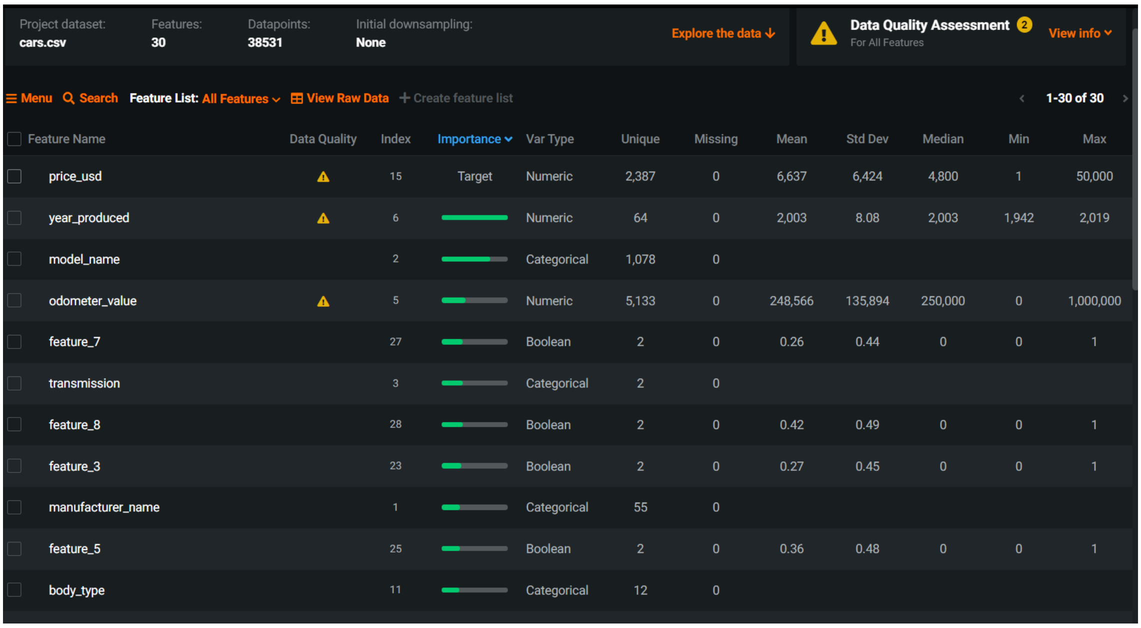
M7 Practical Challenge: Predictive Modeling with DataRobot

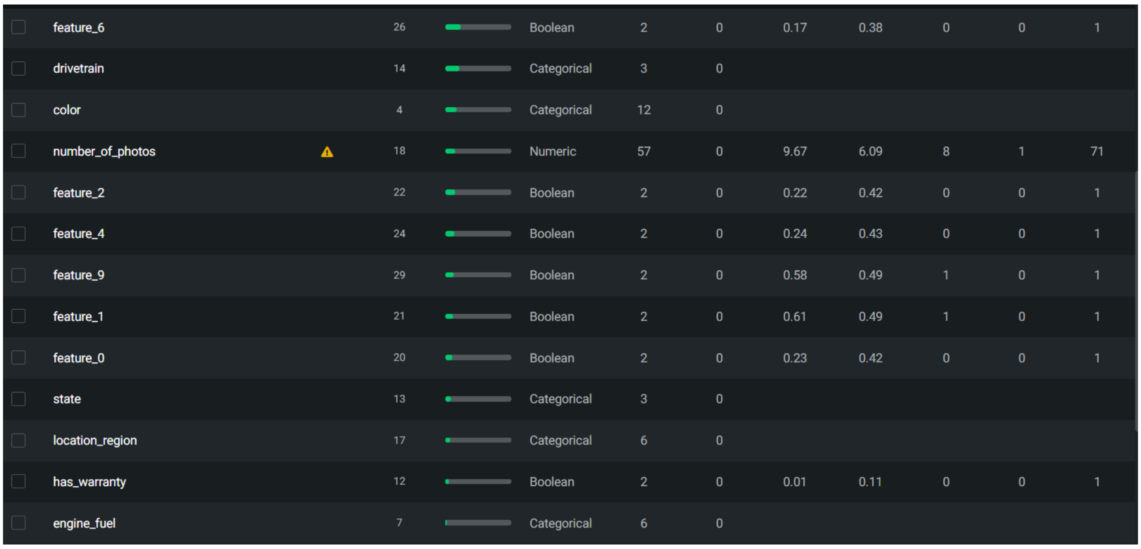
Table of Contents

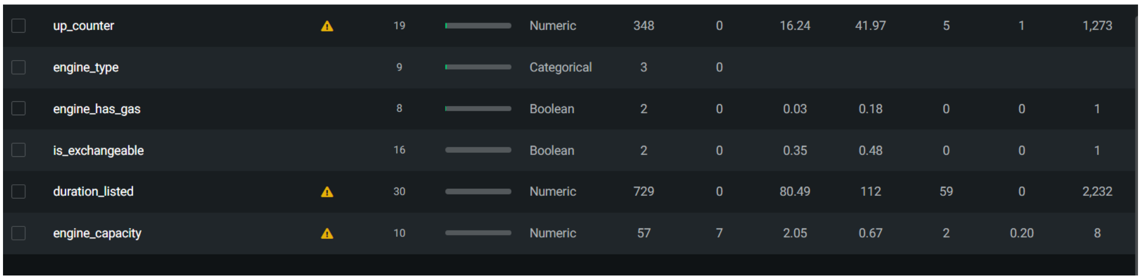
Input Data

The data is the same used in the previous task: Cars.csv Containing information about cars and its prices.

