Machine learning

for lazy smart people

Choose a Lazy Person To Do a Hard Job Because That Person Will Find an Easy Way To Do It

Bill Gates

more information about the quote :
https://quoteinvestigator.com/2014/02/26/lazy-job/

Dataset cleaning

My idea :

Clean as much as possible Kangaroo dataset

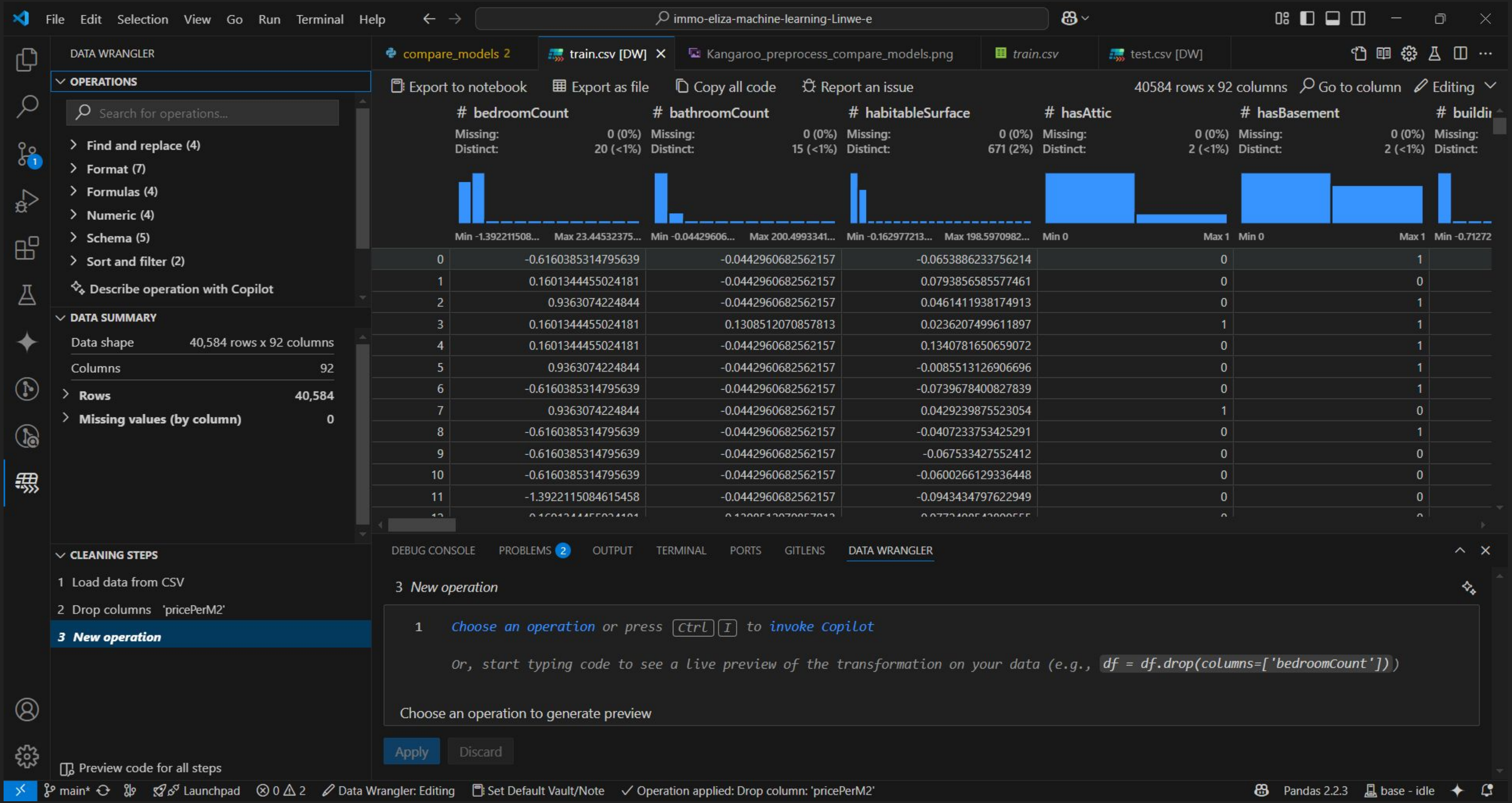
Problem : Post Code

Solution : Giraffe dataset + Kangaroo dataset

Tips for lazy smart beautiful one 🤴 👸

- Data Wrangler
- Steal Robin’s code for preprocessed
- Pycaret library (steal Robin’s idea)

