4_Feature_Selection

February 13, 2026

[2]:
```
# Import libraries
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from utils import *
```

[3]:
```
# Load king_country_dataset
df = pd.read_csv('Data/cleaned_house_sales.csv')
df.head()
```

[3]:
```
      price  bedrooms  bathrooms  sqft_living  sqft_lot  floors  waterfront  \
0  221900.0         3       1.00         1180      5650     1.0           0
1  538000.0         3       2.25         2570      7242     2.0           0
2  180000.0         2       1.00          770     10000     1.0           0
3  604000.0         4       3.00         1960      5000     1.0           0
4  510000.0         3       2.00         1680      8080     1.0           0

   view  condition  grade  …  yr_built  yr_renovated  zipcode      lat  \
0     0          3      7  …      1955             0    98178  47.5112
1     0          3      7  …      1951          1991    98125  47.7210
2     0          3      6  …      1933             0    98028  47.7379
3     0          5      7  …      1965             0    98136  47.5208
4     0          3      8  …      1987             0    98074  47.6168

      long  sqft_living15  sqft_lot15  year_sold  month_sold  day_sold
0 -122.257           1340        5650       2014          10        13
1 -122.319           1690        7639       2014          12         9
2 -122.233           2720        8062       2015           2        25
3 -122.393           1360        5000       2014          12         9
4 -122.045           1800        7503       2015           2        18

[5 rows x 22 columns]
```

[4]:
```
# Load king_country_dataset
base_metrics = pd.read_csv('Metrics/baseline_metrics.csv')
base_metrics = base_metrics.drop([2, 3], axis=0)
base_metrics
```

```
[4]:                   Model  Split      R2  Adjusted_R2          MAE  \
     0       LinearRegression  train  0.6977       0.6974  125948.1118
     1       LinearRegression   test  0.7162       0.7148  125985.6747
     4  RandomForestRegressor  train  0.9820       0.9819   25948.4444
     5  RandomForestRegressor   test  0.8959       0.8954   67269.8770
     6           XGBRegressor  train  0.9780       0.9780   39126.7558
     7           XGBRegressor   test  0.9015       0.9010   65712.7448

              RMSE    MAPE                                           Comments
     0  202864.5703  0.2561                                     Baseline model
     1  191531.3335  0.2596                                     Baseline model
     4   49547.1197  0.0486  Baseline, no normalization, random_state 13, d…
     5  115994.2051  0.1271  Baseline, no normalization, random_state 13, d…
     6   54676.7727  0.0872        Baseline, no normalization, default values.
     7  112860.4995  0.1246        Baseline, no normalization, default values.
```

In the OLS we see that the p-values of some columns are high. We would like to test if without those columns the models work better or change. The columns are:
- sqft_lot - floors

The Ridge and Lasso Coefficients were also low in some columns and we would like to analize the same on: - month_sold - day_sold - yr_renovated

0.1 Drop 'sqft_lot'

Defining target and features

[ ]:
```python
X = df.drop(['price', 'sqft_lot'], axis=1) # Features
y = df['price'] # Target
```

Split the data

[ ]:
```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2,␣
 ↪random_state=13)
```

0.1.1 Linear Regression

[ ]:
```python
from sklearn.linear_model import LinearRegression
 # Create the Linear Regression estimator
lm = LinearRegression()

# Perform the fitting
lm.fit(X_train, y_train)
```

Evaluation

[ ]:
```python
metrics_df = create_metrics_df()
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_train,
    y_train,
    "train",
    "Whithout sqft_lot"
)

metrics_df
```

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_test,
    y_test,
    "test",
    "Whithout sqft_lot"
)

metrics_df
```

0.1.2 Random Forest

[ ]:

```
from sklearn.ensemble import RandomForestRegressor
rf_regressor = RandomForestRegressor(random_state=13)

rf_regressor.fit(X_train, y_train)
```

Evaluation

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_train,
    y_train,
    "train",
    "Whithout sqft_lot"
)
```

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_test,
    y_test,
    "test",
    "Whithout sqft_lot"
)
metrics_df
```

0.1.3 XGBoost

[ ]:

```python
import xgboost as xgb

xgb_reg = xgb.XGBRegressor()
xgb_reg.fit(X_train, y_train)
```

Evaluation

[ ]:

```python
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_train,
    y_train,
    "train",
    "Whithout sqft_lot"
)
metrics_df
```

[ ]:

```python
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_test,
    y_test,
    "test",
    "Whithout sqft_lot"
)
metrics_df
```

0.1.4 Conclusion

[ ]:

```python
base_metrics
```

Without this feature the modules don´t change significally so we decided to drop it

0.2 Drop 'floors'

Defining target and features

[ ]:

```python
X = df.drop(['price','floors'], axis=1) # Features
y = df['price'] # Target
X.head()
```

Split the data

[ ]:

```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2,␣
 ↪random_state=13)
```

0.2.1 Linear Regression

[ ]:

```
from sklearn.linear_model import LinearRegression
 # Create the Linear Regression estimator
lm = LinearRegression()

# Perform the fitting
lm.fit(X_train, y_train)
```

Evaluation

[ ]:

```
metrics_df = create_metrics_df()
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_train,
    y_train,
    "train",
    "Whithout floors"
)

metrics_df
```

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_test,
    y_test,
    "test",
    "Whithout floors"
)

metrics_df
```

0.2.2 Random Forest

[ ]:

```
from sklearn.ensemble import RandomForestRegressor
rf_regressor = RandomForestRegressor(random_state=13)

rf_regressor.fit(X_train, y_train)
```

Evaluation

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_train,
    y_train,
    "train",
    "Whithout floors"
)
metrics_df
```

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_test,
    y_test,
    "test",
    "Whithout floors"
)
metrics_df
```

0.2.3 XGBoost

[ ]:

```
import xgboost as xgb

xgb_reg = xgb.XGBRegressor()
xgb_reg.fit(X_train, y_train)
```

Evaluation

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_train,
    y_train,
    "train",
    "Whithout floors"
)
metrics_df
```

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_test,
    y_test,
    "test",
    "Whithout floors"
)
metrics_df
```

0.2.4 Conclusion

[ ]:

```
metrics_df.to_csv('Metrics/drop_floors.csv')
```

[ ]:

```
base_metrics
```

Without this feature the modules don´t change significally so we decided to drop it

0.3 Drop 'month_sold'

Defining target and features

[ ]:

```
X = df.drop(['price','month_sold'], axis=1) # Features
y = df['price'] # Target
X.head()
```

Split the data

[ ]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2,␣
  ↪random_state=13)
```

0.3.1 Linear Regression

[ ]:

```
from sklearn.linear_model import LinearRegression
 # Create the Linear Regression estimator
lm = LinearRegression()

# Perform the fitting
lm.fit(X_train, y_train)
```

Evaluation

[ ]:

```
metrics_df = create_metrics_df()
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_train,
    y_train,
    "train",
    "Whithout month_sold"
)

metrics_df
```

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_test,
    y_test,
    "test",
    "Whithout month_sold"
)

metrics_df
```

0.3.2 Random Forest

[ ]:

```
from sklearn.ensemble import RandomForestRegressor
rf_regressor = RandomForestRegressor(random_state=13)

rf_regressor.fit(X_train, y_train)
```

Evaluation

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_train,
    y_train,
    "train",
    "Whithout month_sold"
)
metrics_df
```

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_test,
    y_test,
    "test",
    "Whithout month_sold"
)
metrics_df
```

0.3.3 XGBoost

[ ]:

```
import xgboost as xgb

xgb_reg = xgb.XGBRegressor()
xgb_reg.fit(X_train, y_train)
```

Evaluation

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_train,
    y_train,
    "train",
    "Whithout month_sold"
)
metrics_df
```

[ ]:

```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_test,
    y_test,
    "test",
    "Whithout month_sold"
)
metrics_df
```

0.3.4 Conclusion

[ ]:

```
metrics_df.to_csv('Metrics/drop_month_sold.csv')
```

[ ]:

```
base_metrics
```

Without this feature the modules don´t change significally so we decided to drop it

0.4 Drop 'day_sold'

Defining target and features

[128]:

```
X = df.drop(['price','day_sold'], axis=1) # Features
y = df['price'] # Target
X.head()
```

[128]:

```
   bedrooms  bathrooms  sqft_living  sqft_lot  floors  waterfront  view  \
0         3       1.00         1180      5650     1.0           0     0
1         3       2.25         2570      7242     2.0           0     0
2         2       1.00          770     10000     1.0           0     0
3         4       3.00         1960      5000     1.0           0     0
4         3       2.00         1680      8080     1.0           0     0

   condition  grade  sqft_above  sqft_basement  yr_built  yr_renovated  \
0          3      7        1180              0      1955             0
1          3      7        2170            400      1951          1991
2          3      6         770              0      1933             0
3          5      7        1050            910      1965             0
4          3      8        1680              0      1987             0

   zipcode      lat     long  sqft_living15  sqft_lot15  year_sold  month_sold
0    98178  47.5112 -122.257           1340        5650       2014          10
1    98125  47.7210 -122.319           1690        7639       2014          12
2    98028  47.7379 -122.233           2720        8062       2015           2
3    98136  47.5208 -122.393           1360        5000       2014          12
4    98074  47.6168 -122.045           1800        7503       2015           2
```

Split the data

[129]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2,␣
 ↪random_state=13)
```