Once imported to Data Robot the data is presented as follow:

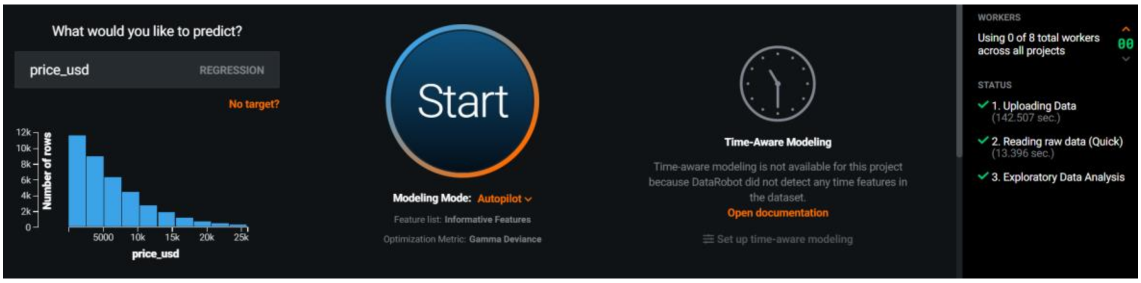






□ Feature Name	Data Quality	Index	Importance	Var Type	Unique	Missing	Mean	Std Dev	Median	Min	Max
□ price_usd	⚠	15	Target	Numeric	2,387	0	6,637	6,424	4,800	1	50,000
□ year_produced	⚠	6		Numeric	64	0	2,003	8.08	2,003	1,942	2,019
□ model_name		2		Categorical	1,078	0					
□ odometer_value	⚠	5		Numeric	5,133	0	248,566	135,894	250,000	0	1,000,000
□ feature_7		27		Boolean	2	0	0.26	0.44	0	0	1
□ transmission		3		Categorical	2	0					
□ feature_8		28		Boolean	2	0	0.42	0.49	0	0	1
□ feature_3		23		Boolean	2	0	0.27	0.45	0	0	1
□ manufacturer_name		1		Categorical	55	0					
□ feature_5		25		Boolean	2	0	0.36	0.48	0	0	1
□ body_type		11		Categorical	12	0					
□ feature_6		26		Boolean	2	0	0.17	0.38	0	0	1
□ drivetrain		14		Categorical	3	0					
□ color		4		Categorical	12	0					
□ number_of_photos	⚠	18		Numeric	57	0	9.67	6.09	8	1	71
□ feature_2		22		Boolean	2	0	0.22	0.42	0	0	1
□ feature_4		24		Boolean	2	0	0.24	0.43	0	0	1
□ feature_9		29		Boolean	2	0	0.58	0.49	1	0	1
□ feature_1		21		Boolean	2	0	0.61	0.49	1	0	1
□ feature_0		20		Boolean	2	0	0.23	0.42	0	0	1
□ state		13		Categorical	3	0					
□ location_region		17		Categorical	6	0					
□ has_warranty		12		Boolean	2	0	0.01	0.11	0	0	1
□ engine_fuel		7		Categorical	6	0					
□ up_counter	⚠	19		Numeric	348	0	16.24	41.97	5	1	1,273
□ engine_type		9		Categorical	3	0					
□ engine_has_gas		8		Boolean	2	0	0.03	0.18	0	0	1
□ is_exchangeable		16		Boolean	2	0	0.35	0.48	0	0	1
□ duration_listed	⚠	30		Numeric	729	0	80.49	112	59	0	2,232
□ engine_capacity	⚠	10		Numeric	57	7	2.05	0.67	2	0.20	8

Then the Target Variable is selected as price_usd:



What would you like to predict?

price_usd REGRESSION

Modeling Mode: Autopilot

Feature list: Informative Features

Optimization Metric: Gamma Deviance

Time-Aware Modeling

Time-aware modeling is not available for this project because DataRobot did not detect any time features in the dataset.

WORKERS

Using 0 of 8 total workers across all projects

STATUS

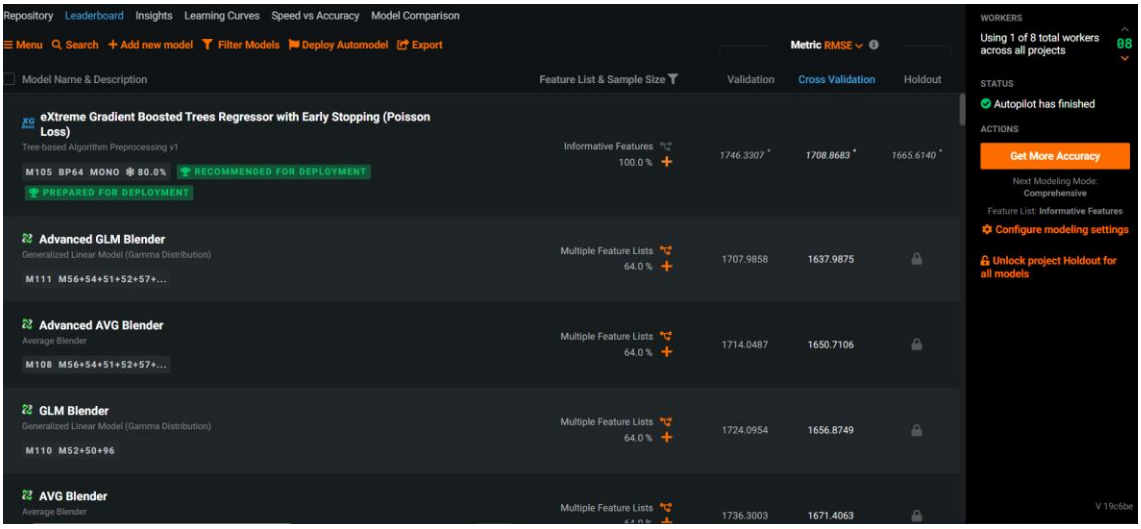
✓ 1. Uploading Data (142.507 sec.)

✓ 2. Reading raw data (Quick) (13.396 sec.)

✓ 3. Exploratory Data Analysis

Model Selection

As we can see the best model is eXtreme Gradient Boosted Trees Regressor with Early Stopping (Poisson Loss), with an RMSE of 1746.



Model Name & Description	Feature List & Sample Size	Validation	Cross Validation	Holdout
eXtreme Gradient Boosted Trees Regressor with Early Stopping (Poisson Loss) Tree-based Algorithm Preprocessing v1 M105 BP64 MONO ❄ 80.0% RECOMMENDED FOR DEPLOYMENT PREPARED FOR DEPLOYMENT	Informative Features 100.0 %	1746.3307 *	1708.8683 *	1665.6140 *
Advanced GLM Blender Generalized Linear Model (Gamma Distribution) M111 M56+54+51+52+57+...	Multiple Feature Lists 64.0 %	1707.9858	1637.9875	🔒
Advanced AVG Blender Average Blender M108 M56+54+51+52+57+...	Multiple Feature Lists 64.0 %	1714.0487	1650.7106	🔒
GLM Blender Generalized Linear Model (Gamma Distribution) M110 M52+50+96	Multiple Feature Lists 64.0 %	1724.0954	1656.8749	🔒
AVG Blender Average Blender	Multiple Feature Lists	1736.3003	1671.4063	🔒

