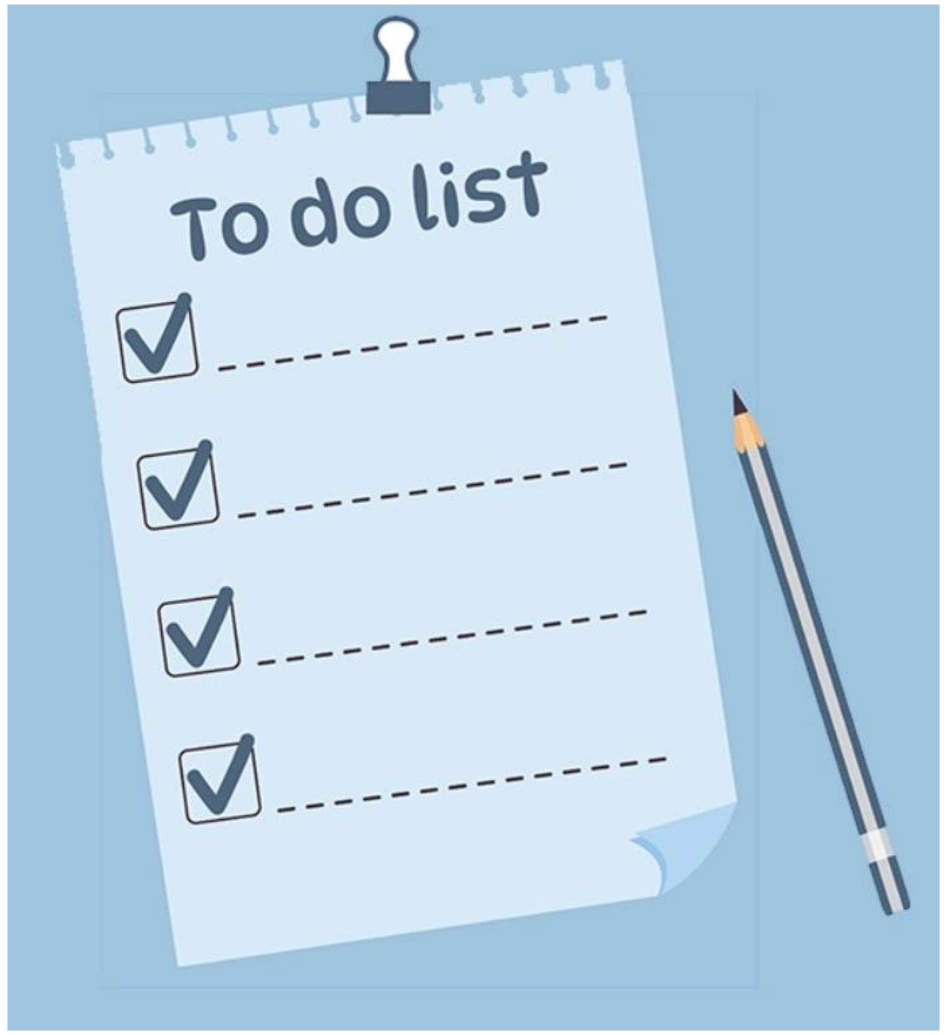
House Predection

Team ML A

21 March 2025

List of Contents

- Background & Problem Statement
- Objectives & Scope
- Data Collection
- Data Preprocessing
- Creating Baseline Mode
- Feature Engineering
- Modelling
- Model Tuning
- Conclusion