0.4.1 Linear Regression

[130]:
```
from sklearn.linear_model import LinearRegression
 # Create the Linear Regression estimator
lm = LinearRegression()

# Perform the fitting
lm.fit(X_train, y_train)
```

[130]:
```
LinearRegression()
```

Evaluation

[131]:
```
metrics_df = create_metrics_df()
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_train,
    y_train,
    "train",
    "Whithout day_sold"
)

metrics_df
```

```
g:\Mi unidad\IronHack\Project IronKaggel\utils.py:115: FutureWarning: The
behavior of DataFrame concatenation with empty or all-NA entries is deprecated.
In a future version, this will no longer exclude empty or all-NA columns when
determining the result dtypes. To retain the old behavior, exclude the relevant
entries before the concat operation.
  updated_df = pd.concat([metrics_df, new_row_df], ignore_index=True)
```

[131]:
```
              Model  Split      R2  Adjusted_R2          MAE         RMSE  \
0  LinearRegression  train  0.6977       0.6973  125937.4648  202884.7569

    MAPE           Comments
0  0.256  Whithout day_sold
```

[132]:
```
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_test,
    y_test,
    "test",
    "Whithout day_sold"
)

metrics_df
```

```
[132]:                Model  Split      R2  Adjusted_R2          MAE          RMSE  \
       0  LinearRegression  train  0.6977       0.6973  125937.4648  202884.7569
       1  LinearRegression   test  0.7161       0.7148  125973.0357  191559.9896

            MAPE          Comments
       0  0.2560  Whithout day_sold
       1  0.2595  Whithout day_sold
```

0.4.2 Random Forest

```
[133]: from sklearn.ensemble import RandomForestRegressor
       rf_regressor = RandomForestRegressor(random_state=13)

       rf_regressor.fit(X_train, y_train)
```

```
[133]: RandomForestRegressor(random_state=13)
```

Evaluation

```
[134]: metrics_df = add_new_metrics(
           metrics_df,
           rf_regressor,
           X_train,
           y_train,
           "train",
           "Whithout day_sold"
       )
       metrics_df
```

```
[134]:                     Model  Split      R2  Adjusted_R2          MAE  \
       0       LinearRegression  train  0.6977       0.6973  125937.4648
       1       LinearRegression   test  0.7161       0.7148  125973.0357
       2  RandomForestRegressor  train  0.9824       0.9824   25800.6256

                 RMSE    MAPE           Comments
       0  202884.7569  0.2560  Whithout day_sold
       1  191559.9896  0.2595  Whithout day_sold
       2   48907.9896  0.0483  Whithout day_sold
```

```
[135]: metrics_df = add_new_metrics(
           metrics_df,
           rf_regressor,
           X_test,
           y_test,
           "test",
           "Whithout day_sold"
       )
       metrics_df
```

[135]:

```
                   Model  Split      R2  Adjusted_R2          MAE  \
0       LinearRegression  train  0.6977       0.6973  125937.4648   
1       LinearRegression   test  0.7161       0.7148  125973.0357   
2  RandomForestRegressor  train  0.9824       0.9824   25800.6256   
3  RandomForestRegressor   test  0.8959       0.8954   67286.1944   

          RMSE    MAPE           Comments  
0  202884.7569  0.2560  Whithout day_sold  
1  191559.9896  0.2595  Whithout day_sold  
2   48907.9896  0.0483  Whithout day_sold  
3  115992.8695  0.1269  Whithout day_sold  
```

0.4.3 XGBoost

[136]:

```
import xgboost as xgb

xgb_reg = xgb.XGBRegressor()
xgb_reg.fit(X_train, y_train)
```

[136]:

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             feature_weights=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=None, num_parallel_tree=None, …)
```

Evaluation

[137]:

```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_train,
    y_train,
    "train",
    "Whithout day_sold"
)
metrics_df
```

```
[137]:                    Model  Split      R2  Adjusted_R2          MAE  \
       0       LinearRegression  train  0.6977       0.6973  125937.4648
       1       LinearRegression   test  0.7161       0.7148  125973.0357
       2  RandomForestRegressor  train  0.9824       0.9824   25800.6256
       3  RandomForestRegressor   test  0.8959       0.8954   67286.1944
       4           XGBRegressor  train  0.9783       0.9782   39015.3232

                RMSE    MAPE           Comments
       0  202884.7569  0.2560  Whithout day_sold
       1  191559.9896  0.2595  Whithout day_sold
       2   48907.9896  0.0483  Whithout day_sold
       3  115992.8695  0.1269  Whithout day_sold
       4   54398.0531  0.0871  Whithout day_sold
```

```
[138]: metrics_df = add_new_metrics(
           metrics_df,
           xgb_reg,
           X_test,
           y_test,
           "test",
           "Whithout day_sold"
       )
       metrics_df
```

