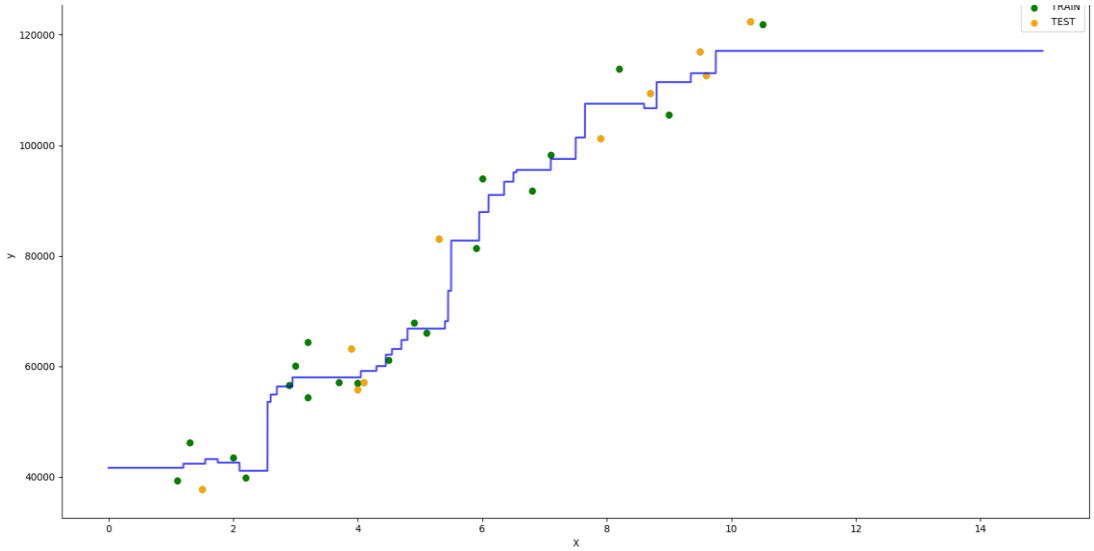
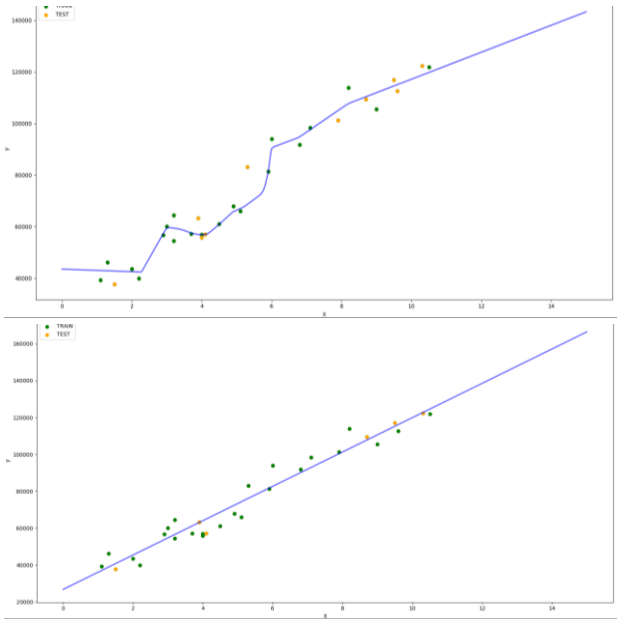
Visualization trick for multivariate regression problems

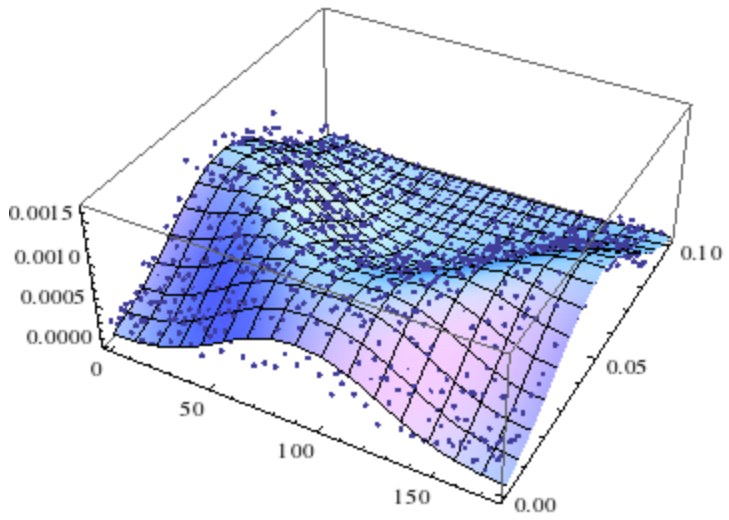
In a univariate regression problem, it is easy to see how your model is generalizing the problem by doing a simple X vs Y plot:

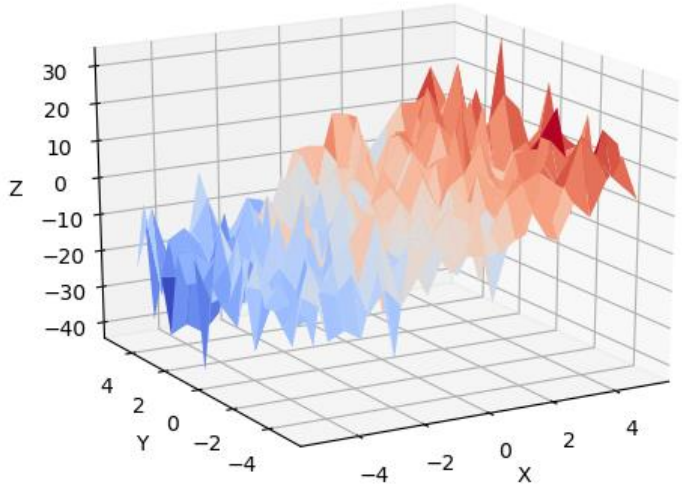


With this plot, we can easily see if the model found a good generalization solution, if it overfit, underfit, etc. It is also a nice tool to compare different models together:



When you add 1 feature to the regression problem, we can still plot the model decision function but it will be in 3D, and the generalization solution is not a line but become a hyperplane:





Source: https://scipy-lectures.org/packages/statistics/auto_examples/plot_regression_3d.html

It is more difficult to evaluate the solution in 3D but still, it is possible !

Now, what happen if we add other features to the regression model, how can we visualize the model regression function in 4D, 5D, 6D… nD ??? Sometimes, it is difficult to evaluate a model performance only by using the RMSE, MAE or R2 scores because it does not give any clues about how the model is reacting when the data changes, so wee need a visualization to help us out!

Off course, this trick is not perfect, nothing is! Still, it helps to understand how the model reacts when the target change and it also help to validate the overall model generalization performance. Something nice about this trick: It is always the same, no matter how many features to your model.

First, when you got your train fit and test predictions ready, create 2 new datasets with 2 columns each:

Dataset_1: **fit_results_df**:

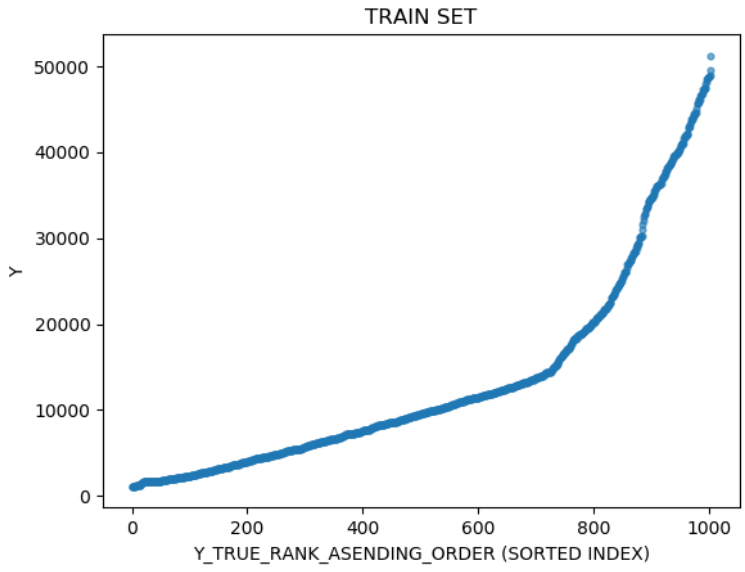
1. Y_TRUE: Y train set original values
2. Y_FIT: Y train fit values