WORKERS

Using 1 of 8 total workers across all projects 08

STATUS

Autopilot has finished

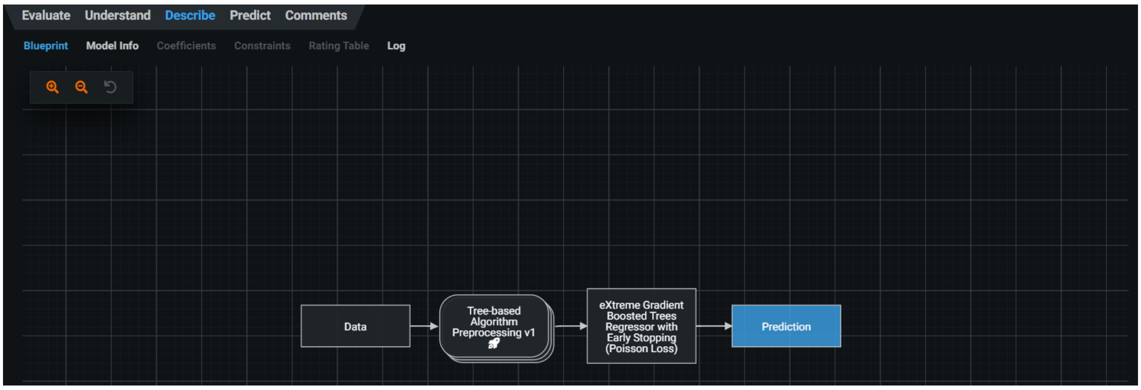
ACTIONS

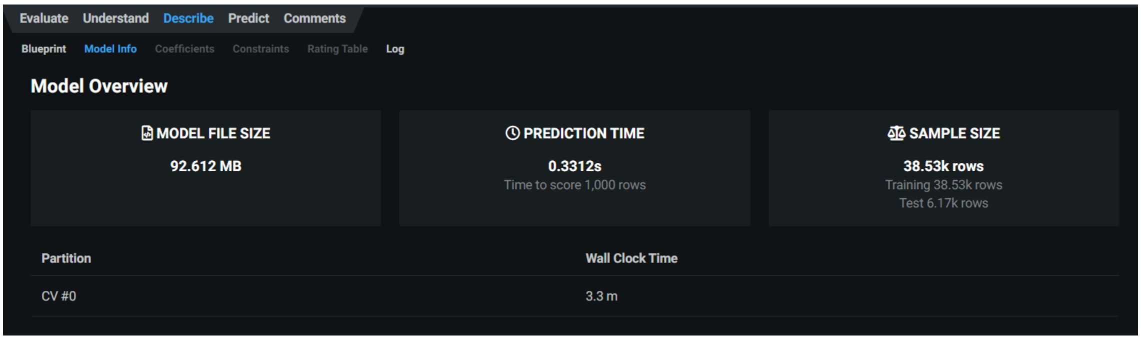
Next Modeling Mode: Comprehensive

Feature List: Informative Features

eXtreme Gradient Boosted Trees Regressor with Early Stopping (Poisson Loss)

Describe Section


Evaluate
Understand
Describe
Predict
Comments
Blueprint
Model Info
Coefficients
Constraints
Rating Table
Log
Data
Tree-based Algorithm Preprocessing v1
eXtreme Gradient Boosted Trees Regressor with Early Stopping (Poisson Loss)
Prediction



Model Overview

MODEL FILE SIZE
92.612 MB

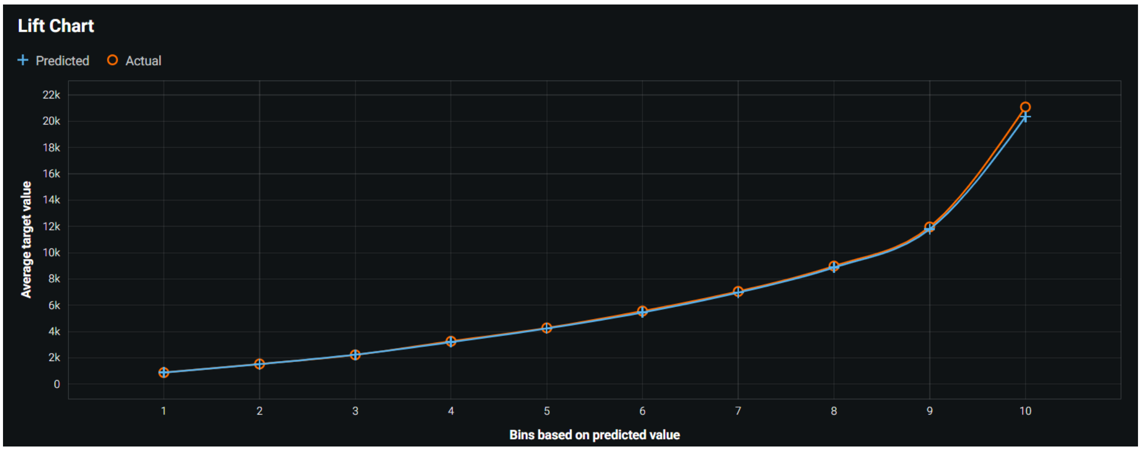
PREDICTION TIME
0.3312s
Time to score 1,000 rows