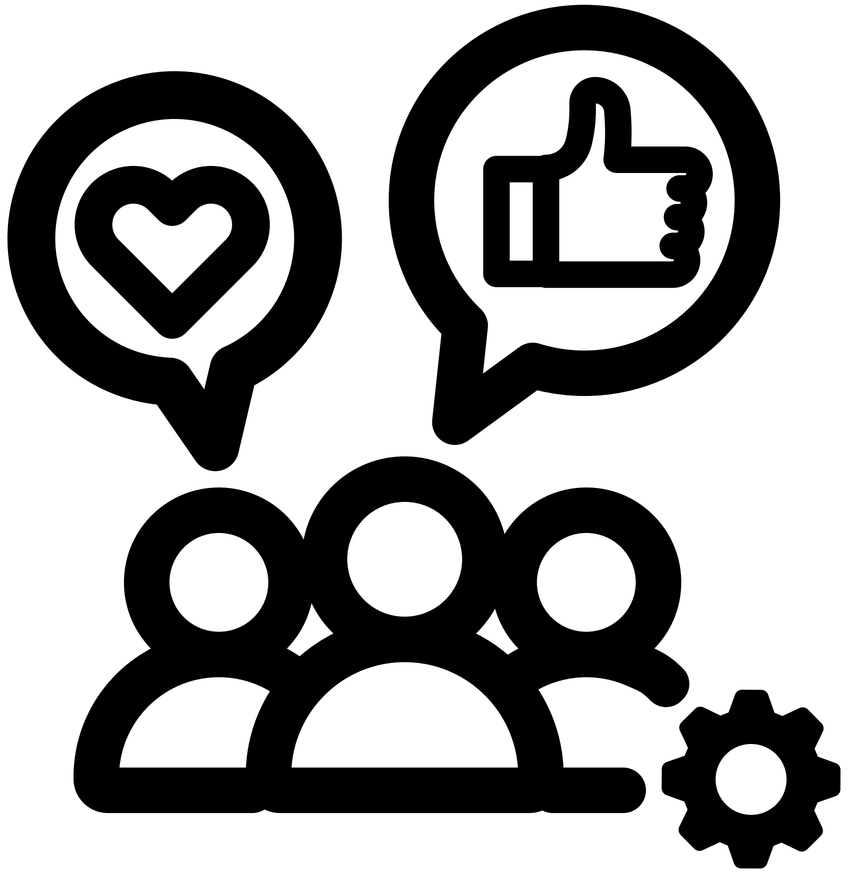
Background & Problem Statement

Background

The real estate market is dynamic and constantly changing, making house price prediction an important tool for buyers, sellers, investors, and real estate professionals. However, macroeconomic factors (interest rates, inflation, government policies), microeconomic factors (location, building size, facilities), and socio-demographic factors (population growth, urbanization trends) often drive property prices.

Problem Statement

Accurate prediction is needed to help stakeholders make the right decisions, whether it's buying a dream home or planning a profitable investment. In recent years, **machine learning** has emerged as a game changer in this field, offering unprecedented accuracy and insights compared to traditional statistical methods.



Reference: https://www.appliedaicourse.com/blog/house-price-prediction-using-machine-learning/

Objectives & Scope

Objective:
Create and compare machine learning model that predicts House Price. The goal is to identify / predict hous price based on important / strategic factors

Scope:
This project using data with house pricing information. The analysis will focus on applying prediction technique using Linear Regression, Random Forest and XGBoost

Data Collection

Data Source:

The dataset can be downloaded from here

Some Supporting Documents:

dataDefinision.md
data_description.txt
CategoricalEncodingnyaPakaiApa.xlsx

Data Preprocessing

- Feature Name Normalization

```
1  df_01_normalize = normalize_column_names(df_original.copy(deep=True))
```

✓ 0.0s

num_mssubclass ,though the content is number, but I think it is only label (based on data description) so it should be handled as text / criteria

```
1  df_01_normalize=df_01_normalize.rename(columns={'num_mssubclass': 'txt_mssubclass'})
```