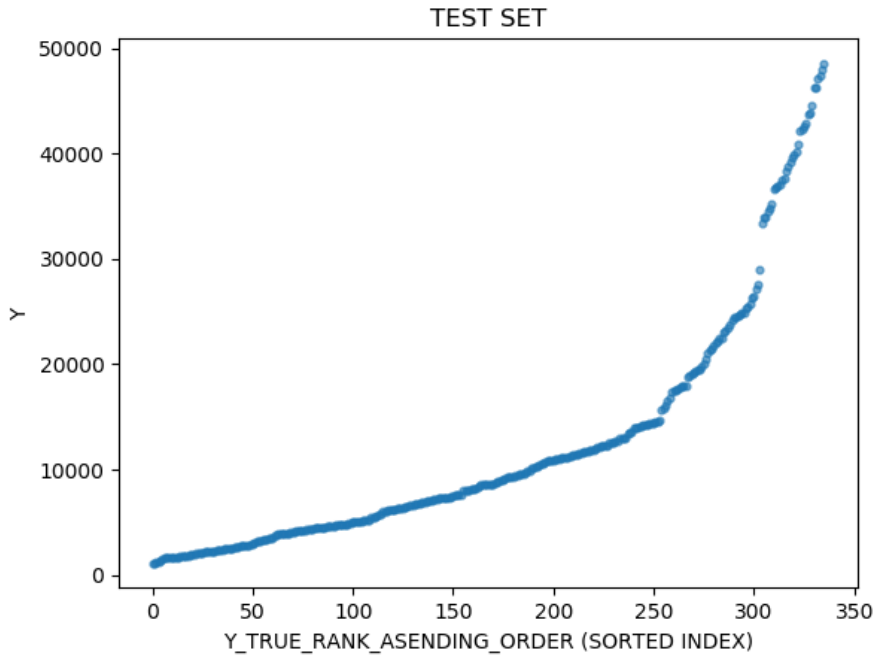
Dataset_2: **test_pred_results_df**:

1. Y_TRUE: Y test set original values
2. Y_PRED: Y test prediction values

Then, for both datasets, simply sort all the rows by Y_TRUE values in an ascending order and finally, reset the indexes.

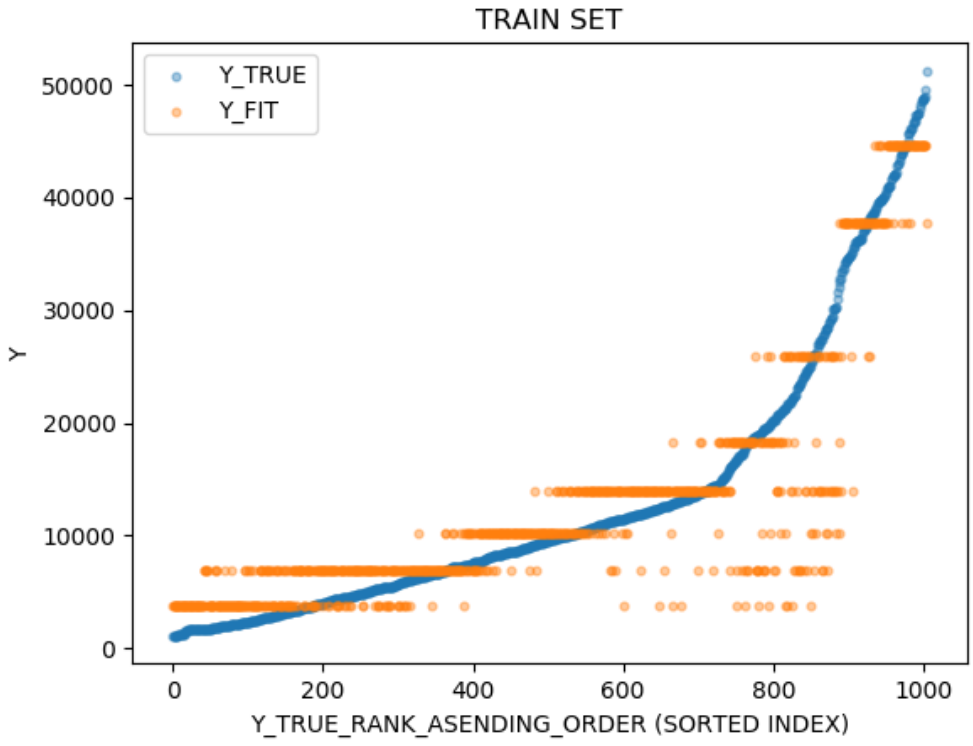
If we plot the Y_TRUE values of both datasets, it will show:

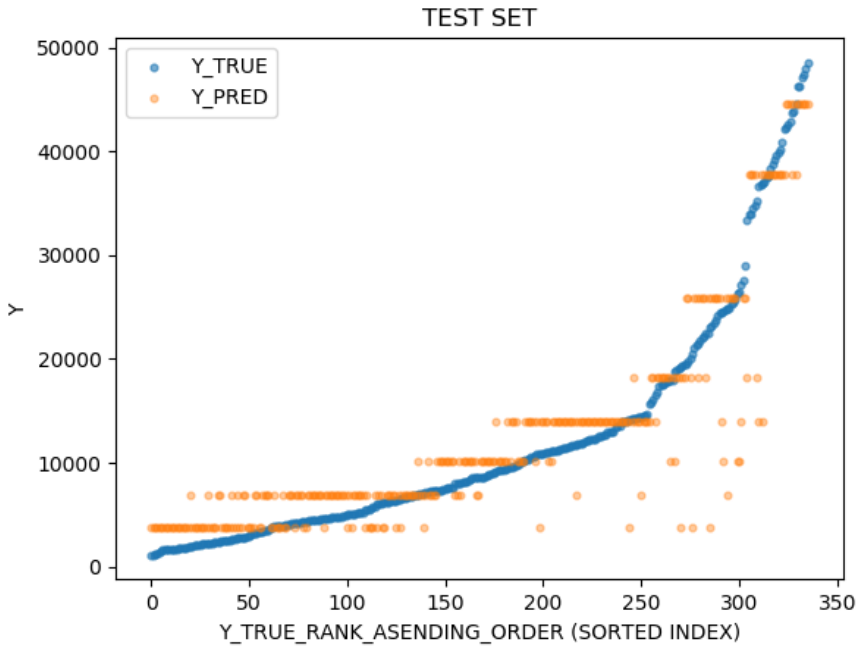




This is all the ordered y values of the train and test set in ascending order.

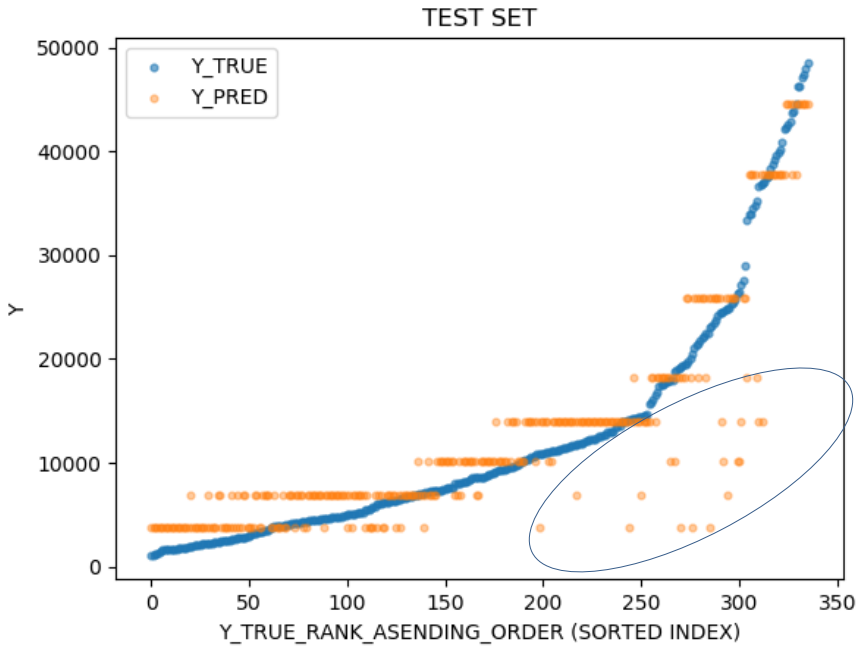
Then, we plot the corresponding y fit and prediction values of both datasets:



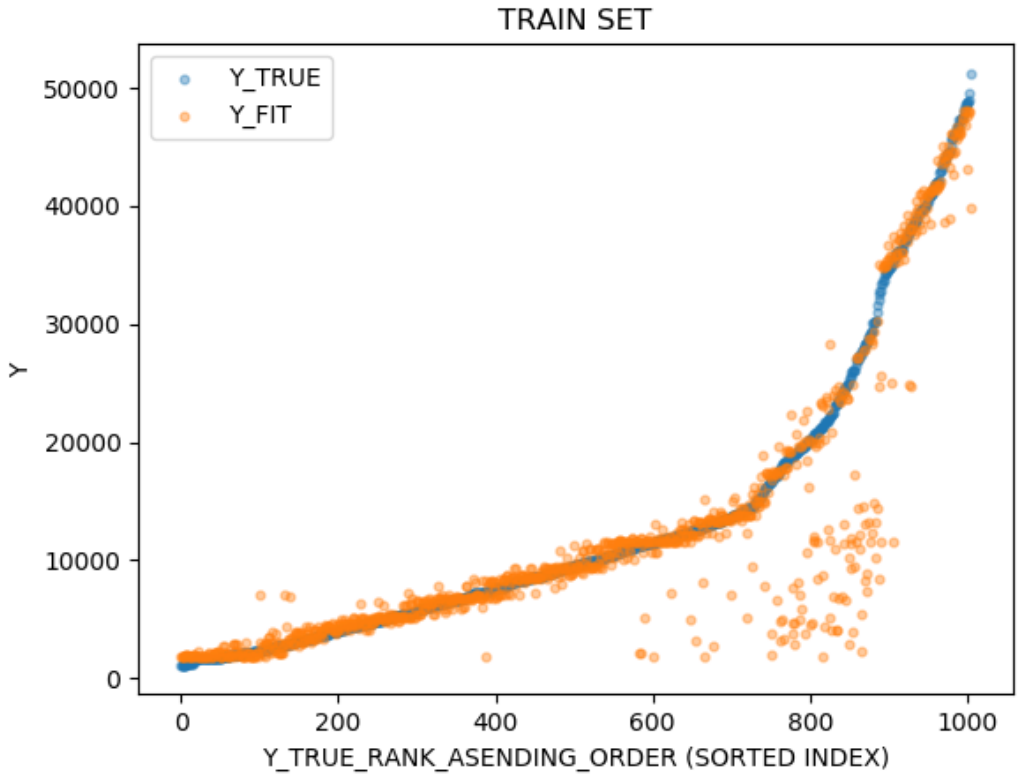


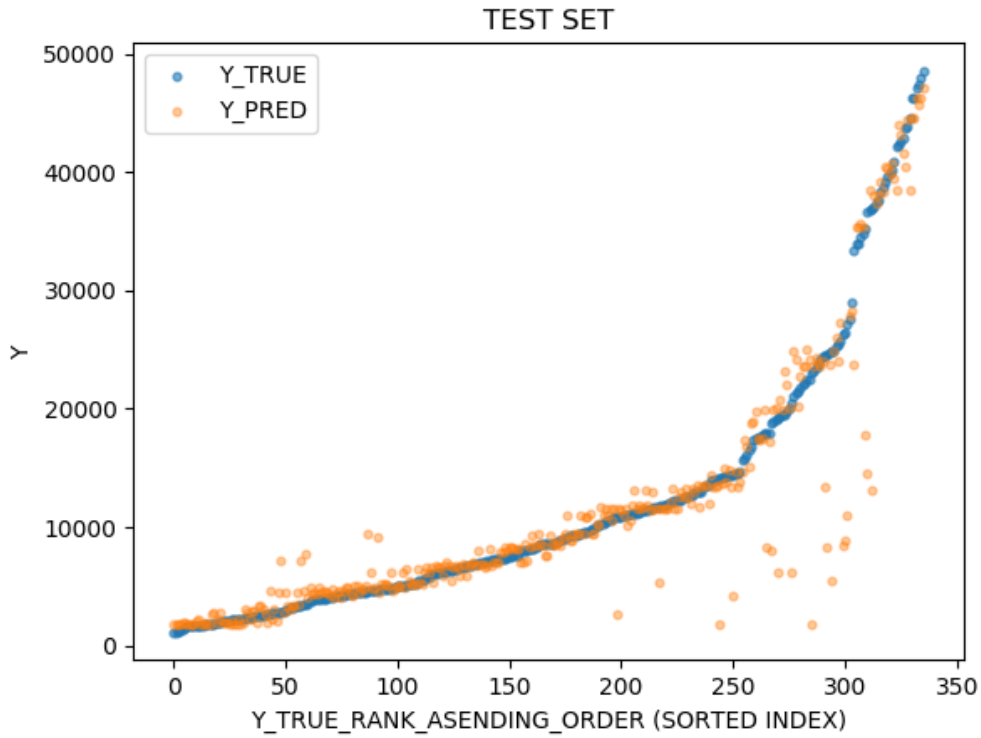
This model is a simple **Decision Tree** with a **max depth** limited to three. We this visualization trick, we can see that:

- The model decision function is “step-alike” (Give a clue that this is a Decision Tree!)
- The model is probably underfitting (Could have more steps!). George St-Pierre would probably say: ***“I am not impressed by your performance”***
- The performance seems very similar between the train fit and the predictions test set
- The model follow the target variations; when the y values goes high, the prediction and fit goes high too most of the time ! (Sometimes, it happens that the model is only good with high or low target values and do not follow the target variations correctly… the visualization help us to see that very easily!)
- There are patterns, bunches of fit/predictions that are out! (Maybe outliers?):



Let us improve the model a bit and see what will happen! :