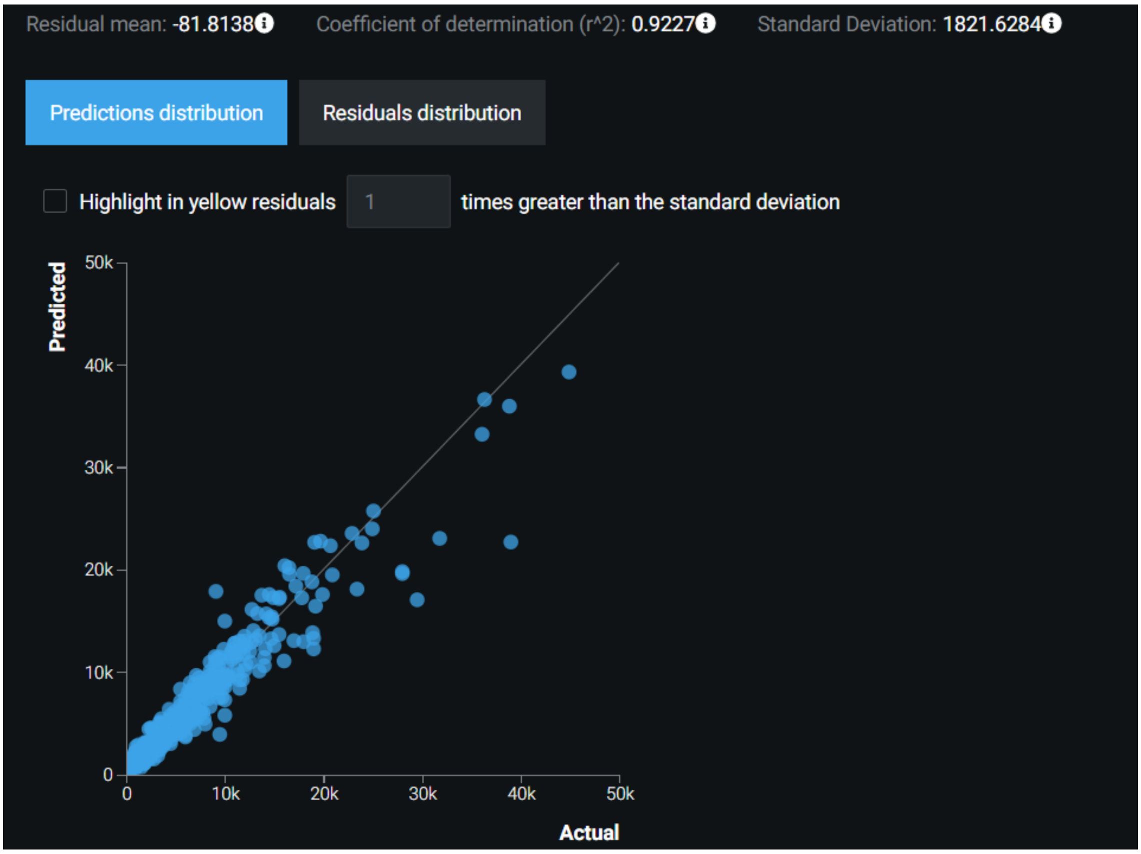
SAMPLE SIZE
38.53k rows
Training 38.53k rows
Test 6.17k rows

Partition	Wall Clock Time
CV #0	3.3 m

As we can see, the model is able to run 1000 rows of data in 0.33 seconds. Meaning it can run a 1 million data sample in less than 6 minutes. Making it a really fast model.

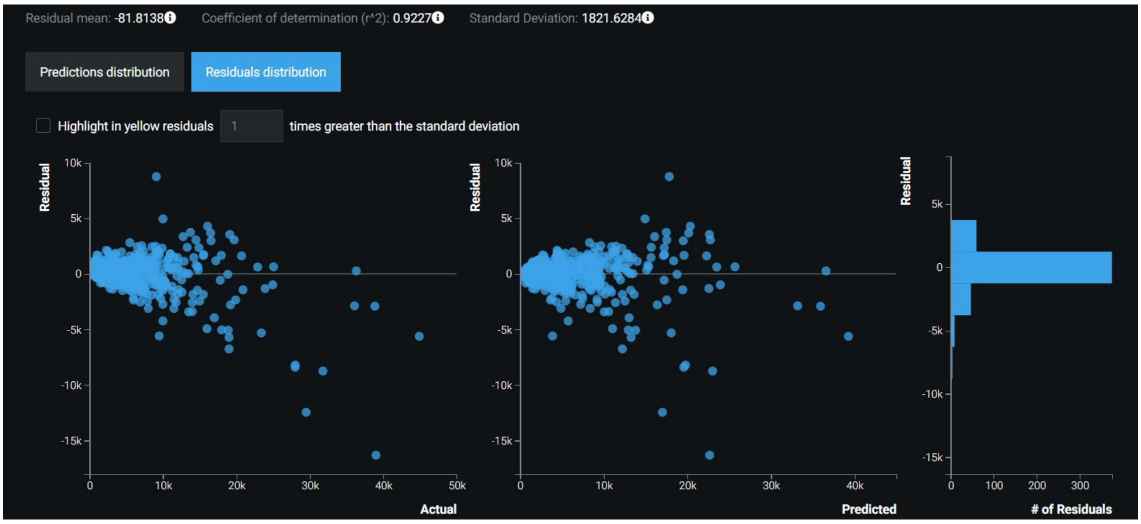
Evaluate Section





Residual mean: -81.8138 Coefficient of determination (r^2): 0.9227 Standard Deviation: 1821.6284

☐ Highlight in yellow residuals 1 times greater than the standard deviation



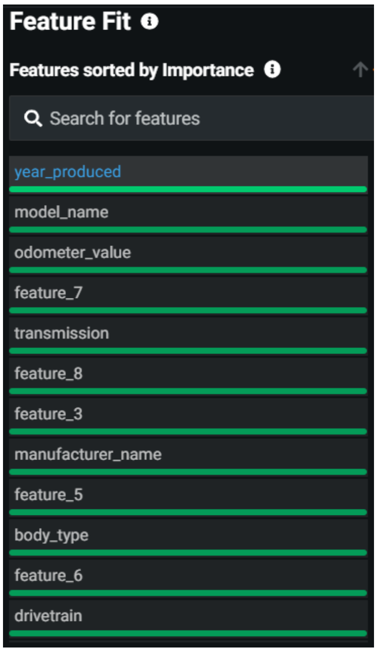
Residual mean: -81.8138 Coefficient of determination (r^2): 0.9227 Standard Deviation: 1821.6284

□ Highlight in yellow residuals 1 times greater than the standard deviation

As we can see with the graphs is that the model is more precise with the lower prices' cars. But this can be because there are less datapoints for the higher prices' cars.