Data Wrangler



immo-eliza-machine-learning-Linwe-e

DATA WRANGLER

OPERATIONS

- Find and replace (4)
- Format (7)
- Formulas (4)
- Numeric (4)
- Schema (5)
- Sort and filter (2)
- Describe operation with Copilot

DATA SUMMARY

Data shape	40,584 rows x 92 columns
Columns	92
Rows	40,584
Missing values (by column)	0

CLEANING STEPS

1 Load data from CSV
2 Drop columns 'pricePerM2'
3 New operation

Preview code for all steps

40584 rows x 92 columns

	# bedroomCount	# bathroomCount	# habitableSurface	# hasAttic	# hasBasement
Missing:	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Distinct:	20 (<1%)	15 (<1%)	671 (2%)	2 (<1%)	2 (<1%)
	Min -1.392211508... Max 23.44532375...	Min -0.04429606... Max 200.4993341...	Min -0.162977213... Max 198.5970982...	Min 0 Max 1	Min 0 Max 1
0	-0.6160385314795639	-0.0442960682562157	-0.0653886233756214	0	1
1	0.1601344455024181	-0.0442960682562157	0.0793856585577461	0	0
2	0.9363074224844	-0.0442960682562157	0.0461411938174913	0	1
3	0.1601344455024181	0.1308512070857813	0.0236207499611897	1	1
4	0.1601344455024181	-0.0442960682562157	0.1340781650659072	0	1
5	0.9363074224844	-0.0442960682562157	-0.0085513126906696	0	1
6	-0.6160385314795639	-0.0442960682562157	-0.0739678400827839	0	1
7	0.9363074224844	-0.0442960682562157	0.0429239875523054	1	0
8	-0.6160385314795639	-0.0442960682562157	-0.0407233753425291	0	1
9	-0.6160385314795639	-0.0442960682562157	-0.067533427552412	0	0
10	-0.6160385314795639	-0.0442960682562157	-0.0600266129336448	0	0
11	-1.3922115084615458	-0.0442960682562157	-0.0943434797622949	0	0

3 New operation

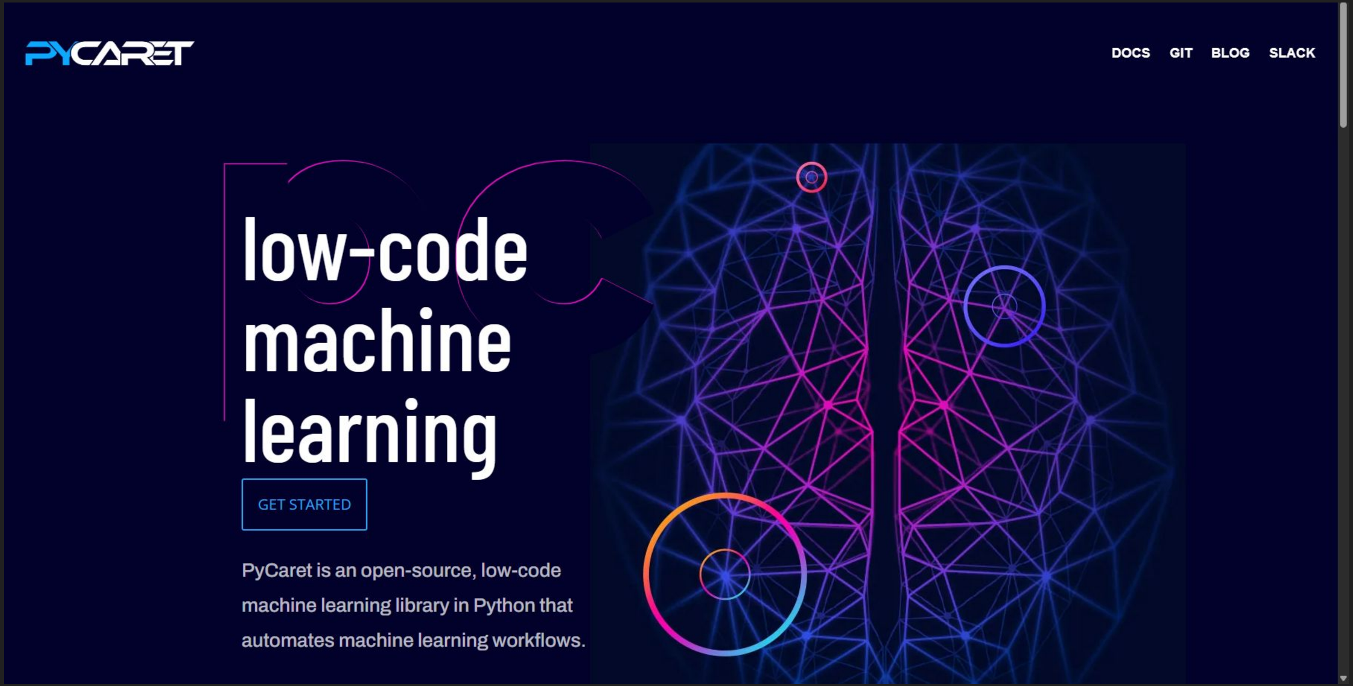
```
Choose an operation or press Ctrl I to invoke Copilot

Or, start typing code to see a live preview of the transformation on your data (e.g., df = df.drop(columns=['bedroomCount']))
```