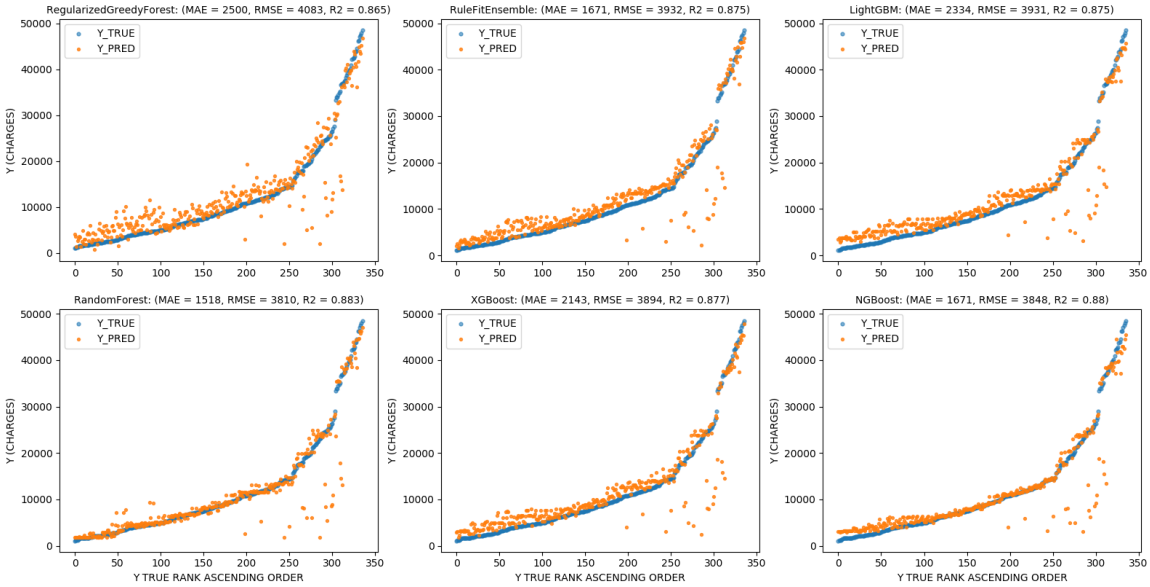




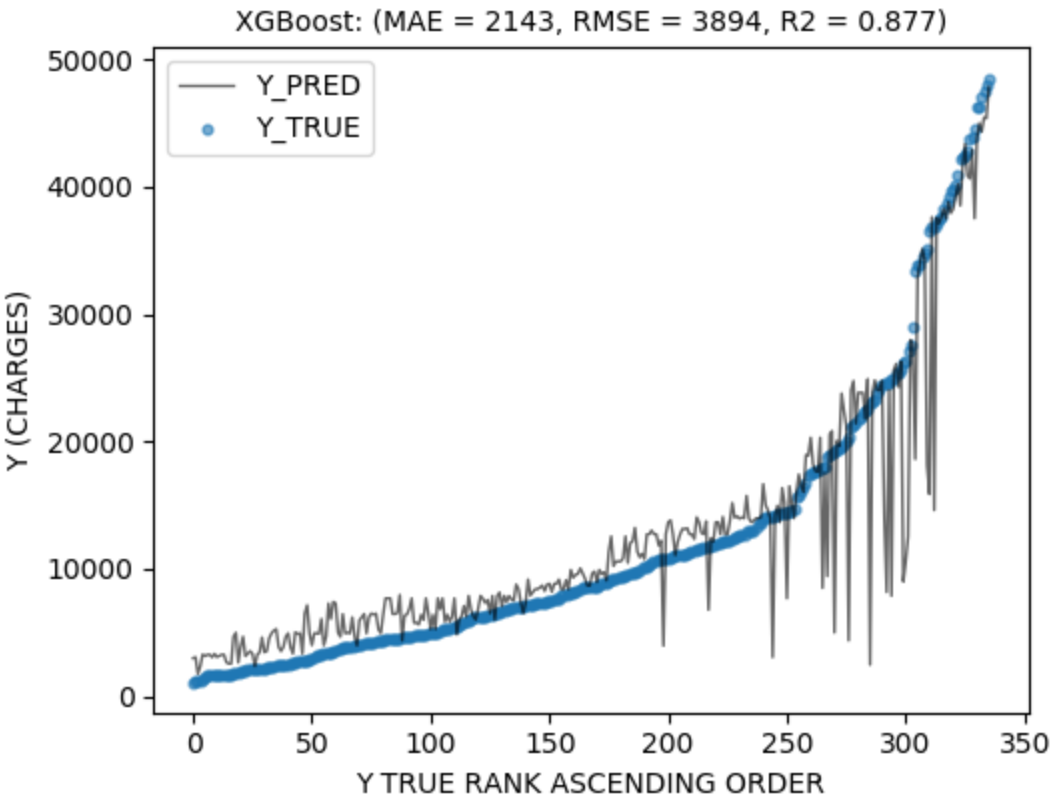
This new model is a **Random Forest** with a **max depth** limited to 5 and **750 estimators**.

