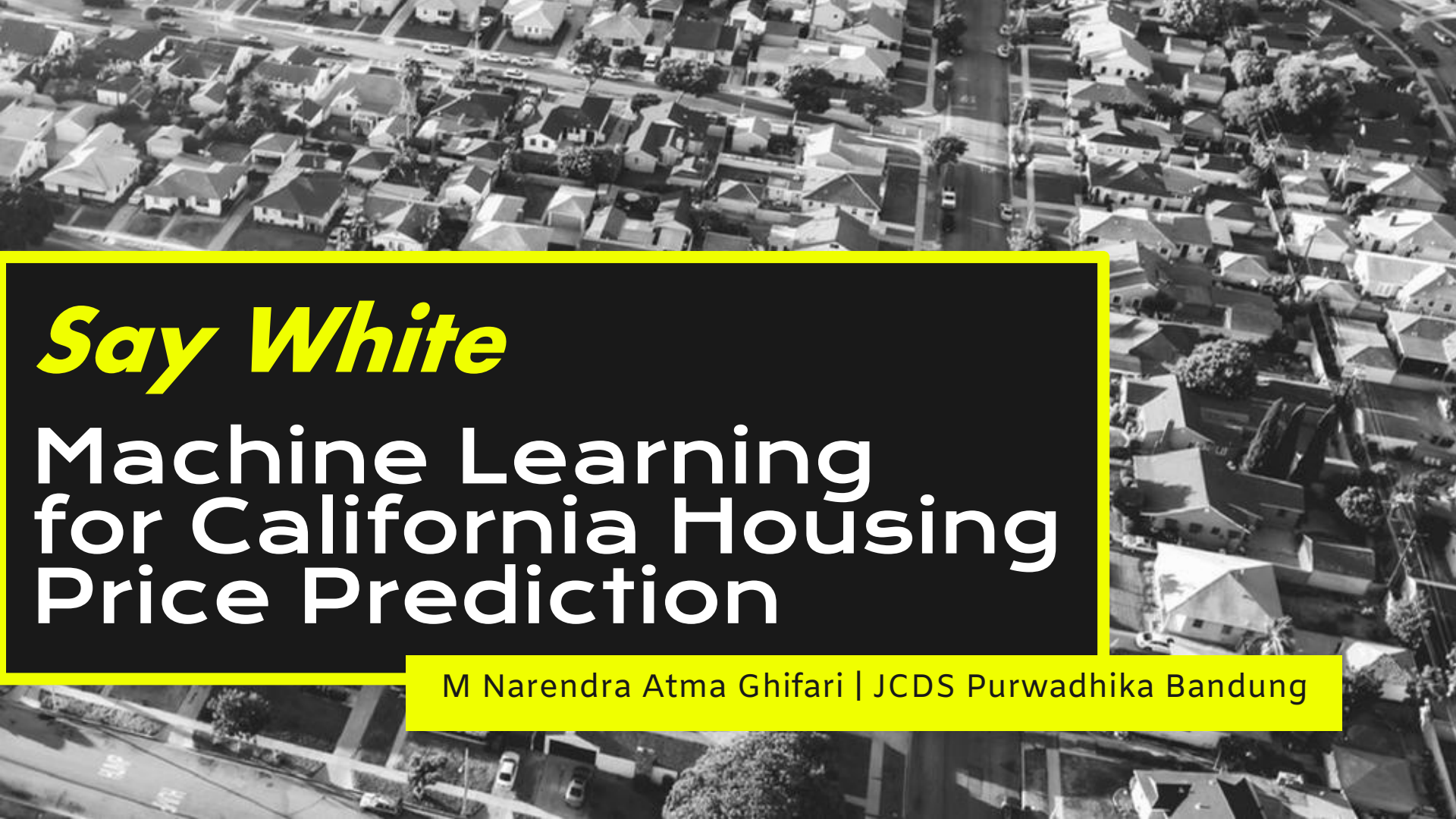


Say White

Machine Learning for California Housing Price Prediction

M Narendra Atma Ghifari | JCDS Purwadhika Bandung

Table of contents

01

Business Overview

About
Say White

Perusahaan properti berbasis di California yang bergerak dalam layanan jual, beli, dan sewa properti residensial maupun komersial.

Say White berkomitmen untuk membantu klien mendapatkan properti terbaik dengan harga transparan, sesuai nilai pasar, dan menguntungkan kedua belah pihak.

Say White

Business Problem Understanding

California merupakan wilayah dengan pasar properti yang dinamis dan kompleks, dimana harga rumah di wilayah ini dipengaruhi oleh berbagai faktor.

Sebagai perusahaan properti yang terus berkembang, Say White perlu memiliki strategi penetapan harga yang akurat dan berbasis data. Namun proses estimasi harga masih sering dilakukan secara manual yang sudah tidak relevan karena memakan waktu dan hasilnya terkadang tidak akurat.

