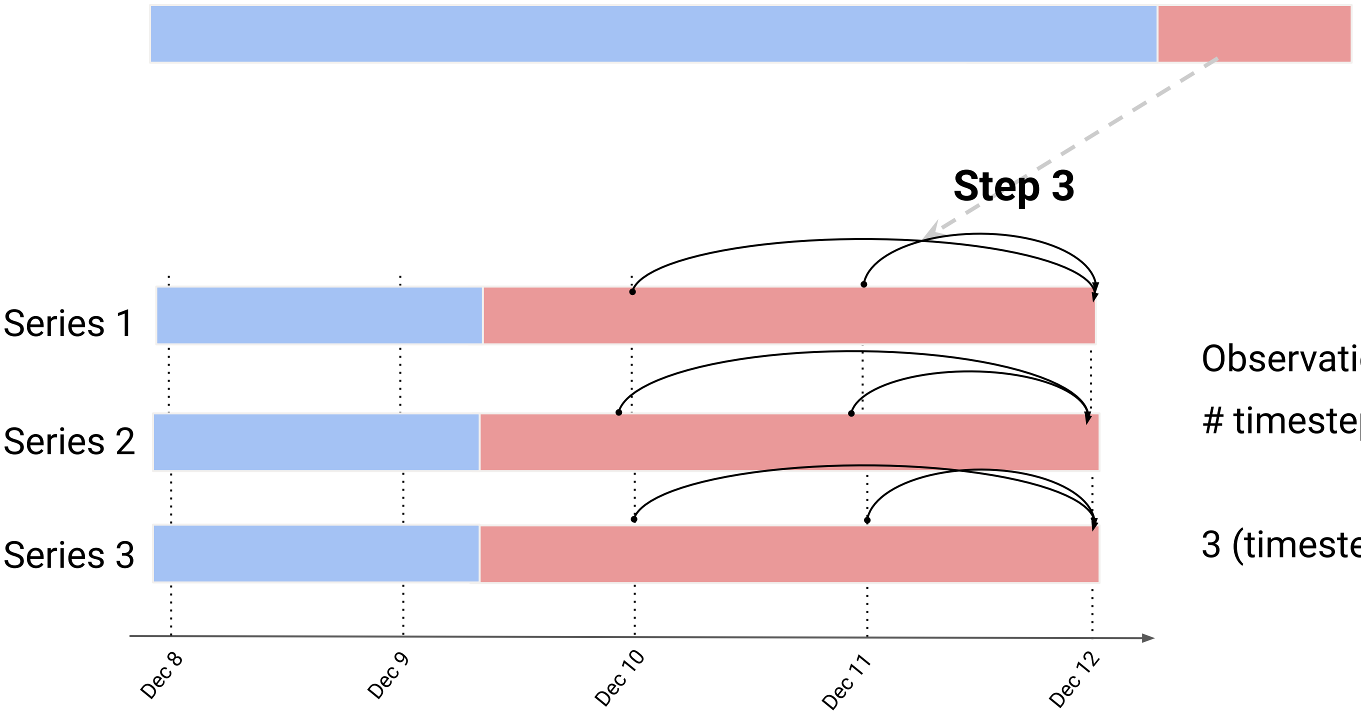
Observations in backtest =

timesteps x # FDs x # series

3 (timesteps) x 2 (FD) x 3 (series) = 18

Validation and Holdout | Closer Look

Each datetime in the holdout or validation has one prediction per forecast distance



Observations in backtest =

timesteps x # FDs x # series

3 (timesteps) x 2 (FD) x 3 (series) = 18

Hierarchical Models | Details

Target variable

Single BP

1

Date	Sales	SKU	Promotion
Monday	1,000	A	1
Monday	2,000	B	0
Tuesday	4,000	A	0
Tuesday	1,000	B	1

2

Total model

Date	Sum Sales
Monday	3,000
Tuesday	5,000

Proportion model

Date	Percent_of _sales	SKU	Promotion
Monday	.33	A	1
Monday	.66	B	0
Tuesday	.8	A	0
Tuesday	.2	B	1

3

$$proportion \times total\ sales = predictions$$

Now that we've built a project, how can we improve performance?