Choose an operation to generate preview

Apply Discard

main* 0 2 Launchpad Data Wrangler: Editing Set Default Vault/Note ✓ Operation applied: Drop column: 'pricePerM2' Pandas 2.2.3 base - idle

Pycaret



low-code machine learning

PyCaret is an open-source, low-code machine learning library in Python that automates machine learning workflows.

https://pycaret.gitbook.io/docs

Typical workflow in Pycaret

➡️ **Setup**

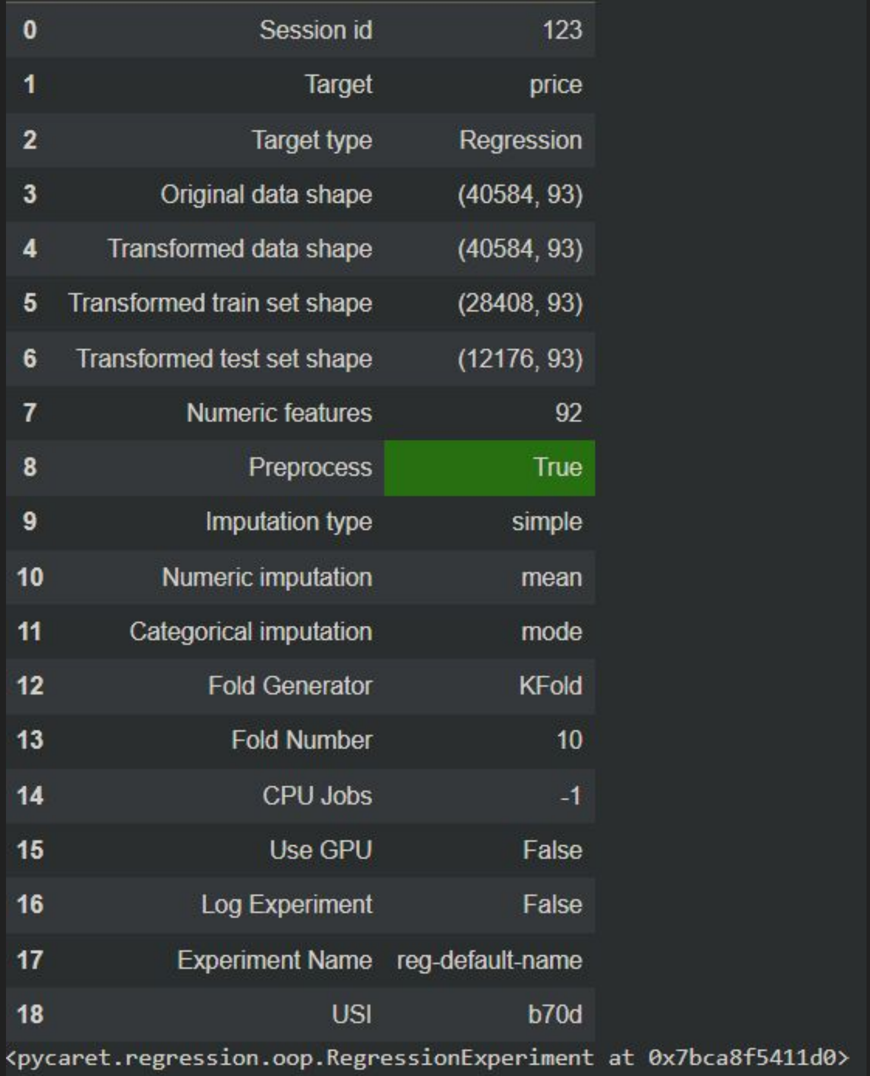
➡️ **Compare Models**

➡️ **Analyze Model**

➡️ **Prediction**

➡️ **Save Model**

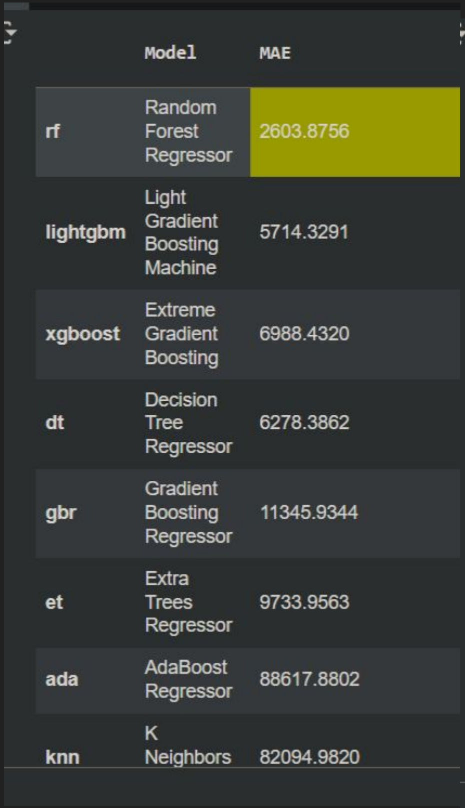
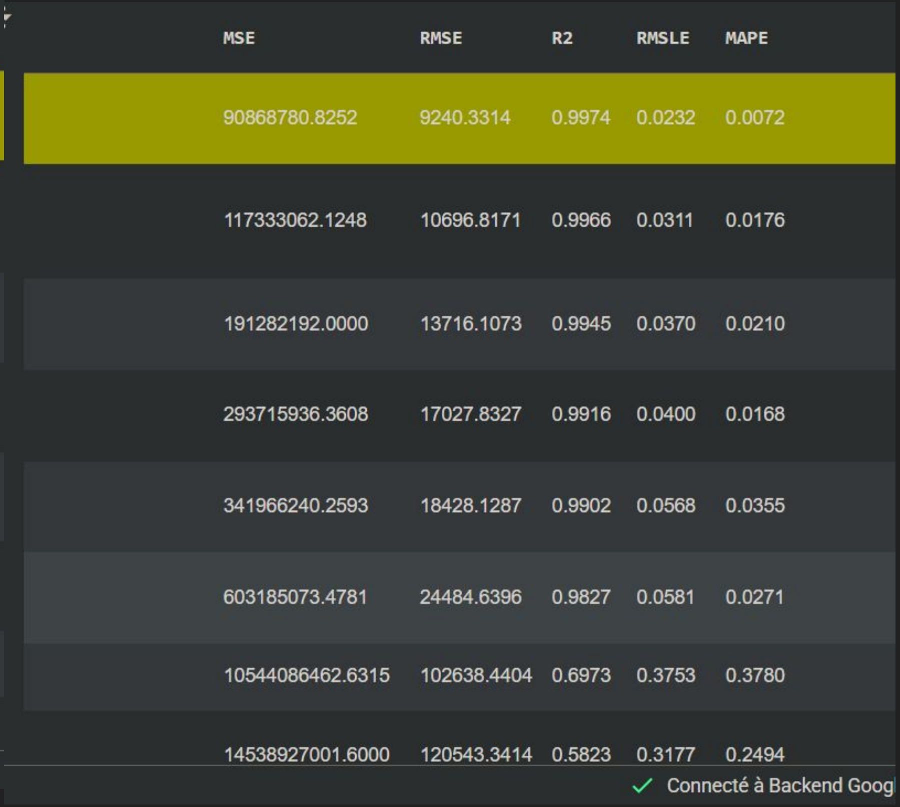
Setup