✓ 0.0s

```
1  df=df_01_normalize.copy(deep=True)
2  print(df.columns)
```

✓ 0.0s

```
Index(['num_id', 'txt_mssubclass', 'txt_mszoning', 'num_lotfrontage',
       'num_lotarea', 'txt_street', 'txt_alley', 'txt_lotshape',
       'txt_landcontour', 'txt_utilities', 'txt_lotconfig', 'txt_landslope',
       'txt_neighborhood', 'txt_condition1', 'txt_condition2', 'txt_bldgtype',
       'txt_housestyle', 'num_overallqual', 'num_overallcond', 'num_yearbuilt',
       'num_yearremodadd', 'txt_roofstyle', 'txt_roofmatl', 'txt_exterior1st',
       'txt_exterior2nd', 'txt_masvnrtype', 'num_masvnrarea', 'txt_exterqual',
       'txt_extercond', 'txt_foundation', 'txt_bsmtqual', 'txt_bsmtcond',
       'txt_bsmtexposure', 'txt_bsmtfintype1', 'num_bsmtfinsf1',
       'txt_bsmtfintype2', 'num_bsmtfinsf2', 'num_bsmtunfsf',
       'num_totalbsmtsf', 'txt_heating', 'txt_heatingqc', 'txt_centralair',
       'txt_electrical', 'num_1stflrsf', 'num_2ndflrsf', 'num_lowqualfinsf',
       'num_grlivarea', 'num_bsmtfullbath', 'num_bsmthalfbath', 'num_fullbath',
       'num_halfbath', 'num_bedroomabvgr', 'num_kitchenabvgr',
       'txt_kitchenqual', 'num_totrmsabvgrd', 'txt_functional',
       'num_fireplaces', 'txt_fireplacequ', 'txt_garagetype',
       'num_garageyrblt', 'txt_garagefinish', 'num_garagecars',
       'num_garagearea', 'txt_garagequal', 'txt_garagecond', 'txt_paveddrive',
       'num_wooddecksf', 'num_openporchsf', 'num_enclosedporch',
       'num_3ssnporch', 'num_screenporch', 'num_poolarea', 'txt_poolqc',
       'txt_fence', 'txt_miscfeature', 'num_miscval', 'num_mosold',
       'num_yrsold', 'txt_saletype', 'txt_salecondition', 'num_saleprice'],
      dtype='object')
```

Data Preprocessing

- Handling Missing Values

Handling Missing Values

```
printMissingInfo(df)
```

✓ 0.0s

	# Missing Count	# Missing Percentage
txt_poolqc	1453	99.52054794520548
txt_miscfeature	1406	96.30136986301369
txt_alley	1369	93.76712328767123
txt_fence	1179	80.75342465753424
txt_fireplacequ	690	47.26027397260274
num_lotfrontage	259	17.73972602739726
txt_garagefinish	81	5.5479452054794525
txt_garagecond	81	5.5479452054794525
txt_garagequal	81	5.5479452054794525
num_garageyrblt	81	5.5479452054794525
txt_garagetype	81	5.5479452054794525
txt_bsmtexposure	38	2.6027397260273974
txt_bsmtfintype2	38	2.6027397260273974
txt_bsmtfintype1	37	2.5342465753424657
txt_bsmtcond	37	2.5342465753424657
txt_bsmtqual	37	2.5342465753424657
num_masvnrarea	8	0.547945205479452
txt_masvnrtype	8	0.547945205479452
txt_electrical	1	0.0684931506849315

19 rows x 2 cols 50 per page

Data Preprocessing

- Define Which One Will be One Hot Encoding and Which one will be Target Encoding

based on checking (see : CategoricalEncodingnyaPakaiApa.xlsx)
####################

no need encoding
'txt_overallqual', 'txt_overallcond',

OHE
'txt_street','txt_alley','txt_landslope','txt_centralair','txt_paveddrive',

Target Encoding
'txt_mssubclass','txt_mszoning','txt_lotshape','txt_landcontour','txt_utilities',
'txt_lotconfig','txt_neighborhood','txt_condition1','txt_condition2','txt_bldgtype',
'txt_housestyle','txt_roofstyle','txt_roofmatl','txt_exterior1st','txt_exterior2nd',
'txt_masvnrtype','txt_exterqual','txt_extercond','txt_foundation','txt_bsmtqual',
'txt_bsmtcond','txt_bsmtexposure','txt_bsmtfintype1','txt_bsmtfintype2','txt_heating',
'txt_heatingqc','txt_electrical','txt_kitchenqual','txt_functional','txt_fireplacequ',
'txt_garagetype','txt_garagefinish','txt_garagequal','txt_garagecond','txt_poolqc',
'txt_fence','txt_miscfeature','txt_saletype','txt_salecondition',

untuk :