1) Clustering series
2) Split Projects by forecast distance
3) Add Calendar with important events
4) Feature Reduction
5) Blend Models

Series Clustering

Grouping series into different clusters and building separate projects can help by:

- Selecting different blueprints
- Applying different differencing strategies (e.g. latest, average, etc.)
- Minimizing different loss functions (e.g. poisson, gamma, gaussian, etc.)
- Reducing the sampling rate
- Choosing longer/shorter feature derivation windows
- Selecting different feature lists (e.g. DR's reduced features)

Note: DataRobot does apply clustering *within* projects. Look for similarity and performance clustered blueprints (you may have to run them from the repository)

Series Clustering | Automation at Work

Example

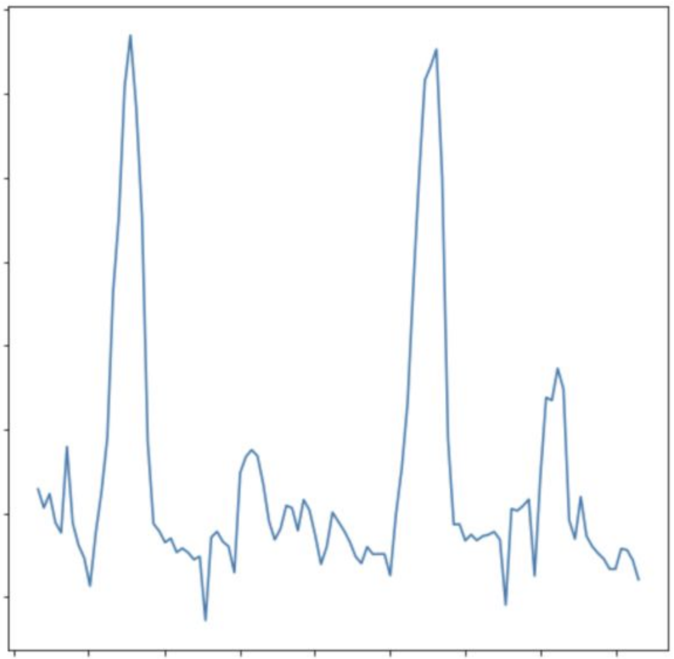
Look at the top model types and feature lists across clusters:

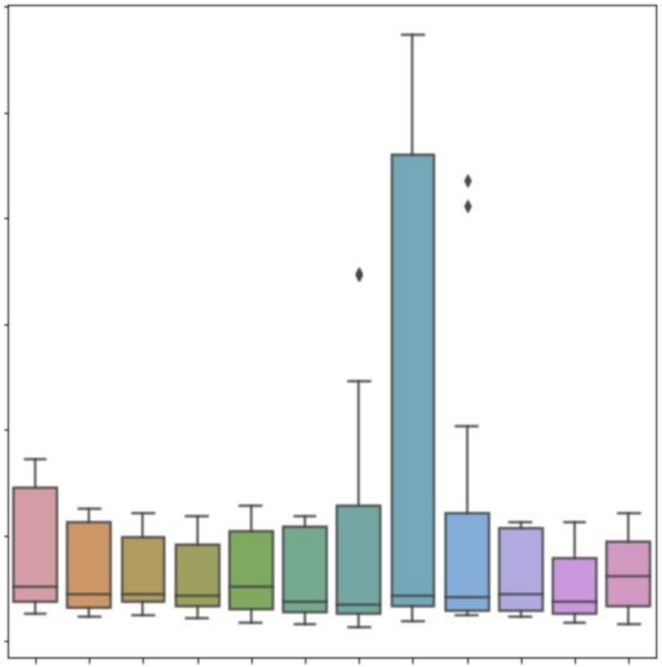
- 3 different model types
- 2 different loss functions
- 5 different feature lists
- Multiple differencing strategies

	project_name	model_type	featurelist_name
0	TS_FD:1-7_FDW:28_Cluster-0	eXtreme Gradient Boosted Trees Regressor with Early Stopping	DR Reduced Features M11
1	TS_FD:1-7_FDW:28_Cluster-1	Light Gradient Boosting on ElasticNet Predictions	DR Reduced Features M15
2	TS_FD:1-7_FDW:28_Cluster-2	eXtreme Gradient Boosted Trees Regressor with Early Stopping	With Differencing (average baseline)
3	TS_FD:1-7_FDW:28_Cluster-3	Zero-Inflated eXtreme Gradient Boosted Trees Regressor with Early Stopping (Poisson Loss)	DR Reduced Features M24
4	TS_FD:1-7_FDW:28_Cluster-4	Zero-Inflated eXtreme Gradient Boosted Trees Regressor with Early Stopping (Poisson Loss)	With Differencing (nonzero average baseline)