```
[138]:                    Model  Split      R2  Adjusted_R2          MAE  \
       0       LinearRegression  train  0.6977       0.6973  125937.4648
       1       LinearRegression   test  0.7161       0.7148  125973.0357
       2  RandomForestRegressor  train  0.9824       0.9824   25800.6256
       3  RandomForestRegressor   test  0.8959       0.8954   67286.1944
       4           XGBRegressor  train  0.9783       0.9782   39015.3232
       5           XGBRegressor   test  0.9060       0.9056   64958.8204

                RMSE    MAPE           Comments
       0  202884.7569  0.2560  Whithout day_sold
       1  191559.9896  0.2595  Whithout day_sold
       2   48907.9896  0.0483  Whithout day_sold
       3  115992.8695  0.1269  Whithout day_sold
       4   54398.0531  0.0871  Whithout day_sold
       5  110209.5275  0.1238  Whithout day_sold
```

0.4.4 Conclusion

```
[139]: metrics_df.to_csv('Metrics/drop_day_sold.csv')
```

```
[144]: base_metrics
```

```
[144]:                     Model  Split      R2  Adjusted_R2          MAE  \
        0       LinearRegression  train  0.6977       0.6974  125948.1118
        1       LinearRegression   test  0.7162       0.7148  125985.6747
        4  RandomForestRegressor  train  0.9820       0.9819   25948.4444
        5  RandomForestRegressor   test  0.8959       0.8954   67269.8770
        6           XGBRegressor  train  0.9780       0.9780   39126.7558
        7           XGBRegressor   test  0.9015       0.9010   65712.7448

                  RMSE    MAPE                                       Comments
        0  202864.5703  0.2561                                 Baseline model
        1  191531.3335  0.2596                                 Baseline model
        4   49547.1197  0.0486  Baseline, no normalization, random_state 13, d…
        5  115994.2051  0.1271  Baseline, no normalization, random_state 13, d…
        6   54676.7727  0.0872        Baseline, no normalization, default values.
        7  112860.4995  0.1246        Baseline, no normalization, default values.
```

0.5 Drop 'yr_renovated'

Defining target and features

```
[145]: X = df.drop(['price','yr_renovated'], axis=1) # Features
       y = df['price'] # Target
       X.head()
```

```
[145]:    bedrooms  bathrooms  sqft_living  sqft_lot  floors  waterfront  view  \
        0         3       1.00         1180      5650     1.0           0     0
        1         3       2.25         2570      7242     2.0           0     0
        2         2       1.00          770     10000     1.0           0     0
        3         4       3.00         1960      5000     1.0           0     0
        4         3       2.00         1680      8080     1.0           0     0

           condition  grade  sqft_above  sqft_basement  yr_built  zipcode      lat  \
        0          3      7        1180              0      1955    98178  47.5112
        1          3      7        2170            400      1951    98125  47.7210
        2          3      6         770              0      1933    98028  47.7379
        3          5      7        1050            910      1965    98136  47.5208
        4          3      8        1680              0      1987    98074  47.6168

              long  sqft_living15  sqft_lot15  year_sold  month_sold  day_sold
        0 -122.257           1340        5650       2014          10        13
        1 -122.319           1690        7639       2014          12         9
        2 -122.233           2720        8062       2015           2        25
        3 -122.393           1360        5000       2014          12         9
        4 -122.045           1800        7503       2015           2        18
```

Split the data

[146]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2,␣
 ↪random_state=13)
```

0.5.1 Linear Regression

[147]:

```
from sklearn.linear_model import LinearRegression
 # Create the Linear Regression estimator
lm = LinearRegression()

# Perform the fitting
lm.fit(X_train, y_train)
```

[147]:

```
LinearRegression()
```

Evaluation

[148]:

```
metrics_df = create_metrics_df()
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_train,
    y_train,
    "train",
    "Whithout yr_renovated"
)

metrics_df
```

```
g:\Mi unidad\IronHack\Project IronKaggel\utils.py:115: FutureWarning: The
behavior of DataFrame concatenation with empty or all-NA entries is deprecated.
In a future version, this will no longer exclude empty or all-NA columns when
determining the result dtypes. To retain the old behavior, exclude the relevant
entries before the concat operation.
  updated_df = pd.concat([metrics_df, new_row_df], ignore_index=True)
```

[148]:

```
              Model  Split      R2  Adjusted_R2         MAE         RMSE  \
0  LinearRegression  train  0.6973       0.6969  126097.584  203019.8769

     MAPE               Comments
0  0.2563  Whithout yr_renovated
```

[149]:

```
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_test,
    y_test,
    "test",
    "Whithout yr_renovated"
)

metrics_df
```

[149]:

```
               Model  Split      R2  Adjusted_R2         MAE         RMSE  \
0  LinearRegression  train  0.6973       0.6969  126097.584  203019.8769
1  LinearRegression   test  0.7158       0.7144  126123.658  191673.0385

     MAPE               Comments
0  0.2563  Whithout yr_renovated
1  0.2600  Whithout yr_renovated
```