- 'txt_mssubclass',
- 'txt_mszoning',
- 'txt_utilities',
- 'txt_condition2',
- 'txt_exterior1st',
- 'txt_exterior2nd',
- 'txt_masvnrtype',
- 'txt_exterqual',
- 'txt_bsmtqual',
- 'txt_bsmtcond',
- 'txt_kitchenqual',
- 'txt_functional',
- 'txt_poolqc',
- 'txt_miscfeature',
- 'txt_saletype',

jumlah unique value pada data is less than what is listed in data description / definition .
but for the simplicity of this project (and we don't have new data that might have
categorical values other than that already listed in the curren data),
i decided not to include any categorical value that is listed in the data description / definition but
not listed in the training/test data

Data Preprocessing

Target Encoding After Splitting the data (to avoid data leakage from test data)

Target Encoding after Splitting

```python
df_train, encodernya = target_encode_columns_general(df_train, target_col='num_saleprice'
                        , columns=['txt_mssubclass','txt_mszoning','txt_lotshape','txt_landcontour','txt_utilities','txt_lotconfig
                        'txt_neighborhood','txt_condition1','txt_condition2','txt_bldgtype','txt_housestyle',
                        'txt_roofstyle','txt_roofmatl','txt_exterior1st','txt_exterior2nd','txt_masvnrtype',
                        'txt_exterqual','txt_extercond','txt_foundation','txt_bsmtqual','txt_bsmtcond',
                        'txt_bsmtexposure','txt_bsmtfintype1','txt_bsmtfintype2','txt_heating','txt_heatingqc',
                        'txt_electrical','txt_kitchenqual','txt_functional','txt_fireplacequ','txt_garagetype',
                        'txt_garagefinish','txt_garagequal','txt_garagecond','txt_poolqc','txt_fence',
                        'txt_miscfeature','txt_saletype','txt_salecondition'])
```

✓ 0.6s Python

```python
df_test_full = df_test.copy()

#endcoder only process the feature that need to be target encoded so does the encoding result
#so we need to copy the full df , DROP the features being encoded, and ADD the encoded result back

df_transformed_cols = encodernya.transform(df_test[['txt_mssubclass','txt_mszoning','txt_lotshape','txt_landcontour','txt_utilities','txt_lotconfig',
                        'txt_neighborhood','txt_condition1','txt_condition2','txt_bldgtype','txt_housestyle',
                        'txt_roofstyle','txt_roofmatl','txt_exterior1st','txt_exterior2nd','txt_masvnrtype',
                        'txt_exterqual','txt_extercond','txt_foundation','txt_bsmtqual','txt_bsmtcond',
                        'txt_bsmtexposure','txt_bsmtfintype1','txt_bsmtfintype2','txt_heating','txt_heatingqc',
                        'txt_electrical','txt_kitchenqual','txt_functional','txt_fireplacequ','txt_garagetype',
                        'txt_garagefinish','txt_garagequal','txt_garagecond','txt_poolqc','txt_fence',
                        'txt_miscfeature','txt_saletype','txt_salecondition']])


df_test_full = df_test_full.drop(columns=['txt_mssubclass','txt_mszoning','txt_lotshape','txt_landcontour','txt_utilities','txt_lotconfig',
                        'txt_neighborhood','txt_condition1','txt_condition2','txt_bldgtype','txt_housestyle',
                        'txt_roofstyle','txt_roofmatl','txt_exterior1st','txt_exterior2nd','txt_masvnrtype',
                        'txt_exterqual','txt_extercond','txt_foundation','txt_bsmtqual','txt_bsmtcond',
                        'txt_bsmtexposure','txt_bsmtfintype1','txt_bsmtfintype2','txt_heating','txt_heatingqc',
                        'txt_electrical','txt_kitchenqual','txt_functional','txt_fireplacequ','txt_garagetype',
                        'txt_garagefinish','txt_garagequal','txt_garagecond','txt_poolqc','txt_fence',
                        'txt_miscfeature','txt_saletype','txt_salecondition'])

df_test = pd.concat([df_test_full, df_transformed_cols], axis=1)
```