0	Session id	123
1	Target	price
2	Target type	Regression
3	Original data shape	(40584, 93)
4	Transformed data shape	(40584, 93)
5	Transformed train set shape	(28408, 93)
6	Transformed test set shape	(12176, 93)
7	Numeric features	92
8	Preprocess	True
9	Imputation type	simple
10	Numeric imputation	mean
11	Categorical imputation	mode
12	Fold Generator	KFold
13	Fold Number	10
14	CPU Jobs	-1
15	Use GPU	False
16	Log Experiment	False
17	Experiment Name	reg-default-name
18	USI	b70d

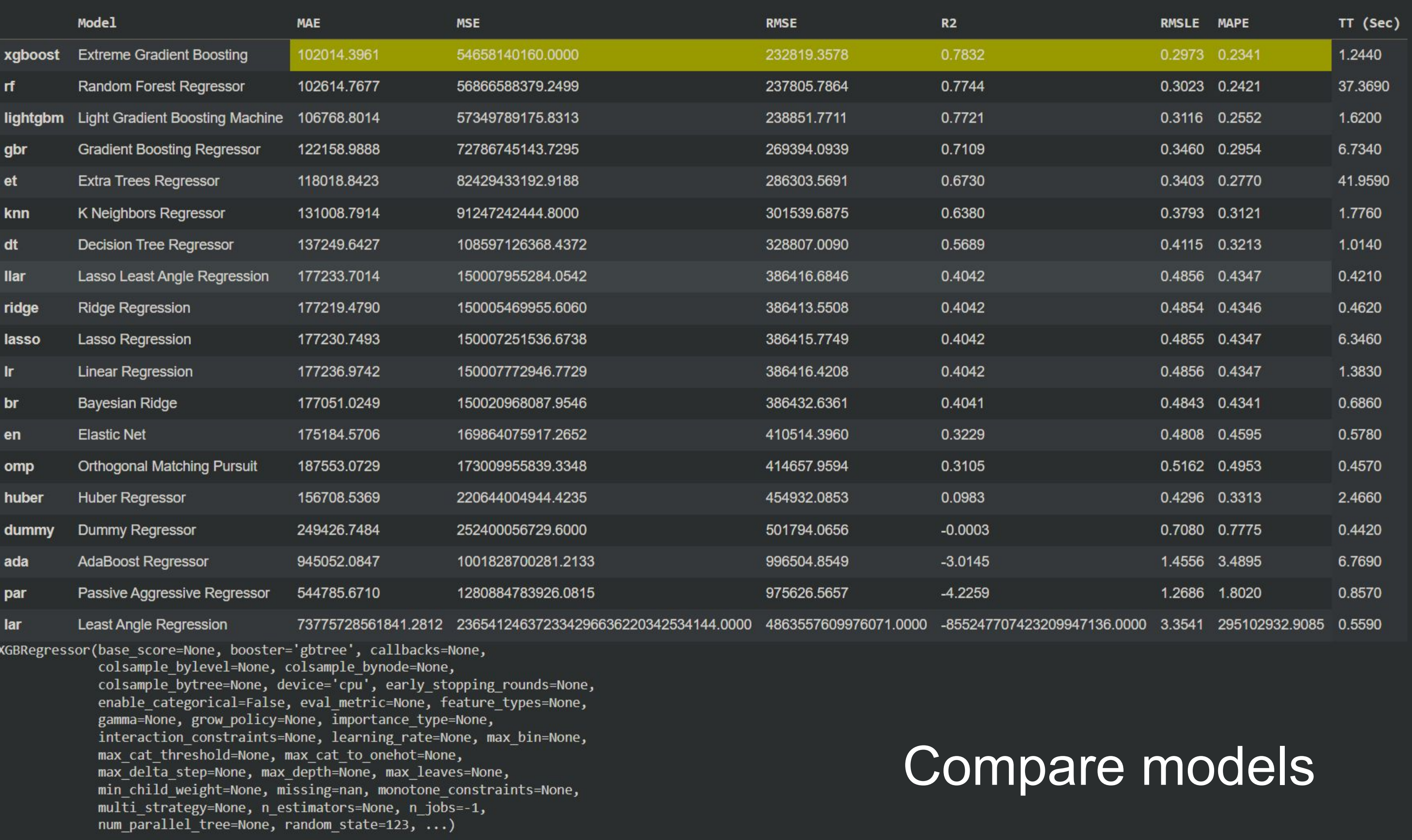
```
<pycaret.regression.oop.RegressionExperiment at 0x7bca8f5411d0>
```