Feature Fit

We can see the importance of each variable on the target variable:

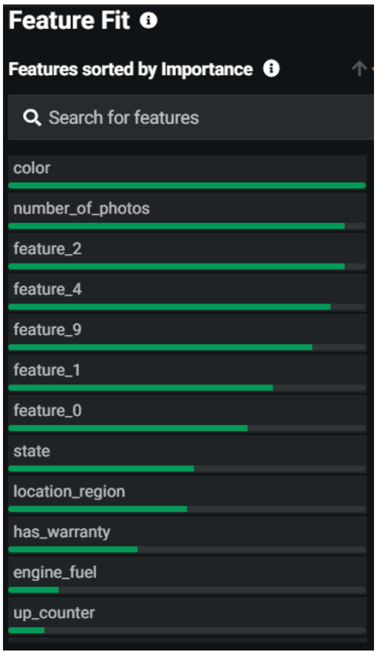


Feature Fit

Features sorted by Importance

Search for features

year_produced
model_name
odometer_value
feature_7
transmission
feature_8
feature_3
manufacturer_name
feature_5
body_type
feature_6
drivetrain

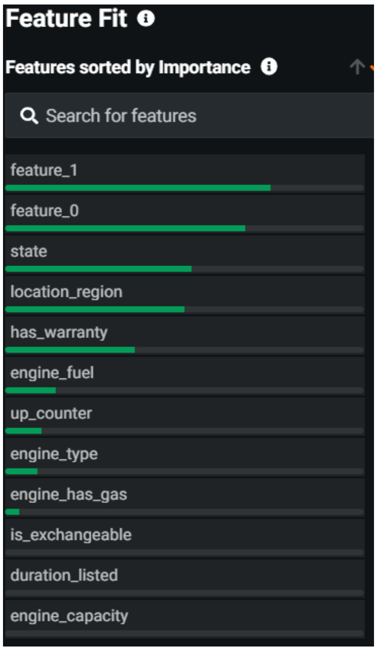


Feature Fit

Features sorted by Importance

Search for features

color
number_of_photos
feature_2
feature_4
feature_9
feature_1
feature_0
state
location_region
has_warranty
engine_fuel
up_counter



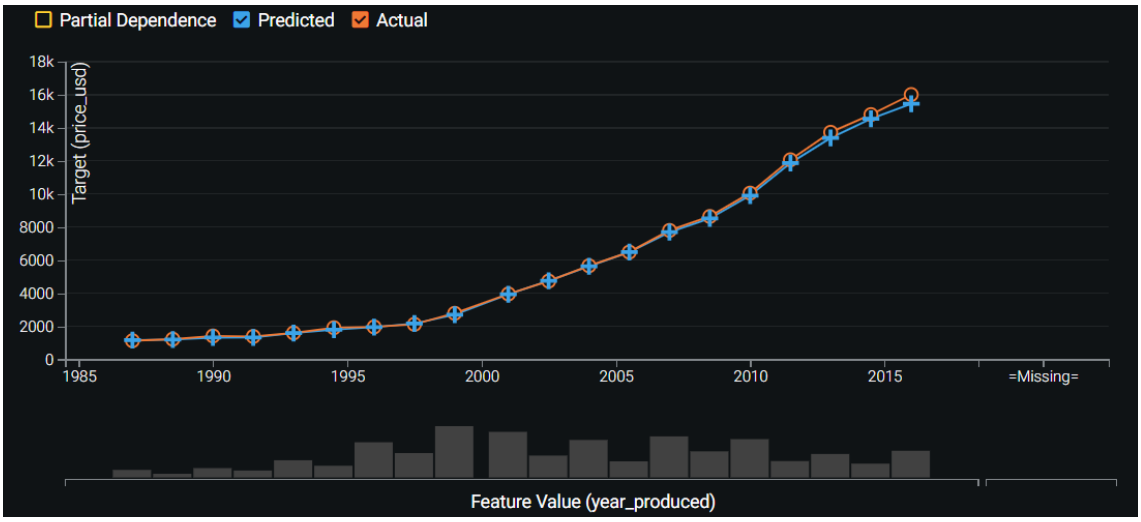
Feature Fit

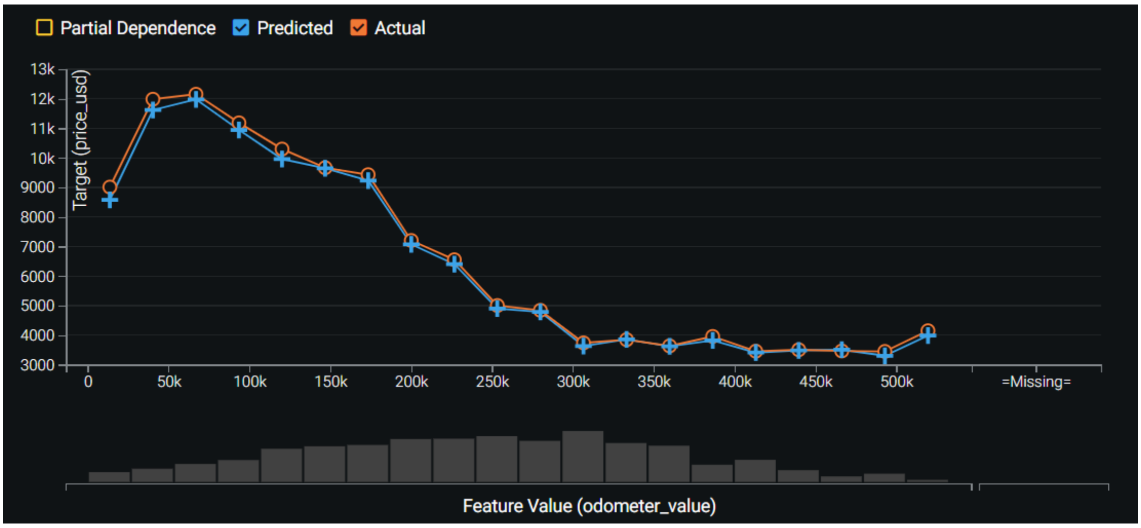
Features sorted by Importance

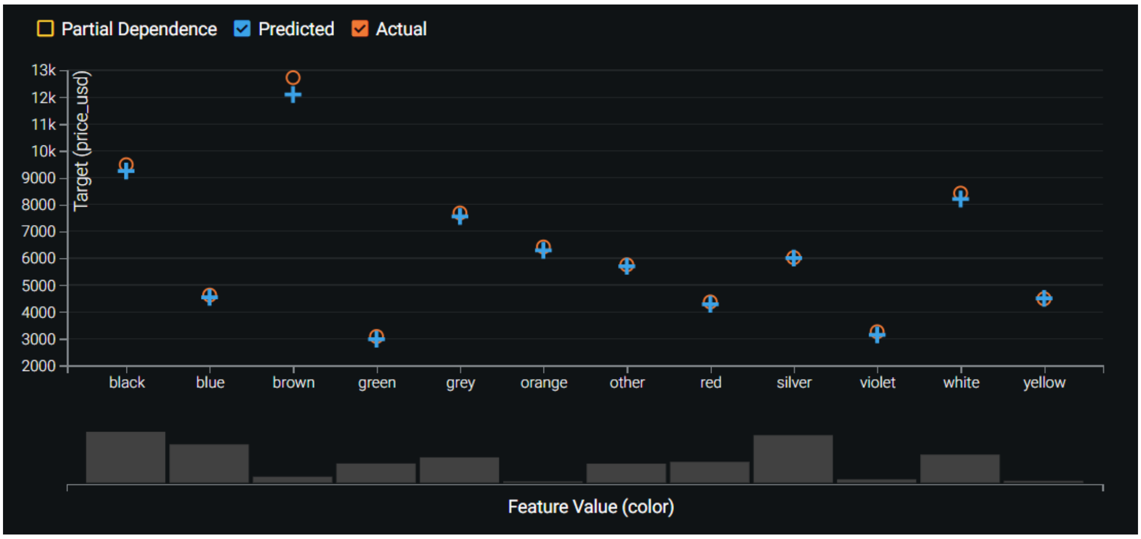
Search for features

feature_1
feature_0
state
location_region
has_warranty
engine_fuel
up_counter
engine_type
engine_has_gas
is_exchangeable
duration_listed
engine_capacity