Say White membutuhkan sistem prediksi harga berbasis *machine learning* yang dapat menentukan harga properti secara optimal, kompetitif, dan sesuai kondisi pasar.

Say White

Goals

- Mengembangkan model machine learning yang dapat memprediksi harga properti secara akurat berdasarkan faktor-faktor yang mempengaruhi.
- Mendukung strategi bisnis Say White dalam menetapkan harga properti yang optimal, kompetitif, dan sesuai nilai pasar, guna mempercepat transaksi serta menjaga transparansi dan keuntungan bagi semua pihak.

Say White

Stakeholder

Pihak-pihak yang berperan dan akan terdampak dalam implementasi project ini:

- **Manajemen Say White:** Menentukan strategi pricing dan keputusan bisnis.
- **Tim Data & Analytics:** Mengembangkan dan mengoptimasi model prediksi harga.
- **Tim Marketing & Sales:** Menggunakan model untuk menetapkan harga properti di pasar.
- **Pemilik Properti (Klien):** Mendapatkan harga jual yang optimal sesuai pasar.
- **Calon Pembeli:** Mendapatkan harga properti yang adil dan sebanding nilainya.

02

Data Analysis & Preprocessing

Say White

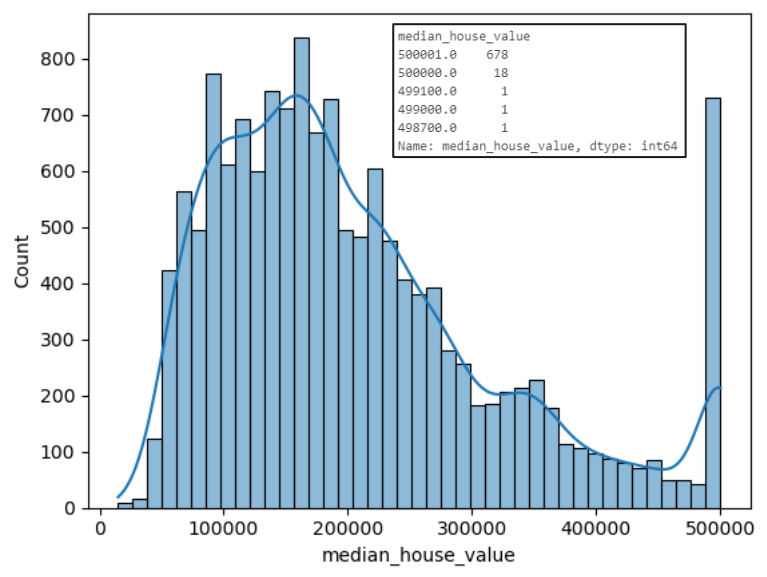
Dataset Understanding

- Dataset yang digunakan merupakan data tentang harga properti di California berdasarkan berbagai fitur yang memengaruhinya.
- Dataset terdiri dari 14448 baris data dan 10 kolom (9 kolom numerikal & 1 kolom kategorikal). Berikut informasi detail dari tiap kolom dataset:

Attribute	Data Type	Description
longitude	Float	Garis bujur dari lokasi properti.
latitude	Float	Garis lintang dari lokasi properti.
housing_median_age	Float	Usia median dari rumah-rumah di area tersebut.
total_rooms	Integer	Total jumlah kamar yang ada di dalam rumah atau properti tersebut.
total_bedrooms	Integer	Total jumlah kamar tidur yang ada dalam rumah tersebut.
population	Integer	Jumlah penduduk yang tinggal di sekitar lokasi rumah tersebut.
households	Integer	Jumlah rumah tangga yang ada di area tersebut.
median_income	Float	Pendapatan median per kapita di area tersebut.
ocean_proximity	Object	Kategori kedekatan rumah dengan lautan (misalnya: INLAND, NEAR BAY, <1H OCEAN).
median_house_value	Integer	Median harga rumah median dalam dolar untuk properti yang ada di area tersebut.

Say White

Distribusi Kolom Target



- Kolom median_house_value memiliki distribusi right-skewed.
- Terdapat lonjakan nilai di $500,001 sebanyak 678 data (4,69%).
- Nilai tersebut dianggap sebagai *outlier* karena tidak spesifik dan proporsinya kecil.
- Data ini dihapus agar distribusi lebih normal dan model tetap konsisten dalam menangkap pola mayoritas harga rumah.