Data Preprocessing

One Hot Encoding

One Hot Encoding

```
df = one_hot_encode_columns_general(df, ['txt_street','txt_alley','txt_landslope','txt_centralair','txt_paveddrive'])
df.columns
```

[36] ✓ 0.0s

```
Index(['num_id', 'txt_mssubclass', 'txt_mszoning', 'num_lotarea',
       'txt_lotshape', 'txt_landcontour', 'txt_utilities', 'txt_lotconfig',
       'txt_neighborhood', 'txt_condition1', 'txt_condition2', 'txt_bldgtype',
       'txt_housestyle', 'num_overallqual', 'num_overallcond', 'num_yearbuilt',
       'num_yearremodadd', 'txt_roofstyle', 'txt_roofmatl', 'txt_exterior1st',
       'txt_exterior2nd', 'txt_masvnrtype', 'num_masvnrarea', 'txt_exterqual',
       'txt_extercond', 'txt_foundation', 'txt_bsmtqual', 'txt_bsmtcond',
       'txt_bsmtexposure', 'txt_bsmtfintype1', 'num_bsmtfinsf1',
       'txt_bsmtfintype2', 'num_bsmtfinsf2', 'num_bsmtunfsf',
       'num_totalbsmtsf', 'txt_heating', 'txt_heatingqc', 'txt_electrical',
       'num_1stflrsf', 'num_2ndflrsf', 'num_lowqualfinsf', 'num_grlivarea',
       'num_bsmtfullbath', 'num_bsmthalfbath', 'num_fullbath', 'num_halfbath',
       'num_bedroomabvgr', 'num_kitchenabvgr', 'txt_kitchenqual',
       'num_totrmsabvgrd', 'txt_functional', 'num_fireplaces',
       'txt_fireplacequ', 'txt_garagetype', 'num_garageyrblt',
       'txt_garagefinish', 'num_garagecars', 'num_garagearea',
       'txt_garagequal', 'txt_garagecond', 'num_wooddecksf', 'num_openporchsf',
       'num_enclosedporch', 'num_3ssnporch', 'num_screenporch', 'num_poolarea',
       'txt_poolqc', 'txt_fence', 'txt_miscfeature', 'num_miscval',
       'num_mosold', 'num_yrsold', 'txt_saletype', 'txt_salecondition',
       'num_saleprice', 'num_lotfrontage', 'txt_street_Grvl',
       'txt_street_Pave', 'txt_alley_Grvl', 'txt_alley_NA', 'txt_alley_Pave',
       'txt_landslope_Gtl', 'txt_landslope_Mod', 'txt_landslope_Sev',
       'txt_centralair_N', 'txt_centralair_Y', 'txt_paveddrive_N',
       'txt_paveddrive_P', 'txt_paveddrive_Y'],
      dtype='object')
```

Creating Base Line Models

- **Correlation Matrix**

```
                               Model          RMSE           MAE       MAPE  \
0                                  -      0.000000      0.000000   0.000000
1  baseline_model_linear_regression  32170.161993  20080.659064  11.820330
2      baseline_model_random_Forest  28636.283593  17416.832671  10.520635
3                baseline_model_XGB  29502.500001  16806.905982  10.269786

         R2
0  0.000000
1  0.865075
2  0.893090
3  0.886524
```

