Using Machine Learning to predict property prices

The Krieken: Floriane, Augustin, Nadiya

Choosing set of model parameters

based on linear regression model

1) Obvious to drop:

irrelevant, non- precise, dependent on price

`ID, URL, Locality, Municipality, Price per square meter`

2) Obvious to keep:

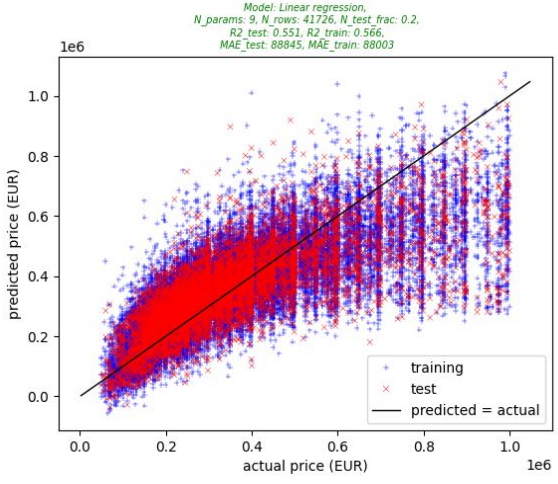
numerical, logical -> numerical, categorial ordinal -> numerical

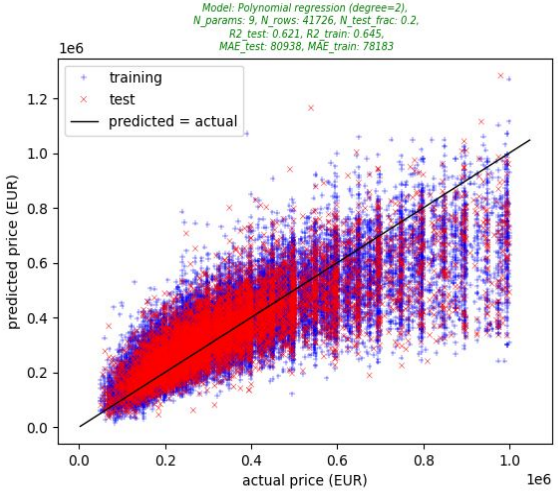
`Bedroom count, Habitable surface, Has terrace, Has parking, Building condition, EPC score`

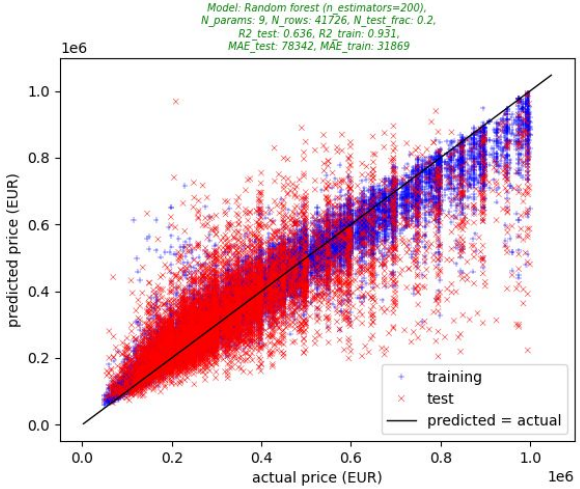
3) Choosing between dependent variables:

parameters	R2_test	R2_train	MAE_test	MAE_train	
"Core 6"	**0.403**	**0.416**	**104632**	**104301**	
+ *Has garden*	-0.000	-0.000	37	-12	
+ *Garden surface*	+0.001	+0.000	-138	-79	←
+ *Type*	+0.007	+0.008	-327	-615	
+ *Subtype*	+0.040	+0.040	-3138	-3683	←
+ *Postcode (-> latitude, longitude)*	+0.058	+0.055	-4720	-4556	
+ *Region*	+0.088	+0.091	-9011	-9616	
+ *Province*	+0.114	+0.116	-12156	-12600	←
"Core 6 "+ 3 ←	**0.551**	**0.566**	**88845**	**88003**	