Compare models

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
rf	Random Forest Regressor	2603.8756	90868780.8252	9240.3314	0.9974	0.0232	0.0072
lightgbm	Light Gradient Boosting Machine	5714.3291	117333062.1248	10696.8171	0.9966	0.0311	0.0176
xgboost	Extreme Gradient Boosting	6988.4320	191282192.0000	13716.1073	0.9945	0.0370	0.0210
dt	Decision Tree Regressor	6278.3862	293715936.3608	17027.8327	0.9916	0.0400	0.0168
gbr	Gradient Boosting Regressor	11345.9344	341966240.2593	18428.1287	0.9902	0.0568	0.0355
et	Extra Trees Regressor	9733.9563	603185073.4781	24484.6396	0.9827	0.0581	0.0271
ada	AdaBoost Regressor	88617.8802	10544086462.6315	102638.4404	0.6973	0.3753	0.3780
knn	K Neighbors	82094.9820	14538927001.6000	120543.3414	0.5823	0.3177	0.2494

✓ Connecté à Backend Googl

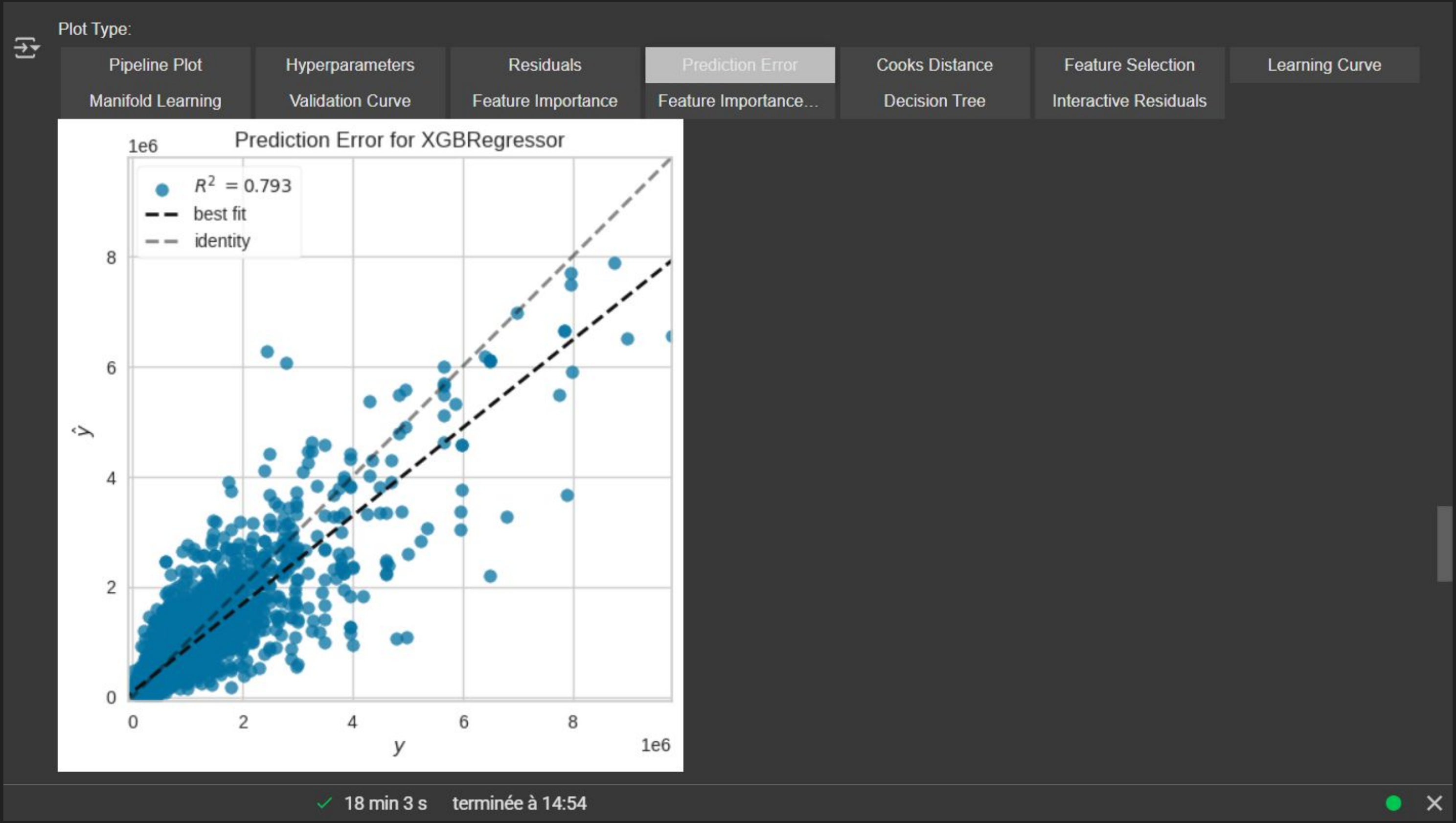


	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
xgboost	Extreme Gradient Boosting	102014.3961	54658140160.0000	232819.3578	0.7832	0.2973	0.2341	1.2440
rf	Random Forest Regressor	102614.7677	56866588379.2499	237805.7864	0.7744	0.3023	0.2421	37.3690
lightgbm	Light Gradient Boosting Machine	106768.8014	57349789175.8313	238851.7711	0.7721	0.3116	0.2552	1.6200
gbr	Gradient Boosting Regressor	122158.9888	72786745143.7295	269394.0939	0.7109	0.3460	0.2954	6.7340
et	Extra Trees Regressor	118018.8423	82429433192.9188	286303.5691	0.6730	0.3403	0.2770	41.9590
knn	K Neighbors Regressor	131008.7914	91247242444.8000	301539.6875	0.6380	0.3793	0.3121	1.7760
dt	Decision Tree Regressor	137249.6427	108597126368.4372	328807.0090	0.5689	0.4115	0.3213	1.0140