Feature Engineering

- **Create New Features**

combined features

```python
def add_combined_features(dfnya):

    #custom feature ini dibuat as simple as grouping some feature that have similarity on their characteristics on supporting the house price
    #you can find the process of the grouping in supportind document "CategoricalEncodingnyaPakaiApa.xlsx"

    dfnya['new_qualities'] =    dfnya['num_overallqual'] + dfnya['txt_exterqual'] + dfnya['txt_bsmtqual'] + dfnya['txt_heatingqc'] + dfnya['num_lowqualfinsf'] + dfnya['num_ha
    dfnya['new_condition'] =    dfnya['txt_condition1'] + dfnya['txt_condition2'] + dfnya['num_overallcond'] + dfnya['txt_extercond'] + dfnya['txt_bsmtcond'] + dfnya['txt_hea
    dfnya['new_square'] =   dfnya['num_lotarea'] + dfnya['num_masvnrarea'] + dfnya['num_bsmtfinsf1'] + dfnya['num_bsmtfinsf2'] + dfnya['num_bsmtunfsf'] + dfnya['num_totalbsmt
    dfnya['new_counts'] =   dfnya['num_totalbsmtsf'] + dfnya['num_bsmtfullbath'] + dfnya['num_bsmthalfbath'] + dfnya['num_fullbath'] + dfnya['num_bedroomabvgr'] + dfnya['num_
    dfnya['new_types'] =    dfnya['txt_mssubclass'] + (dfnya['txt_street_Grvl']*1)+ (dfnya['txt_street_Pave']*2)+(dfnya['txt_alley_NA'] *0) + (dfnya['txt_alley_Grvl'] *1) + (
    dfnya['new_interiorexterior'] =     dfnya['num_overallqual'] + dfnya['txt_roofstyle'] + dfnya['txt_roofmatl'] + dfnya['txt_exterior1st'] + dfnya['txt_exterior2nd'] + dfny
    dfnya['new_neighbour'] =    dfnya['txt_mssubclass'] + dfnya['txt_mszoning'] + dfnya['num_lotfrontage'] + (dfnya['txt_street_Grvl']*1)+ (dfnya['txt_street_Pave']*2) + (dfn
    dfnya['new_facilities'] =   dfnya['txt_utilities'] + dfnya['txt_heating'] + dfnya['txt_electrical'] + dfnya['num_fireplaces'] + dfnya['txt_miscfeature'] + dfnya['num_misc
    dfnya['new_shapes'] =   dfnya['txt_lotshape'] + dfnya['txt_lotconfig'] + dfnya['txt_housestyle'] + dfnya['txt_bsmtexposure']
    dfnya['new_time'] =     ((dfnya['num_yrsold']- ((dfnya['num_yearbuilt'] + dfnya['num_yearremodadd'])/2))+(dfnya['num_yrsold']- dfnya['num_garageyrblt']))/2
    return dfnya
```

✓ 0.0s

Python

Feature Engineering

- **Other steps :**

- Drop features with weak correlation to target (<=0.05)
- Drop features with high correlation between features (multi collinearity) (>0.8)
- ~~Drop features with high VIF Score (>5)~~ ;
  canceled since it making model performance
  worse (by removing some important features)

Modelling

- **Modelling after feature engineering prior to Tuning**

```
1 results_df
```

✓ 0.0s Open 'results_df' in Data Wrangler

	Model	RMSE	MAE	MAPE	R2
0	-	0.0	0.0	0.0	0.0
1	baseline_model_linear_regression	32170.16199264107	20080.65906364246	11.820329982934373	0.86507492710505
2	baseline_model_random_Forest	28636.283592974698	17416.83267123287	10.520634941837901	0.893089717977197
3	baseline_model_XGB	29502.50000135622	16806.90598244863	10.26978639957796	0.8865240561067067
4	feauterEng_model_linear_regression	32811.140791704776	21068.336528171305	12.847550251115214	0.8596446968520945
5	feauterEng_model_random_Forest	28428.91196131549	17292.90373287671	10.992029606489169	0.8946325079959523
6	feauterEng_model_XGB	35132.40002770671	19010.372324486303	11.127679146282079	0.8390830457053735