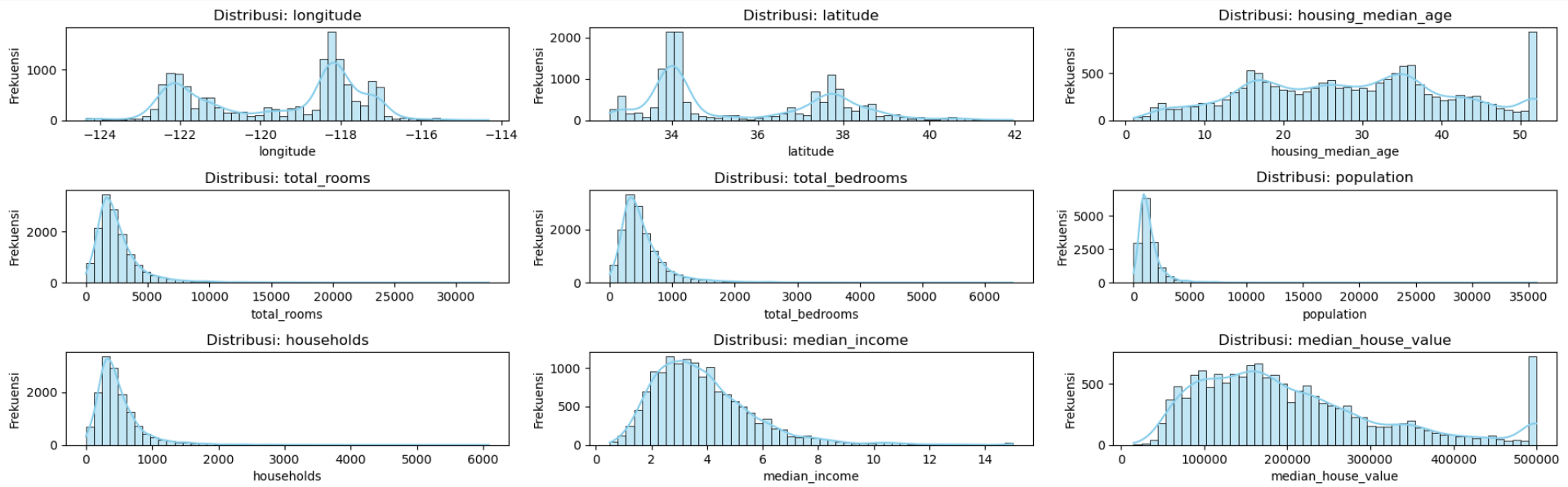
```
# Drop data yang memiliki nilai median_house_value 500001
df = df.drop(df[df['median_house_value']==500001.0].index)
```

Say White

Distribusi Kolom Numerik

Sebagian besar data numerik memiliki:

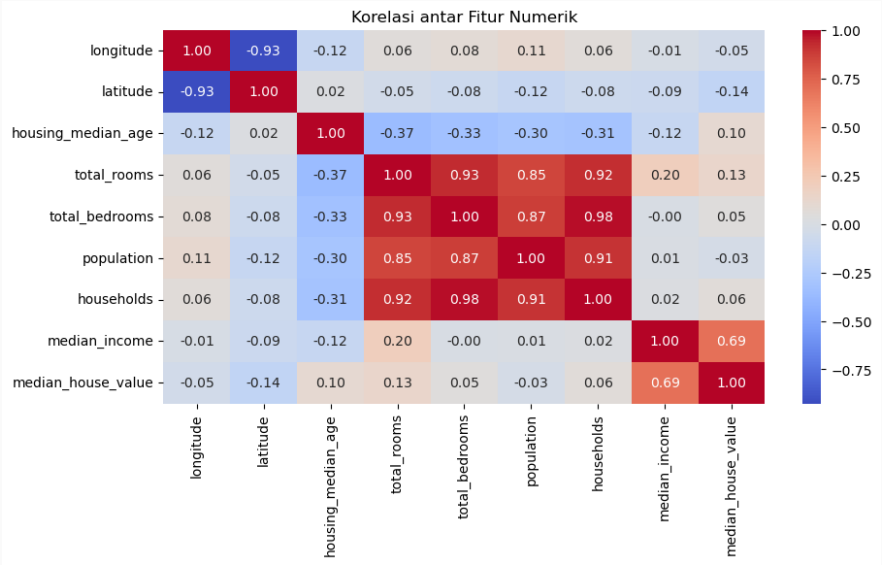
- Diistribusi data yang tidak normal dan cenderung *right-skewed.*
- Perlu dilakukan *scaling* agar model lebih stabil dan akurat.