Different model types and feature lists performed better for different clusters. This is a major benefit of splitting up your series across multiple projects

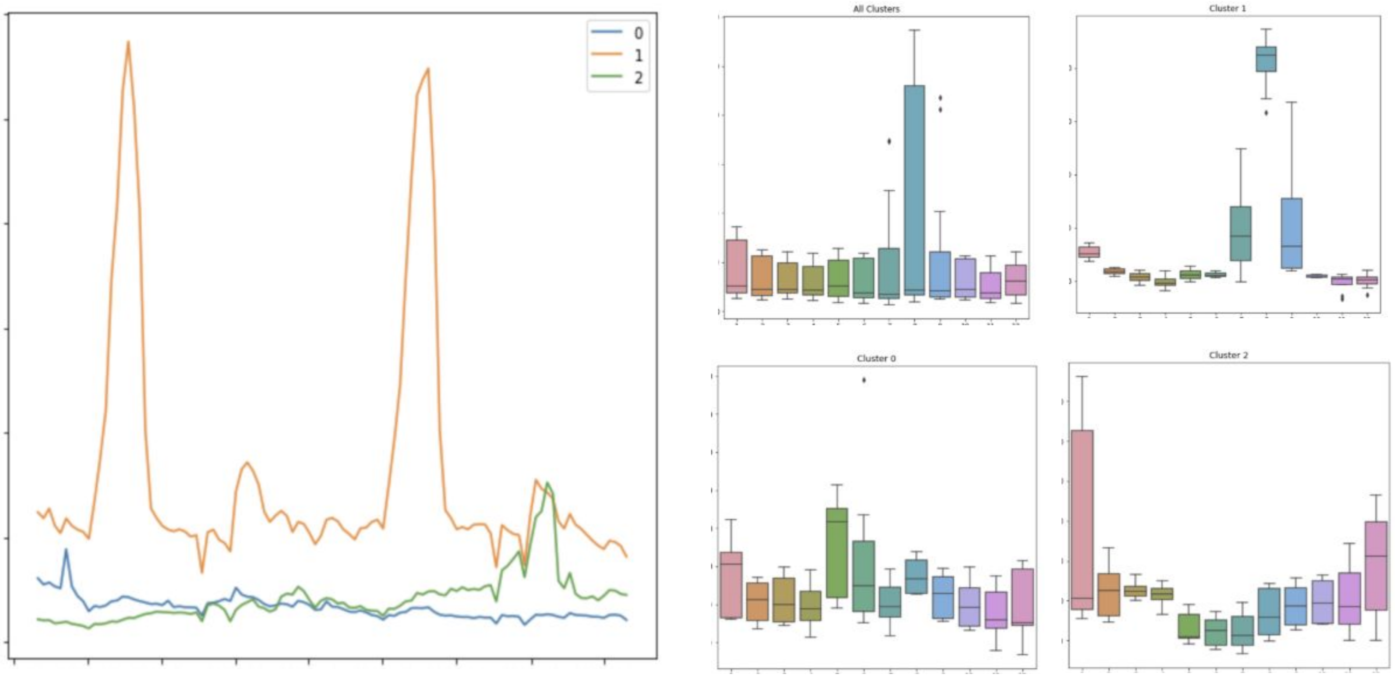
Example Without Series Clustering





Plotting the average target value over time can blur potential clusters in your data

Same Example With Clustering



Certain clusters have distinct time trends. For example, only Cluster 1 (orange) has a sharp sales spike each year. This is a strong indication you may want to build separate projects per cluster

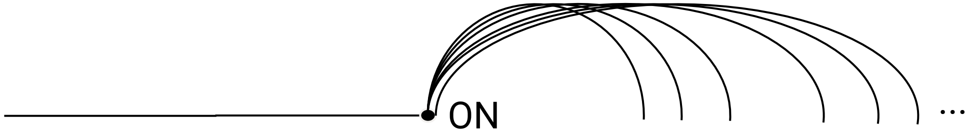
This Begs the Question....How Do We Cluster?