Choosing model type







Linear regression

Polynomial regression 2nd order

Random forest N_est. = 200

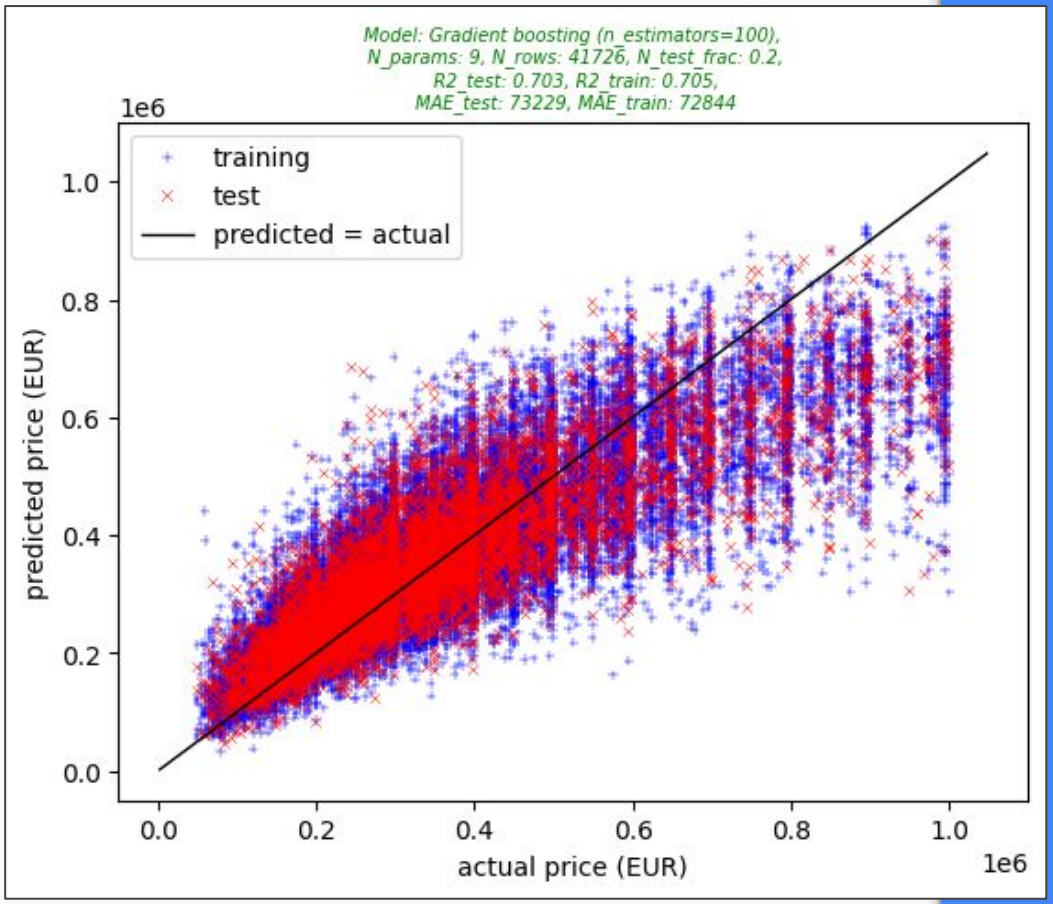
Problems:

over-/under-predicted prices

large scatter

inconsistency training vs testing

And the winner is : Gradient Boosting !



Best fitting line of regression

Train vs Testing

Learning from mistakes

1. Growing a tree
2. Basic prediction
3. Correcting the residuals
4. Growing another tree
5. Then a forest

n_estimators = 100

Models and results

Model	MAE test	R^2 test	MAE train	R^2 train
Linear regression	96969	0.5003	97641	0.4918
Random Forest	62582	0.7632	23639	0.9652
Gradient Boosting	73229	0.7035	72844	0.7053
Polynomial Regression	87150	0.5849	86626	0.5923