We can see that there are some variables that are extremely important, such as: year_produced, model_name, odometer_value, feature_7, transmission, feature_8, feature_3, manufacturer_name, feature_5, body_type, feature_6, drivetrain, color. Then the importance of each variable diminishes, until the last variables that have no impact on the model. We can see a couple of graphs: (year_produced, odometer_value, color)



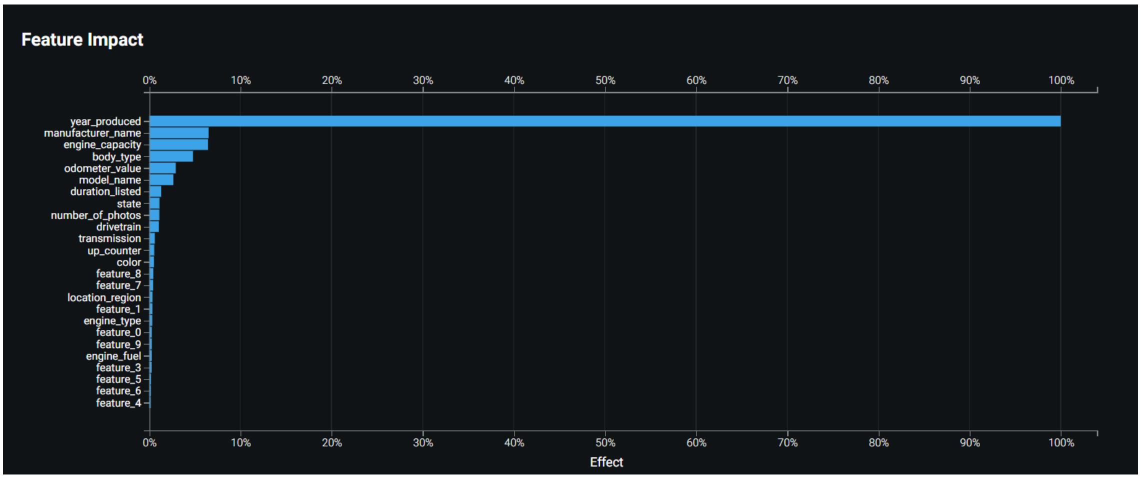




We can see, that the older the car is, the lower its price is, we can also see that the number of kilometres travel by a car have an invers relation to the price, the higher the number, the lower the price. The color also have an impact on the value, as shown on the graph, different color having a different impact on the price.

Understand section

Feature Impact



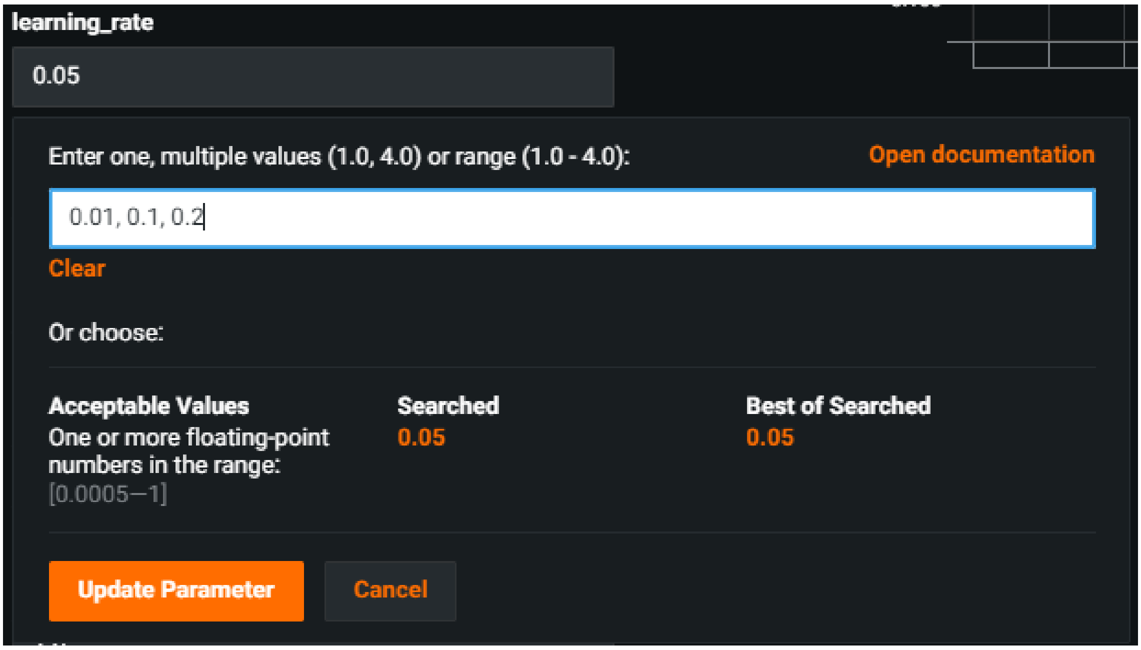
We can also check the importance of each feature to the model. We can see that the year have the highest importance, similar to what we mention on the previous sections.