Most Common Techniques:

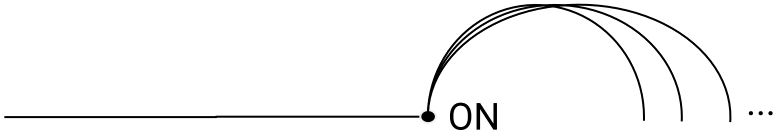
- Domain expertise (e.g. region, department, product type, seasonality, etc.)
- Correlation (Pearson)
- Performance
- Periodicity (e.g. weekly, monthly, or yearly seasonality)
- Level (e.g. average target value)
- Partial autocorrelation (PAC)
- Dynamic time warping (DTW)

Split Projects | Forecast Distance

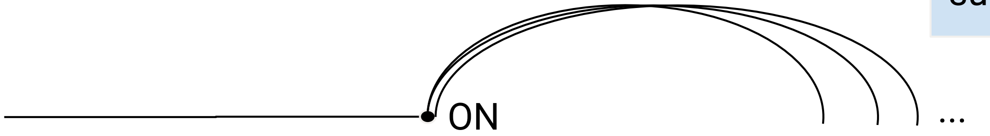
Original Project
FD = [1,30]



Project 1
FD = [1,14]



Project 2
FD = [15,30]



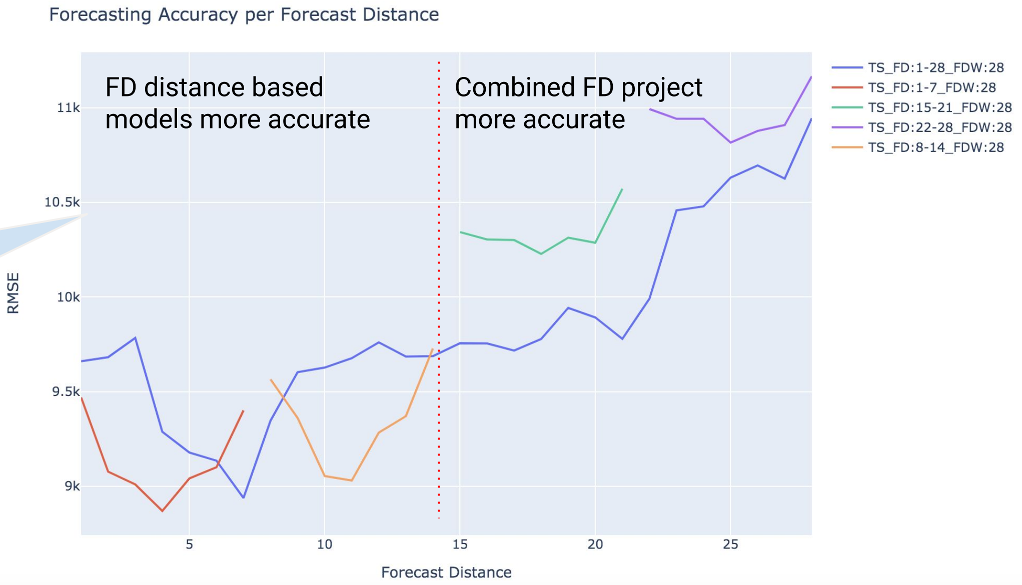
It's possible a tree based model is better at predicting the first 14 days, and a linear model is better at predicting the subsequent 15 days

Different model types may be better at forecasting long or short distances

Split Projects | Forecast Distance

Different models can be more/less accurate across forecast distances:

Building a separate model for the first 14 FDs could improve performance!