0.5.2 Random Forest

[150]:

```
from sklearn.ensemble import RandomForestRegressor
rf_regressor = RandomForestRegressor(random_state=13)

rf_regressor.fit(X_train, y_train)
```

[150]:

```
RandomForestRegressor(random_state=13)
```

Evaluation

[151]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_train,
    y_train,
    "train",
    "Whithout yr_renovated"
)
metrics_df
```

[151]:

```
                    Model  Split      R2  Adjusted_R2         MAE         RMSE  \
0       LinearRegression  train  0.6973       0.6969  126097.584  203019.8769
1       LinearRegression   test  0.7158       0.7144  126123.658  191673.0385
2  RandomForestRegressor  train  0.9819       0.9819   25962.452   49633.6552

     MAPE               Comments
0  0.2563  Whithout yr_renovated
1  0.2600  Whithout yr_renovated
2  0.0487  Whithout yr_renovated
```

[152]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_test,
    y_test,
    "test",
    "Whithout yr_renovated"
)
metrics_df
```

[152]:

```
                   Model  Split      R2  Adjusted_R2          MAE  \
0       LinearRegression  train  0.6973       0.6969  126097.5840
1       LinearRegression   test  0.7158       0.7144  126123.6580
2  RandomForestRegressor  train  0.9819       0.9819   25962.4520
3  RandomForestRegressor   test  0.8958       0.8953   67255.3388

          RMSE    MAPE               Comments
0  203019.8769  0.2563  Whithout yr_renovated
1  191673.0385  0.2600  Whithout yr_renovated
2   49633.6552  0.0487  Whithout yr_renovated
3  116066.5787  0.1272  Whithout yr_renovated
```

0.5.3 XGBoost

[153]:

```
import xgboost as xgb

xgb_reg = xgb.XGBRegressor()
xgb_reg.fit(X_train, y_train)
```

[153]:

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             feature_weights=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=None, num_parallel_tree=None, …)
```

Evaluation

[154]:

```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_train,
    y_train,
    "train",
    "Whithout yr_renovated"
)
metrics_df
```

[154]:
```
                     Model  Split      R2  Adjusted_R2          MAE  \
0         LinearRegression  train  0.6973       0.6969  126097.5840
1         LinearRegression   test  0.7158       0.7144  126123.6580
2    RandomForestRegressor  train  0.9819       0.9819   25962.4520
3    RandomForestRegressor   test  0.8958       0.8953   67255.3388
4             XGBRegressor  train  0.9774       0.9774   39451.3132

          RMSE    MAPE               Comments
0  203019.8769  0.2563  Whithout yr_renovated
1  191673.0385  0.2600  Whithout yr_renovated
2   49633.6552  0.0487  Whithout yr_renovated
3  116066.5787  0.1272  Whithout yr_renovated
4   55430.4617  0.0869  Whithout yr_renovated
```

[155]:
```python
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_test,
    y_test,
    "test",
    "Whithout yr_renovated"
)
metrics_df
```

[155]:
```
                     Model  Split      R2  Adjusted_R2          MAE  \
0         LinearRegression  train  0.6973       0.6969  126097.5840
1         LinearRegression   test  0.7158       0.7144  126123.6580
2    RandomForestRegressor  train  0.9819       0.9819   25962.4520
3    RandomForestRegressor   test  0.8958       0.8953   67255.3388
4             XGBRegressor  train  0.9774       0.9774   39451.3132
5             XGBRegressor   test  0.8991       0.8986   66615.4646

          RMSE    MAPE               Comments
0  203019.8769  0.2563  Whithout yr_renovated
1  191673.0385  0.2600  Whithout yr_renovated
2   49633.6552  0.0487  Whithout yr_renovated
3  116066.5787  0.1272  Whithout yr_renovated
4   55430.4617  0.0869  Whithout yr_renovated
5  114219.5770  0.1262  Whithout yr_renovated
```

0.5.4 Conclusion

[ ]:
```python
metrics_df.to_csv('Metrics/drop_day_sold.csv')
```

[ ]:
```python
base_metrics
```

```
                     Model  Split      R2  Adjusted_R2          MAE  \
0         LinearRegression  train  0.6977       0.6974  125948.1118
```

```
1       LinearRegression   test  0.7162       0.7148  125985.6747
4  RandomForestRegressor  train  0.9820       0.9819   25948.4444
5  RandomForestRegressor   test  0.8959       0.8954   67269.8770
6           XGBRegressor  train  0.9780       0.9780   39126.7558
7           XGBRegressor   test  0.9015       0.9010   65712.7448