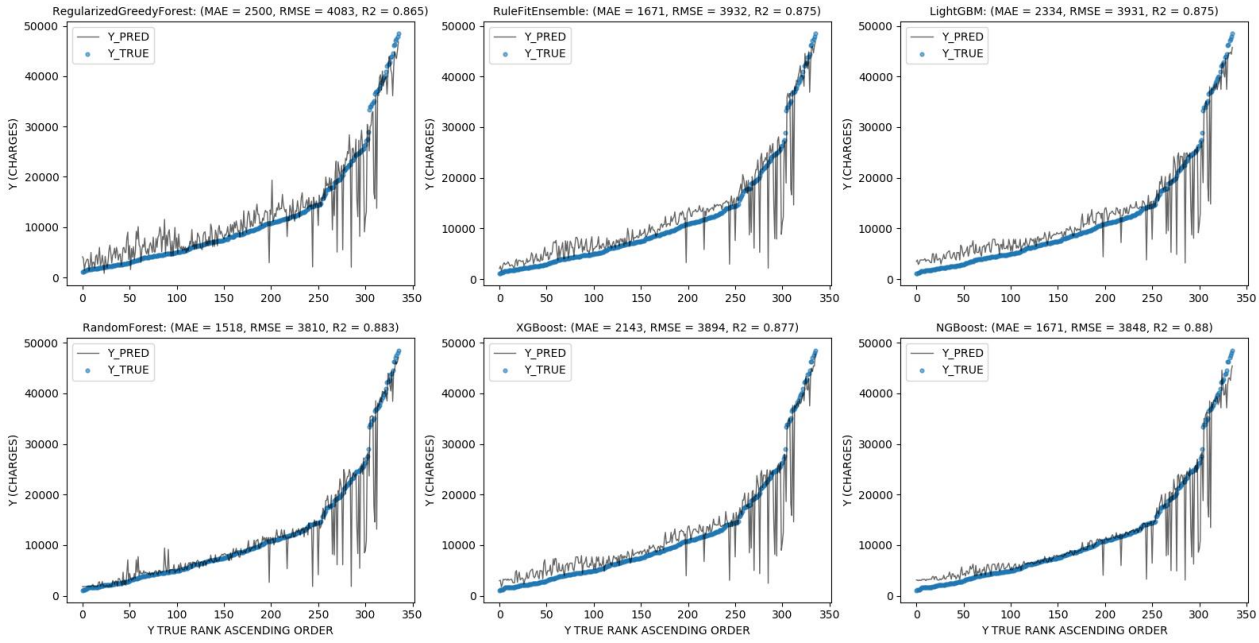
We can easily say that this model is better than the old one, without checking at the Scores! (Although, you should check the scores!). We can also see that there are still bunches of fit/predictions that are out in the right-bottom corner of the plot, those data points need an investigation!

If we want, we can also compare multiple models at a time (Let say we are only interested in the predictions of the test set):