Hypertuning

So, now we are going to change some parameters to try to improve the performance:

Learning rate

We are trying: 0.01, 0.1, 0.2



learning_rate

0.05

Enter one, multiple values (1.0, 4.0) or range (1.0 - 4.0):

0.01, 0.1, 0.2

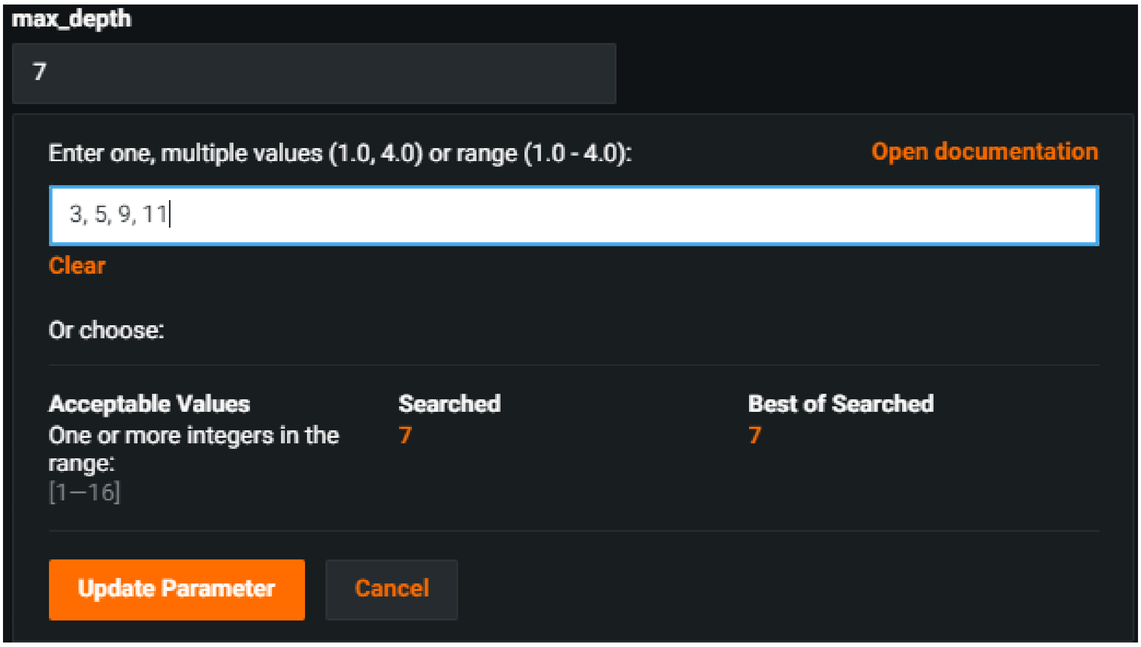
Acceptable Values
One or more floating-point numbers in the range:
[0.0005—1]

Searched
0.05

Best of Searched
0.05

Max depth

We are trying: 3, 5, 9, 11



max_depth

7

Enter one, multiple values (1.0, 4.0) or range (1.0 - 4.0):

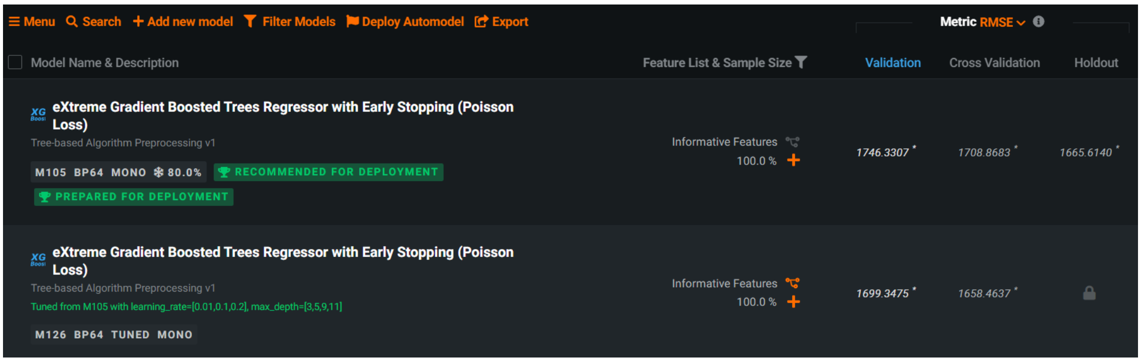
3, 5, 9, 11

Acceptable Values
One or more integers in the range:
[1—16]