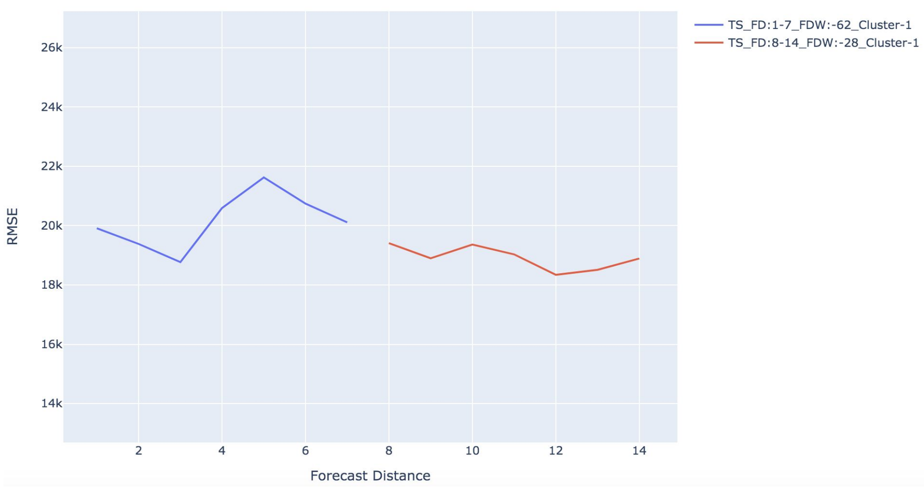


Always recompute and aggregate predictions then compare against a baseline model to measure lift!

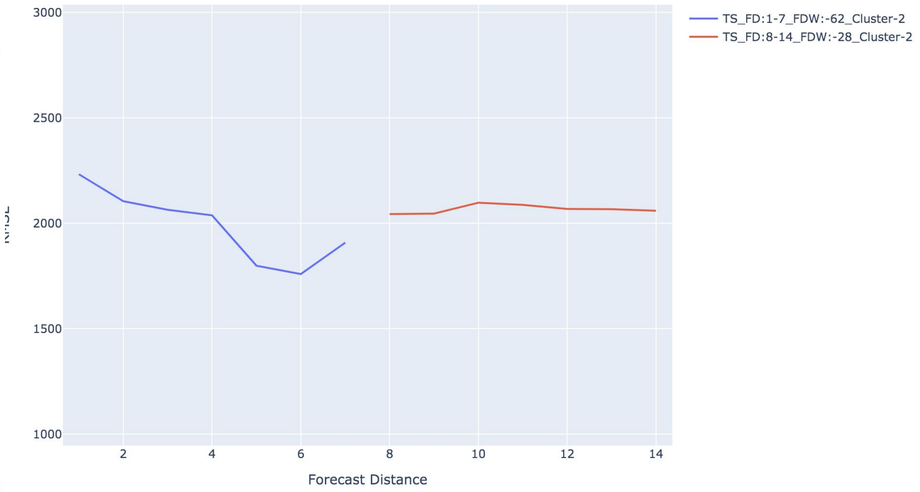
Split Projects | Forecast Distance and Cluster

Why stop here? Take it a step further and split up each forecast distance project by cluster:

Cluster 1

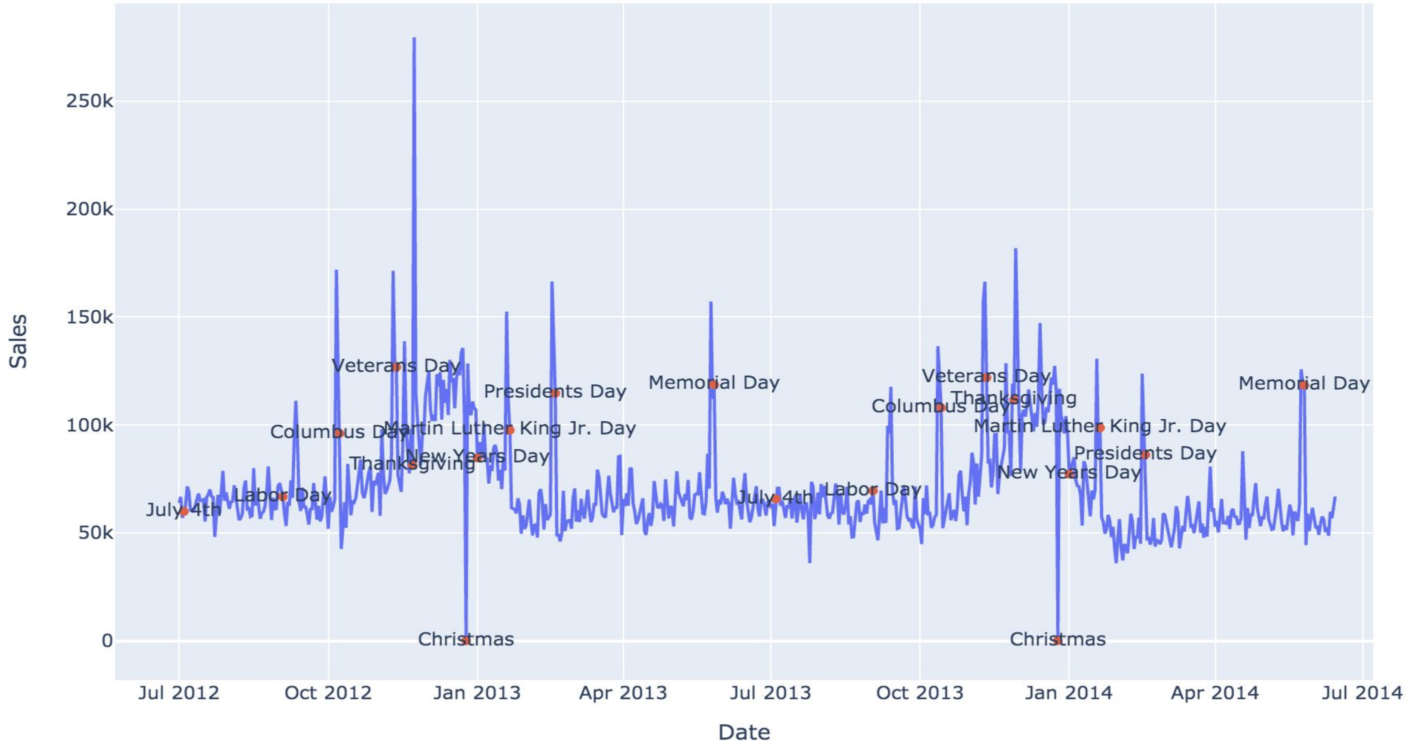


Cluster 2



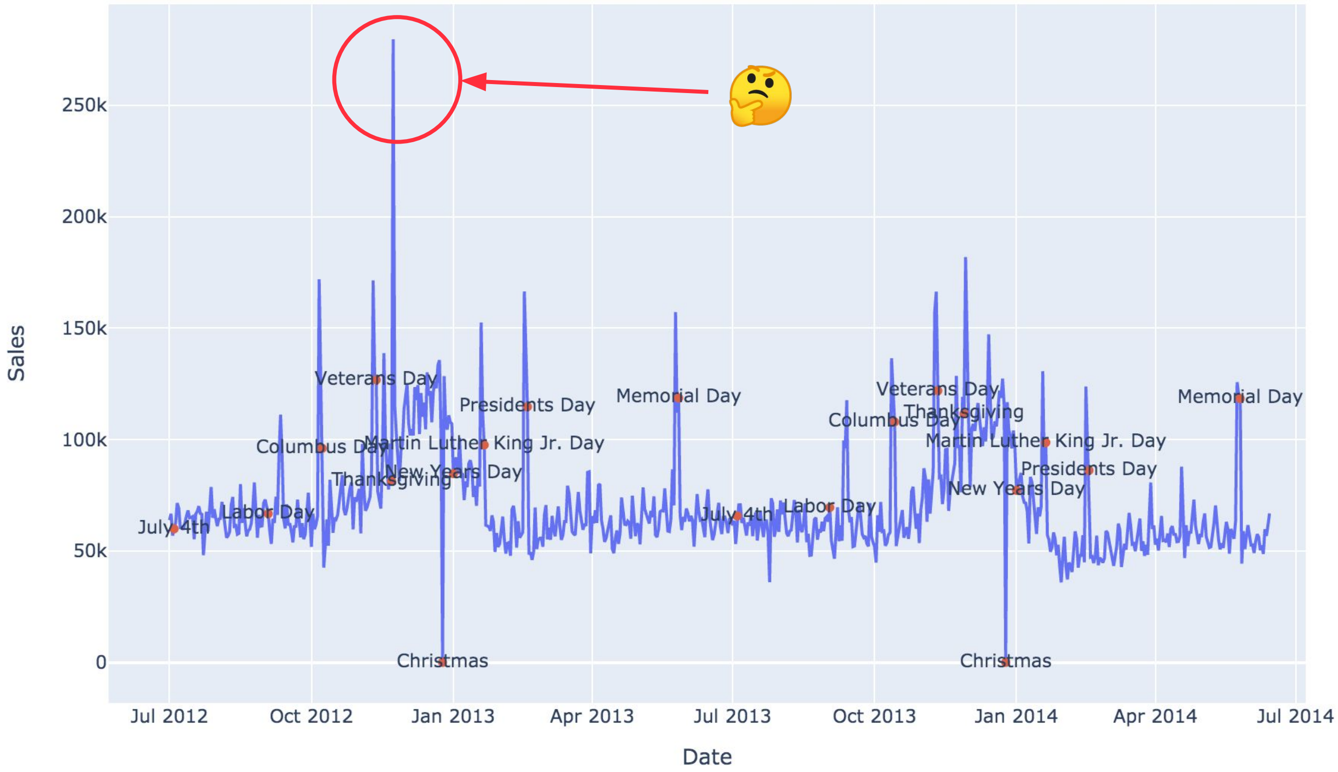
Calendar Events | Adding Important Features