llar	Lasso Least Angle Regression	177233.7014	150007955284.0542	386416.6846	0.4042	0.4856	0.4347	0.4210
ridge	Ridge Regression	177219.4790	150005469955.6060	386413.5508	0.4042	0.4854	0.4346	0.4620
lasso	Lasso Regression	177230.7493	150007251536.6738	386415.7749	0.4042	0.4855	0.4347	6.3460
lr	Linear Regression	177236.9742	150007772946.7729	386416.4208	0.4042	0.4856	0.4347	1.3830
br	Bayesian Ridge	177051.0249	150020968087.9546	386432.6361	0.4041	0.4843	0.4341	0.6860
en	Elastic Net	175184.5706	169864075917.2652	410514.3960	0.3229	0.4808	0.4595	0.5780
omp	Orthogonal Matching Pursuit	187553.0729	173009955839.3348	414657.9594	0.3105	0.5162	0.4953	0.4570
huber	Huber Regressor	156708.5369	220644004944.4235	454932.0853	0.0983	0.4296	0.3313	2.4660
dummy	Dummy Regressor	249426.7484	252400056729.6000	501794.0656	-0.0003	0.7080	0.7775	0.4420
ada	AdaBoost Regressor	945052.0847	1001828700281.2133	996504.8549	-3.0145	1.4556	3.4895	6.7690
par	Passive Aggressive Regressor	544785.6710	1280884783926.0815	975626.5657	-4.2259	1.2686	1.8020	0.8570
lar	Least Angle Regression	73775728561841.2812	236541246372334296636220342534144.0000	4863557609976071.0000	-855247707423209947136.0000	3.3541	295102932.9085	0.5590

```
XGBRegressor(base_score=None, booster='gbtree', callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device='cpu', early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=-1,
             num_parallel_tree=None, random_state=123, ...)
```

Compare models

Analyze model



✓ 18 min 3 s   terminée à 14:54

Be grateful to others who facilitate your work, making productive laziness possible

❤️