          RMSE    MAPE                                          Comments
0  202864.5703  0.2561                                    Baseline model
1  191531.3335  0.2596                                    Baseline model
4   49547.1197  0.0486  Baseline, no normalization, random_state 13, d…
5  115994.2051  0.1271  Baseline, no normalization, random_state 13, d…
6   54676.7727  0.0872        Baseline, no normalization, default values.
7  112860.4995  0.1246        Baseline, no normalization, default values.
```

It makes it slightly worse

0.6 Drop 'bedrooms'

Defining target and features

[5]:
```
X = df.drop(['price','bedrooms'], axis=1) # Features
y = df['price'] # Target
X.head()
```

[5]:
```
   bathrooms  sqft_living  sqft_lot  floors  waterfront  view  condition  \
0       1.00         1180      5650     1.0           0     0          3
1       2.25         2570      7242     2.0           0     0          3
2       1.00          770     10000     1.0           0     0          3
3       3.00         1960      5000     1.0           0     0          5
4       2.00         1680      8080     1.0           0     0          3

   grade  sqft_above  sqft_basement  yr_built  yr_renovated  zipcode      lat  \
0      7        1180              0      1955             0    98178  47.5112
1      7        2170            400      1951          1991    98125  47.7210
2      6         770              0      1933             0    98028  47.7379
3      7        1050            910      1965             0    98136  47.5208
4      8        1680              0      1987             0    98074  47.6168

      long  sqft_living15  sqft_lot15  year_sold  month_sold  day_sold
0 -122.257           1340        5650       2014          10        13
1 -122.319           1690        7639       2014          12         9
2 -122.233           2720        8062       2015           2        25
3 -122.393           1360        5000       2014          12         9
4 -122.045           1800        7503       2015           2        18
```

Split the data

[6]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2,␣
 ↪random_state=13)
```

0.6.1 Random Forest

[8]:

```
from sklearn.ensemble import RandomForestRegressor
rf_regressor = RandomForestRegressor(random_state=13)

rf_regressor.fit(X_train, y_train)
```

[8]:

```
RandomForestRegressor(random_state=13)
```

Evaluation

[9]:

```
metrics_df = create_metrics_df()
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_train,
    y_train,
    "train",
    "Whithout bedrooms"
)
metrics_df
```

```
g:\Mi unidad\IronHack\Project IronKaggel\utils.py:115: FutureWarning: The
behavior of DataFrame concatenation with empty or all-NA entries is deprecated.
In a future version, this will no longer exclude empty or all-NA columns when
determining the result dtypes. To retain the old behavior, exclude the relevant
entries before the concat operation.
  updated_df = pd.concat([metrics_df, new_row_df], ignore_index=True)
```

[9]:

```
                   Model  Split      R2  Adjusted_R2         MAE        RMSE  \
0  RandomForestRegressor  train  0.9823       0.9822  25888.0036  49132.8562

     MAPE           Comments
0  0.0486  Whithout bedrooms
```

[10]:

```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_test,
    y_test,
    "test",
    "Whithout yr_renovated"
)
metrics_df
```

[10]:

```
                   Model  Split      R2  Adjusted_R2         MAE         RMSE  \
0  RandomForestRegressor  train  0.9823       0.9822  25888.0036   49132.8562
1  RandomForestRegressor   test  0.8969       0.8964  67042.6859  115463.7677
```

```
     MAPE               Comments
0  0.0486       Whithout bedrooms
1  0.1270  Whithout yr_renovated
```

0.6.2 XGBoost

[11]:
```
import xgboost as xgb

xgb_reg = xgb.XGBRegressor()
xgb_reg.fit(X_train, y_train)
```

[11]:
```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             feature_weights=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=None, num_parallel_tree=None, …)
```

Evaluation

[12]:
```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_train,
    y_train,
    "train",
    "Whithout yr_renovated"
)
metrics_df
```

[12]:
```
                   Model  Split      R2  Adjusted_R2         MAE         RMSE  \
0  RandomForestRegressor  train  0.9823       0.9822  25888.0036   49132.8562
1  RandomForestRegressor   test  0.8969       0.8964  67042.6859  115463.7677
2           XGBRegressor  train  0.9786       0.9786  38678.5863   54008.8395

     MAPE               Comments
0  0.0486       Whithout bedrooms
1  0.1270  Whithout yr_renovated
2  0.0860  Whithout yr_renovated
```

[13]:
```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_test,
    y_test,
    "test",
    "Whithout yr_renovated"
)
metrics_df
```

[13]:

```
                   Model  Split      R2  Adjusted_R2         MAE         RMSE  \
0  RandomForestRegressor  train  0.9823       0.9822  25888.0036   49132.8562
1  RandomForestRegressor   test  0.8969       0.8964  67042.6859  115463.7677
2           XGBRegressor  train  0.9786       0.9786  38678.5863   54008.8395
3           XGBRegressor   test  0.9003       0.8999  65793.1286  113502.9973

     MAPE               Comments
0  0.0486      Whithout bedrooms
1  0.1270  Whithout yr_renovated
2  0.0860  Whithout yr_renovated
3  0.1241  Whithout yr_renovated
```