- Plot data with calendar events overlayed
- Check to see if you are missing important events



Check that your calendar captures all major events *(don't cheat and include your test periods)*

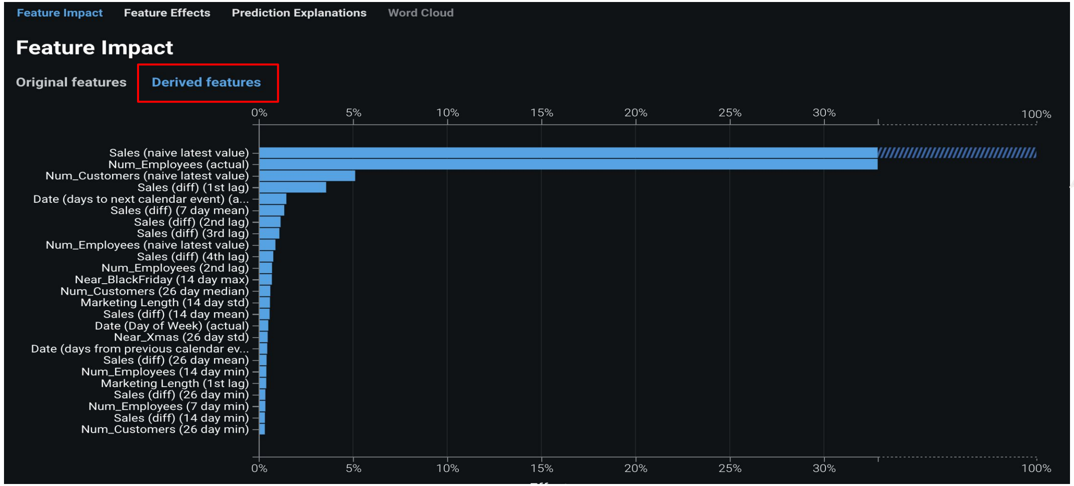
Calendar Events | Detecting Missing Events



We missed Black Friday. Add this to your calendar and re-upload.

Model Improvements | Feature Reduction

- Reducing the feature set can improve forecasts by preventing overfitting
  - Get un-normalized feature impact scores → keep features that account for 99% of impact score total



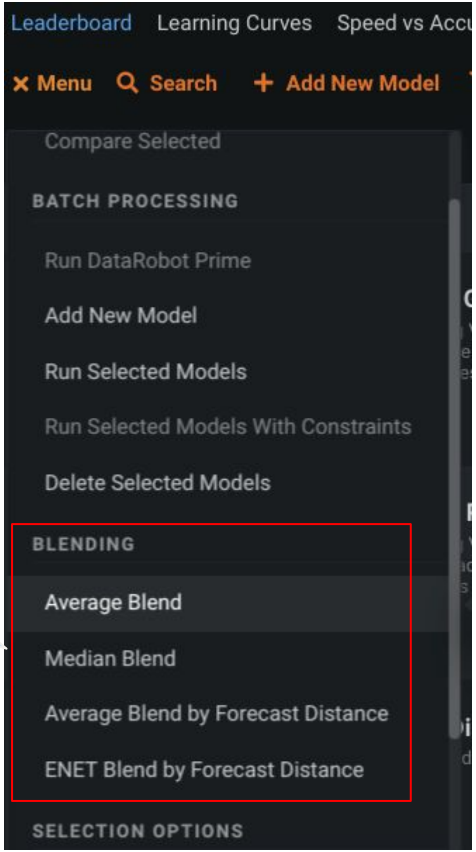
Use the Python or R API to get un-normalized feature importance

Iterate by removing features based on percent of total importance. There is no “correct” answer on how many features to remove. Treat this as a hyperparamter you need to tune.