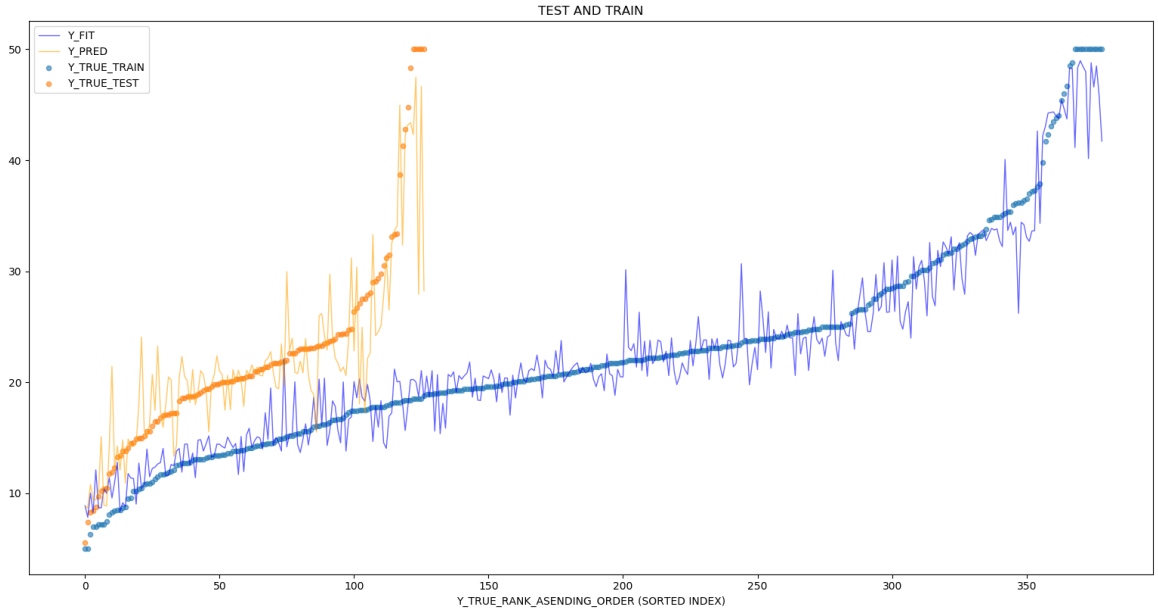


If you prefer, you can also use a line instead of data points to represent the "simplified" model decision function on Y Target:



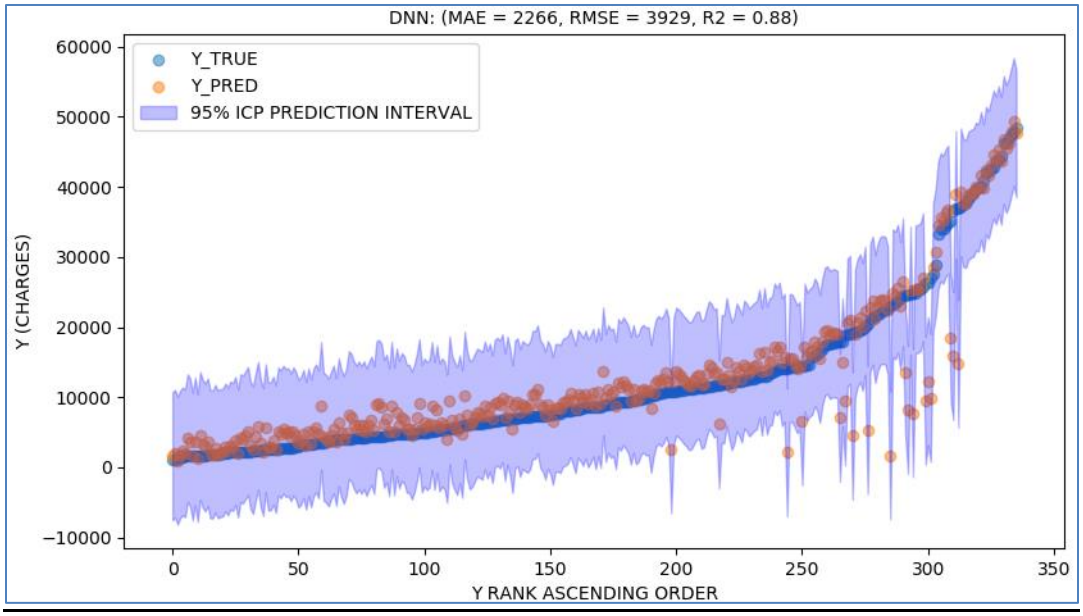


You can plot the train and the test sets at the same time if you want!



Example: Random Forest on boston dataset, test and train datasets

You can also add a prediction interval, like the inductive conformal prediction, around Y prediction target if needed:



Example: Deep Neural Network with an Inductive Conformal Prediction

Conclusion: We saw that this tool:

- Is not perfect but can help to validate and visualize a "simplified" multivariate regression model decision function when you have more than two features in your problem
- Can be used to compare different models together
- Can help to identify patterns, for example: If the model follows correctly the target variations or if there are data points clusters that did not fit/predict correctly / need investigations (Maybe some feature engineering / cleaning steps to help the model with those data points!)
- Can be use to compare Train vs Test performance
- Can be customized depending on the preferences or needs of the data sicence practitioner (Scatter vs line plot, inductive conformal prediction, train vs test, etc.)
- Off course, this tool is not sufficient; you should continue to use regression scores like RMSE to validate your models!

I hope that this shared visualization trick could be useful to you!

The simplified codes of this experimentation is available on my github (Experimentation on the boston dataset): [github]

Dave Cote, Data Scientist, M.Sc. BI