7 rows x 9 cols 10 per page Page 1 of 1

- Got performance drop for XGB
- Got slight performance drop on Linear Regression
- Got promising result with Random Forest

Model Tuning

- **Tune the Model with Random Search**

✓ 0.0s

	Model	RMSE	MAE	MAPE	R2
0	-	0.0000	0.0000	0.0000	0.0000
1	baseline_model_linear_regression	32,170.1620	20,080.6591	11.8203	0.8651
2	baseline_model_random_Forest	28,636.2836	17,416.8327	10.5206	0.8931
3	baseline_model_XGB	29,502.5000	16,806.9060	10.2698	0.8865
4	feauterEng_model_linear_regression	32,811.1408	21,068.3365	12.8476	0.8596
5	feauterEng_model_random_Forest	28,428.9120	17,292.9037	10.9920	0.8946
6	feauterEng_model_XGB	35,132.4000	19,010.3723	11.1277	0.8391
7	tuned_RndSrch_model_RF	29,302.0300	17,340.6957	11.0487	0.8881
8	tuned_RndSrch_model_XGB	27,488.0232	16,400.9090	10.0323	0.9015

- XGB Model is the current Best Model
- Random Forest without tuning #2 model

Model Stacking Models (Ensemble)

- **Stacking 2 Models : Tuned_XGB and Untuned Feature Enginered Random Forest Model**

0.0s

	Model	RMSE	MAE	MAPE	R2
8	tuned_RndSrch_model_XGB	27,488.0232	16,400.9090	10.0323	0.9015
5	feauterEng_model_random_Forest	28,428.9120	17,292.9037	10.9920	0.8946
2	baseline_model_random_Forest	28,636.2836	17,416.8327	10.5206	0.8931
7	tuned_RndSrch_model_RF	29,297.1201	17,325.6348	11.0337	0.8881
3	baseline_model_XGB	29,502.5000	16,806.9060	10.2698	0.8865
1	baseline_model_linear_regression	32,170.1620	20,080.6591	11.8203	0.8651
4	feauterEng_model_linear_regression	32,811.1408	21,068.3365	12.8476	0.8596
6	feauterEng_model_XGB	35,132.4000	19,010.3723	11.1277	0.8391
0	-	0.0000	0.0000	0.0000	0.0000

	Model	RMSE	MAE	MAPE	R2
9	ensembel_tuned_rf_xgb	26,859.0342	16,223.4370	9.9526	0.9059
8	tuned_RndSrch_model_XGB	27,488.0232	16,400.9090	10.0323	0.9015
5	feauterEng_model_random_Forest	28,428.9120	17,292.9037	10.9920	0.8946
2	baseline_model_random_Forest	28,636.2836	17,416.8327	10.5206	0.8931
7	tuned_RndSrch_model_RF	29,297.1201	17,325.6348	11.0337	0.8881
3	baseline_model_XGB	29,502.5000	16,806.9060	10.2698	0.8865
1	baseline_model_linear_regression	32,170.1620	20,080.6591	11.8203	0.8651
4	feauterEng_model_linear_regression	32,811.1408	21,068.3365	12.8476	0.8596
6	feauterEng_model_XGB	35,132.4000	19,010.3723	11.1277	0.8391
0	-	0.0000	0.0000	0.0000	0.0000

Conclusion

Combining the tuned XGB and the feature-engineered Random Forest resulted in the model with the lowest error in this experiment; although the improvement is not very significant, it at least managed to smooth out the previous best result (the tuned XGB model).