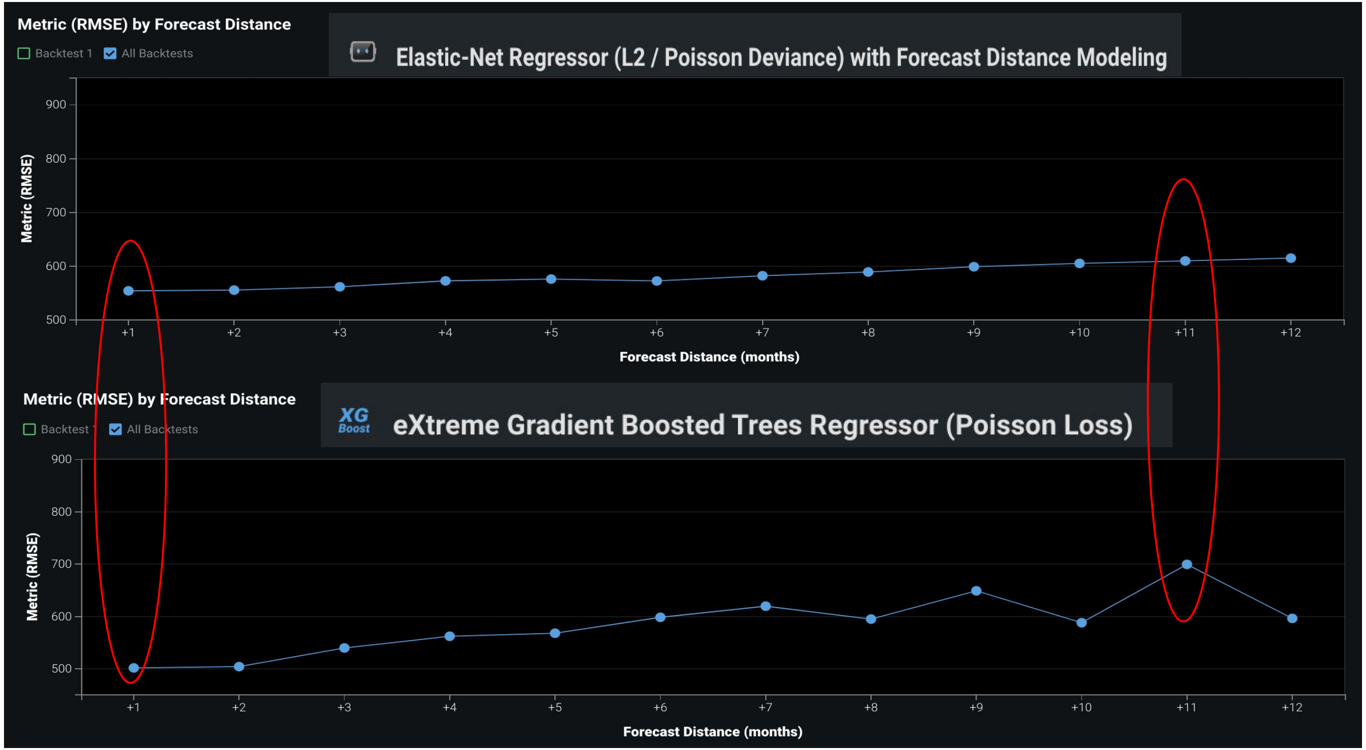
Blend Top Performing Models

- The more diverse the blueprint, the better
- Try blending different types of modeling families
  - ARIMA
  - Tree-based (XGBoost, RF, LGBM)
  - Linear (Ridge, Elastic-Net, Eureqa)

Experiment with different blending options



Blend Models by Forecast Distance



Elastic-Net has lower error on **FD 11**

XGB has lower error on **FDs 1-5**

We can leverage the FD blender to take advantage of differences in FD accuracy across models

Blending models based on forecast distance can provide a large accuracy lift

Your Turn, Use the API or GUI!

- **‘DR_Demo_Stock_Disperion.csv’**
  - Target Column: Dispersion
  - Single Series
  - No known in advance (KIA) features

- **Kick off Time Series Project**
  - Adjust Datetime partitioning
    - 3 backtests, 1 year each
    - 0-28 day feature derivation window
    - 1-3 day forecast distance