0.6.3 Conclusion

[156]:

```
metrics_df.to_csv('Metrics/drop_yr_renovated.csv')
```

[ ]:

```
base_metrics
```

```
                   Model  Split      R2  Adjusted_R2          MAE  \
0       LinearRegression  train  0.6977       0.6974  125948.1118
1       LinearRegression   test  0.7162       0.7148  125985.6747
4  RandomForestRegressor  train  0.9820       0.9819   25948.4444
5  RandomForestRegressor   test  0.8959       0.8954   67269.8770
6           XGBRegressor  train  0.9780       0.9780   39126.7558
7           XGBRegressor   test  0.9015       0.9010   65712.7448

          RMSE    MAPE                                         Comments
0  202864.5703  0.2561                                   Baseline model
1  191531.3335  0.2596                                   Baseline model
4   49547.1197  0.0486  Baseline, no normalization, random_state 13, d…
5  115994.2051  0.1271  Baseline, no normalization, random_state 13, d…
6   54676.7727  0.0872      Baseline, no normalization, default values.
7  112860.4995  0.1246      Baseline, no normalization, default values.
```

It improves a lot

0.7 Drop ‘sqft_lot’, ‘floors’, ‘month_sold’ and ‘day_sold’ toguether

Defining target and features

[157]:

```
X = df.drop(['price', 'sqft_lot', 'floors', 'month_sold','day_sold'], axis=1) #␣
 ↪Features
y = df['price'] # Target
X.head()
```

[157]:

```
   bedrooms  bathrooms  sqft_living  waterfront  view  condition  grade  \
0         3       1.00         1180           0     0          3      7
1         3       2.25         2570           0     0          3      7
2         2       1.00          770           0     0          3      6
3         4       3.00         1960           0     0          5      7
4         3       2.00         1680           0     0          3      8

   sqft_above  sqft_basement  yr_built  yr_renovated  zipcode      lat  \
0        1180              0      1955             0    98178  47.5112
1        2170            400      1951          1991    98125  47.7210
2         770              0      1933             0    98028  47.7379
3        1050            910      1965             0    98136  47.5208
4        1680              0      1987             0    98074  47.6168

      long  sqft_living15  sqft_lot15  year_sold
0 -122.257           1340        5650       2014
1 -122.319           1690        7639       2014
2 -122.233           2720        8062       2015
3 -122.393           1360        5000       2014
4 -122.045           1800        7503       2015
```

Split the data

[158]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.2,␣
 ↪random_state=13)
```

0.7.1 Linear Regression

[159]:

```
from sklearn.linear_model import LinearRegression
 # Create the Linear Regression estimator
lm = LinearRegression()

# Perform the fitting
lm.fit(X_train, y_train)
```

[159]:

```
LinearRegression()
```

Evaluation

[160]:

```
metrics_df = create_metrics_df()
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_train,
    y_train,
    "train",
    "Whithout 4 columns"
)

metrics_df
```

```
g:\Mi unidad\IronHack\Project IronKaggel\utils.py:115: FutureWarning: The
behavior of DataFrame concatenation with empty or all-NA entries is deprecated.
In a future version, this will no longer exclude empty or all-NA columns when
determining the result dtypes. To retain the old behavior, exclude the relevant
entries before the concat operation.
  updated_df = pd.concat([metrics_df, new_row_df], ignore_index=True)
```

[160]:

```
              Model  Split      R2  Adjusted_R2          MAE        RMSE  \
0  LinearRegression  train  0.6976       0.6973  126045.8997  202924.743

     MAPE            Comments
0  0.2561  Whithout 4 columns
```

[161]:

```
metrics_df = add_new_metrics(
    metrics_df,
    lm,
    X_test,
    y_test,
    "test",
    "Whithout 4 columns"
)

metrics_df
```

[161]:

```
              Model  Split      R2  Adjusted_R2          MAE         RMSE  \
0  LinearRegression  train  0.6976       0.6973  126045.8997  202924.7430
1  LinearRegression   test  0.7157       0.7145  126103.8682  191706.7474

     MAPE            Comments
0  0.2561  Whithout 4 columns
1  0.2597  Whithout 4 columns
```

0.7.2 Random Forest

[162]:

```
from sklearn.ensemble import RandomForestRegressor
rf_regressor = RandomForestRegressor(random_state=13)

rf_regressor.fit(X_train, y_train)
```

[162]:

```
RandomForestRegressor(random_state=13)
```

Evaluation

[163]:
```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_train,
    y_train,
    "train",
    "Whithout 4 columns"
)
metrics_df
```

[163]:
```
                   Model  Split      R2  Adjusted_R2          MAE  \
0       LinearRegression  train  0.6976       0.6973  126045.8997
1       LinearRegression   test  0.7157       0.7145  126103.8682
2  RandomForestRegressor  train  0.9823       0.9823   25887.6388

          RMSE    MAPE            Comments
0  202924.7430  0.2561  Whithout 4 columns
1  191706.7474  0.2597  Whithout 4 columns
2   49105.0748  0.0486  Whithout 4 columns
```

[164]:
```
metrics_df = add_new_metrics(
    metrics_df,
    rf_regressor,
    X_test,
    y_test,
    "test",
    "Whithout 4 columns"
)
metrics_df
```

[164]:
```
                   Model  Split      R2  Adjusted_R2          MAE  \
0       LinearRegression  train  0.6976       0.6973  126045.8997
1       LinearRegression   test  0.7157       0.7145  126103.8682
2  RandomForestRegressor  train  0.9823       0.9823   25887.6388
3  RandomForestRegressor   test  0.8975       0.8971   67388.4881

          RMSE    MAPE            Comments
0  202924.7430  0.2561  Whithout 4 columns
1  191706.7474  0.2597  Whithout 4 columns
2   49105.0748  0.0486  Whithout 4 columns
3  115083.0470  0.1276  Whithout 4 columns
```

0.7.3 XGBoost

[165]:
```
import xgboost as xgb

xgb_reg = xgb.XGBRegressor()
xgb_reg.fit(X_train, y_train)
```

[165]:
```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             feature_weights=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=None, num_parallel_tree=None, …)
```

Evaluation

[166]:
```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_train,
    y_train,
    "train",
    "Whithout 4 columns"
)
metrics_df
```

[166]:
```
                   Model  Split      R2  Adjusted_R2          MAE  \
0       LinearRegression  train  0.6976       0.6973  126045.8997
1       LinearRegression   test  0.7157       0.7145  126103.8682
2  RandomForestRegressor  train  0.9823       0.9823   25887.6388
3  RandomForestRegressor   test  0.8975       0.8971   67388.4881
4           XGBRegressor  train  0.9756       0.9756   40967.3963

          RMSE    MAPE            Comments
0  202924.7430  0.2561  Whithout 4 columns
1  191706.7474  0.2597  Whithout 4 columns
2   49105.0748  0.0486  Whithout 4 columns
3  115083.0470  0.1276  Whithout 4 columns
4   57584.1626  0.0907  Whithout 4 columns
```

[167]:
```
metrics_df = add_new_metrics(
    metrics_df,
    xgb_reg,
    X_test,
    y_test,
    "test",
    "Whithout 4 columns"
)
metrics_df
```

[167]:

```
                   Model  Split      R2  Adjusted_R2          MAE  \
0       LinearRegression  train  0.6976       0.6973  126045.8997
1       LinearRegression   test  0.7157       0.7145  126103.8682
2  RandomForestRegressor  train  0.9823       0.9823   25887.6388
3  RandomForestRegressor   test  0.8975       0.8971   67388.4881
4           XGBRegressor  train  0.9756       0.9756   40967.3963
5           XGBRegressor   test  0.9008       0.9004   67442.8650

         RMSE    MAPE            Comments
0  202924.7430  0.2561  Whithout 4 columns
1  191706.7474  0.2597  Whithout 4 columns
2   49105.0748  0.0486  Whithout 4 columns
3  115083.0470  0.1276  Whithout 4 columns
4   57584.1626  0.0907  Whithout 4 columns
5  113232.2624  0.1288  Whithout 4 columns
```

0.7.4 Conclusion

[168]:

```
metrics_df.to_csv('Metrics/drop_4_columns.csv')
```

[169]:

```
base_metrics
```

[169]:

```
                   Model  Split      R2  Adjusted_R2          MAE  \
0       LinearRegression  train  0.6977       0.6974  125948.1118
1       LinearRegression   test  0.7162       0.7148  125985.6747
4  RandomForestRegressor  train  0.9820       0.9819   25948.4444
5  RandomForestRegressor   test  0.8959       0.8954   67269.8770
6           XGBRegressor  train  0.9780       0.9780   39126.7558
7           XGBRegressor   test  0.9015       0.9010   65712.7448

         RMSE    MAPE                                   Comments
0  202864.5703  0.2561                             Baseline model
1  191531.3335  0.2596                             Baseline model
4   49547.1197  0.0486  Baseline, no normalization, random_state 13, d…
5  115994.2051  0.1271  Baseline, no normalization, random_state 13, d…
6   54676.7727  0.0872        Baseline, no normalization, default values.
7  112860.4995  0.1246        Baseline, no normalization, default values.
```

Dropping this columns improves slightly in general the scores of Linear Regression and XGB but not Random Forest