Searched
7

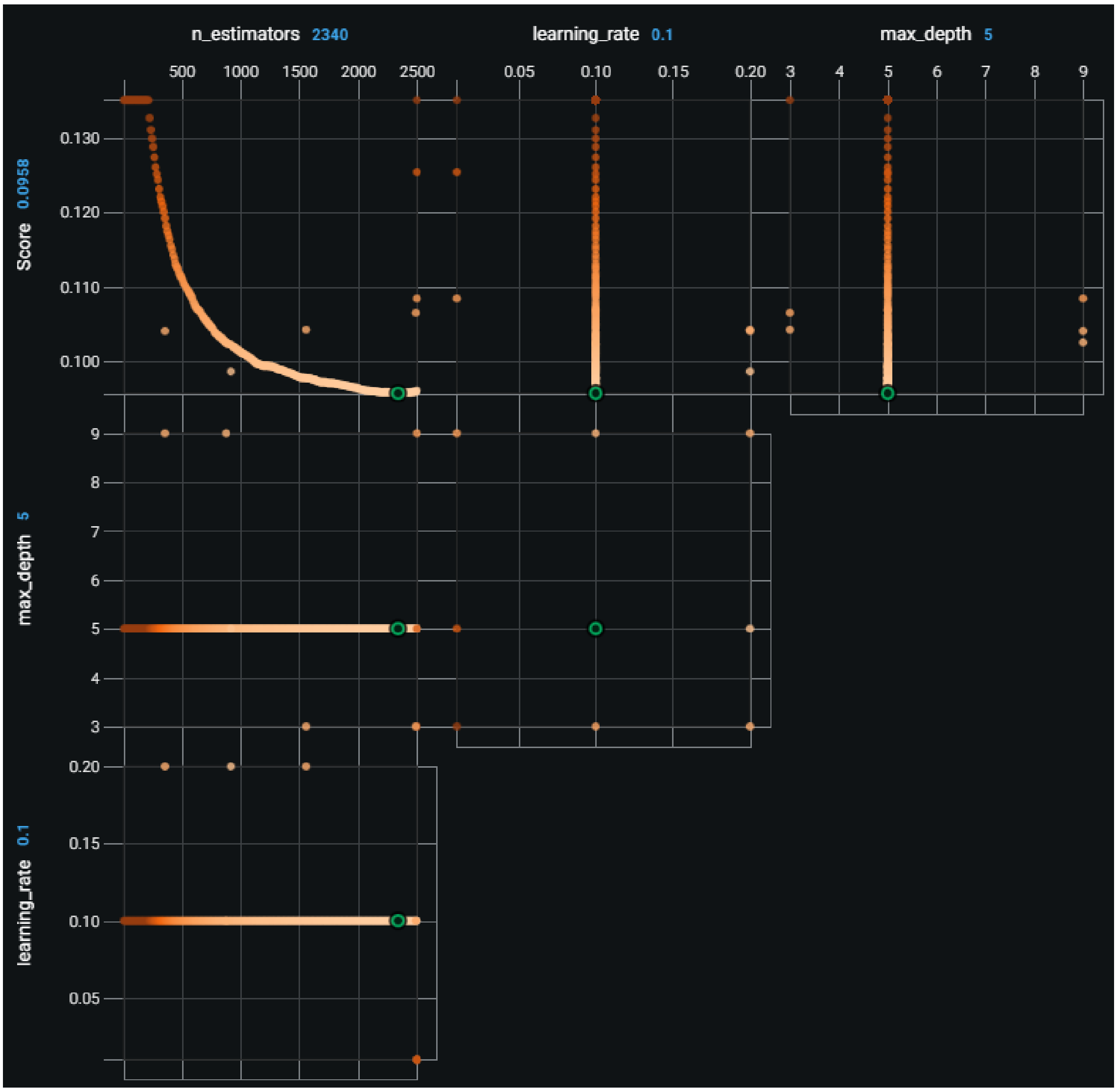
Best of Searched
7

Once we test with this parameter, we get the next results:



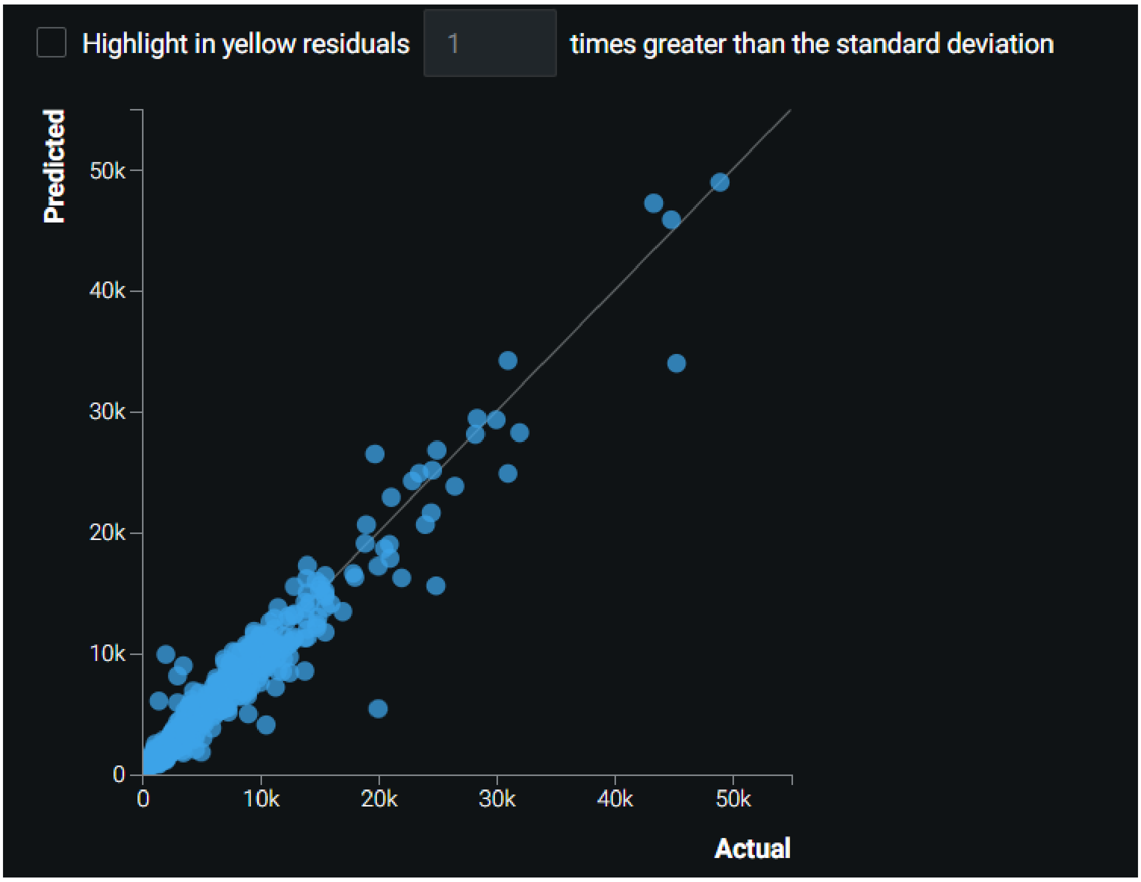
Model Name & Description	Feature List & Sample Size	Validation	Cross Validation	Holdout
eXtreme Gradient Boosted Trees Regressor with Early Stopping (Poisson Loss) Tree-based Algorithm Preprocessing v1 M105 BP64 MONO ❄ 80.0% RECOMMENDED FOR DEPLOYMENT PREPARED FOR DEPLOYMENT	Informative Features 100.0 %	1746.3307 *	1708.8683 *	1665.6140 *
eXtreme Gradient Boosted Trees Regressor with Early Stopping (Poisson Loss) Tree-based Algorithm Preprocessing v1 Tuned from M105 with learning_rate=[0.01,0.1,0.2], max_depth=[3,5,9,11] M126 BP64 TUNED MONO	Informative Features 100.0 %	1699.3475 *	1658.4637 *	

We can see that the new model achieved a better Validation and Cross Validation metrics. So now we need to check which values get the best results, for that we now go to check the model itself and check the values. We get the next results.



We see that the best performance is achieved with a learning rate of 0.1 and a max_depth of 5. With this metrics the RMSE of the cross validation of 1658.46

If we now check the lift chart of the new model, as shown on the next graph, and compare it with the previous one, we see that the model achive a higher precision in the most expensive cars.



□ Highlight in yellow residuals 1 times greater than the standard deviation

Benefits/Concerns

Some of the Benefits, are the extremely easy nature of the program, since you need to only drag and drop to get the results, it is also extremely fast, as it can process multiple different models to get the highest performance. Combining these two features you get a powerful tool, easy to used and fast for creating predictive models.

The concerns revolve around the fact that the software is employed online, meaning that in the event of failure, you lose connections to your built models. Also, in training the data, you need to uploaded, which can be risky or illegal in the case of sensitive information.