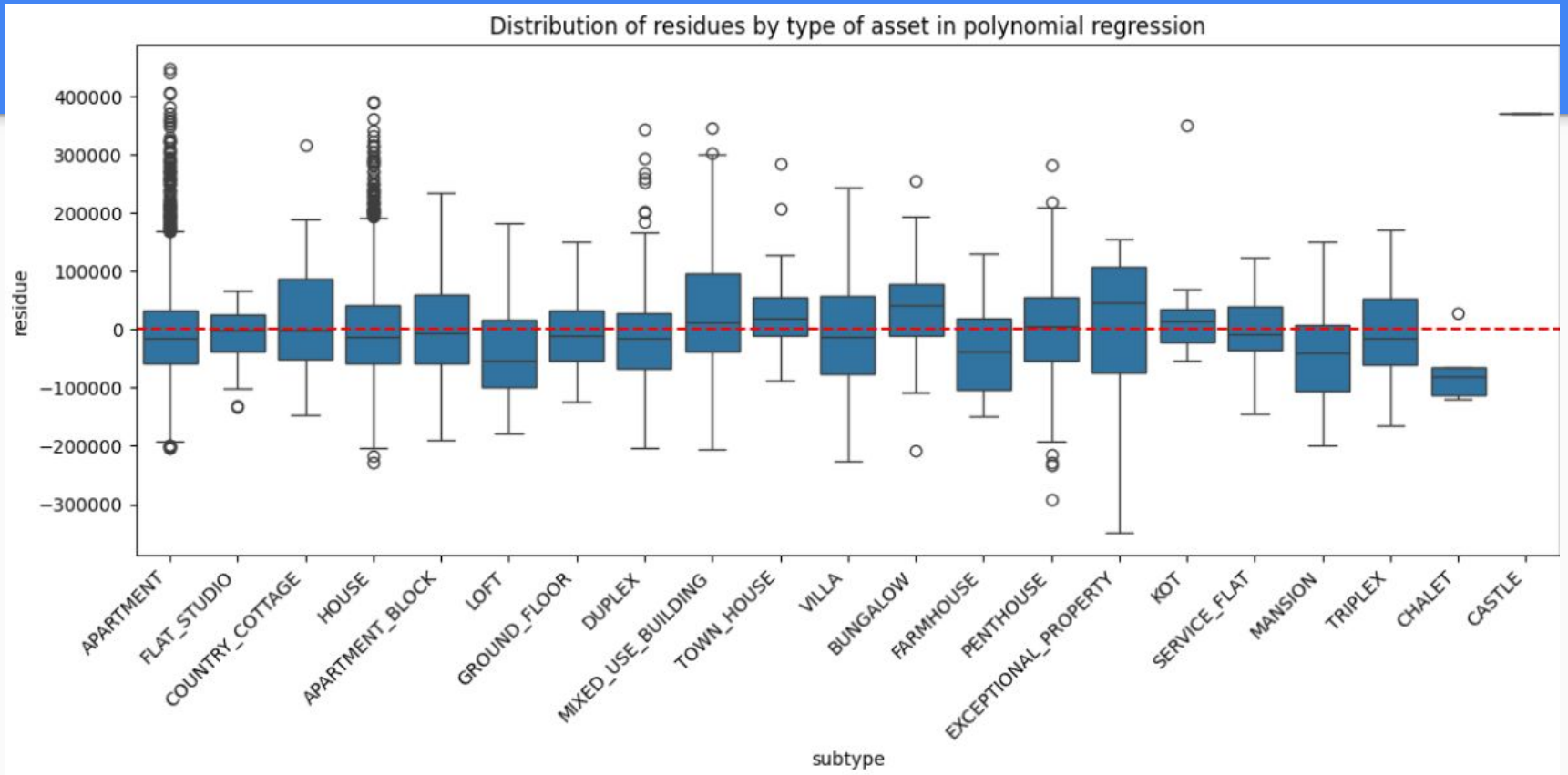
ADVANTAGES

- Highest R^2
- Stable
- No overfitting
- Best management for low prices properties

LIMITATIONS

- MAE is high = 73.000 euros
- Struggle with expensive properties
- Need to improve to have higher R^2

Distribution of residues by type of asset



Distribution of residues by type of asset