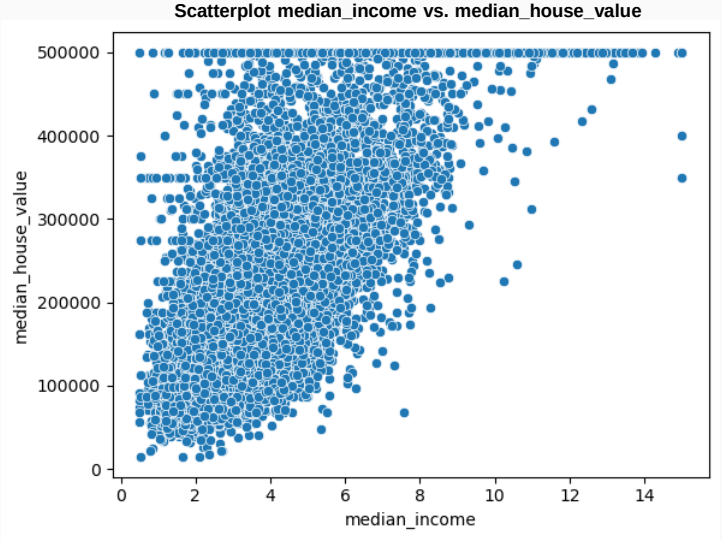


Say White

Korelasi Kolom Numerik

- Kolom median_income memiliki korelasi paling kuat terhadap median_house_value (0.69).
- Artinya, semakin tinggi pendapatan median di suatu area, cenderung semakin tinggi nilai median harga rumah di area tersebut.

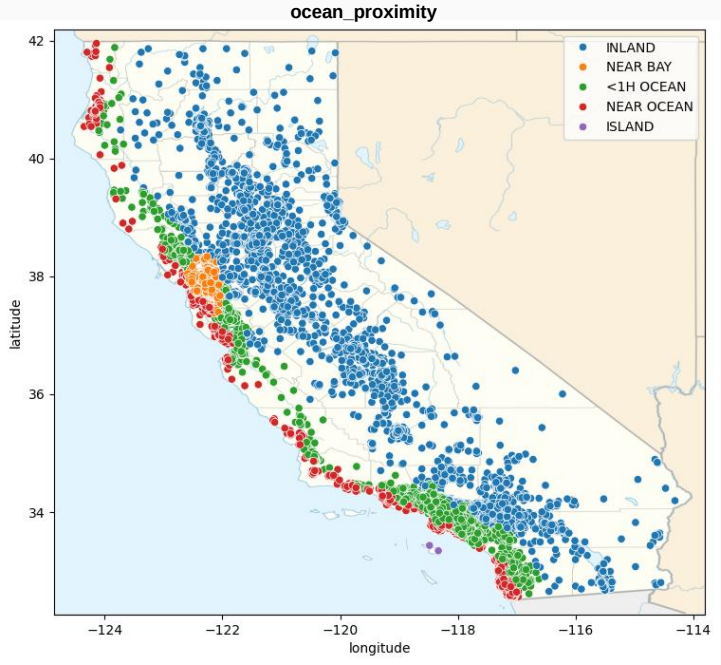


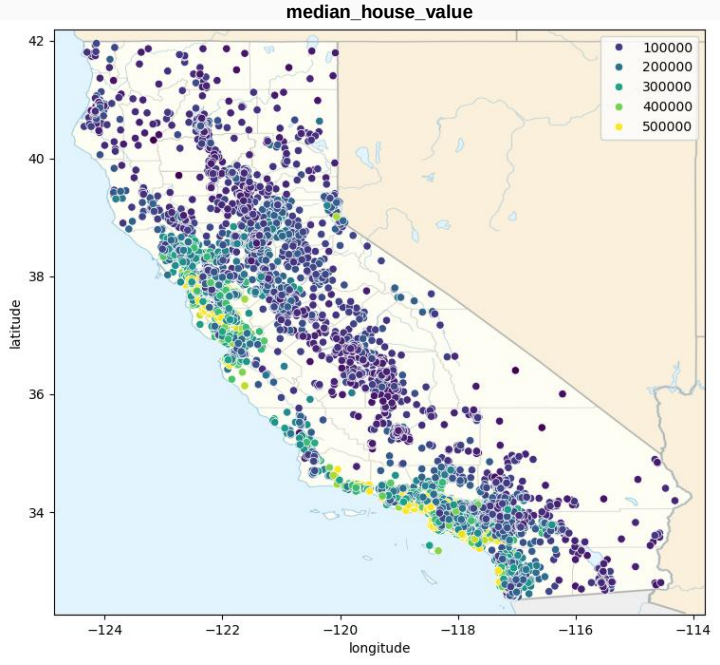


Say White

Analisis Lokasi terhadap Harga Properti

- Berdasarkan scatterplot antara lokasi dan harga properti, dapat disimpulkan bahwa harga properti dipengaruhi oleh lokasinya.
- Semakin dekat dengan pesisir maka harga properti cenderung akan semakin tinggi.





Say White

Preprocessing: Duplicate & Missing Value

```
df.isna().sum()
```

```
longitude             0
latitude              0
housing_median_age    0
total_rooms           0
total_bedrooms      137
population            0
households            0
median_income         0
ocean_proximity       0
median_house_value    0
dtype: int64
```

```
df.duplicated().sum()
```

```
0
```

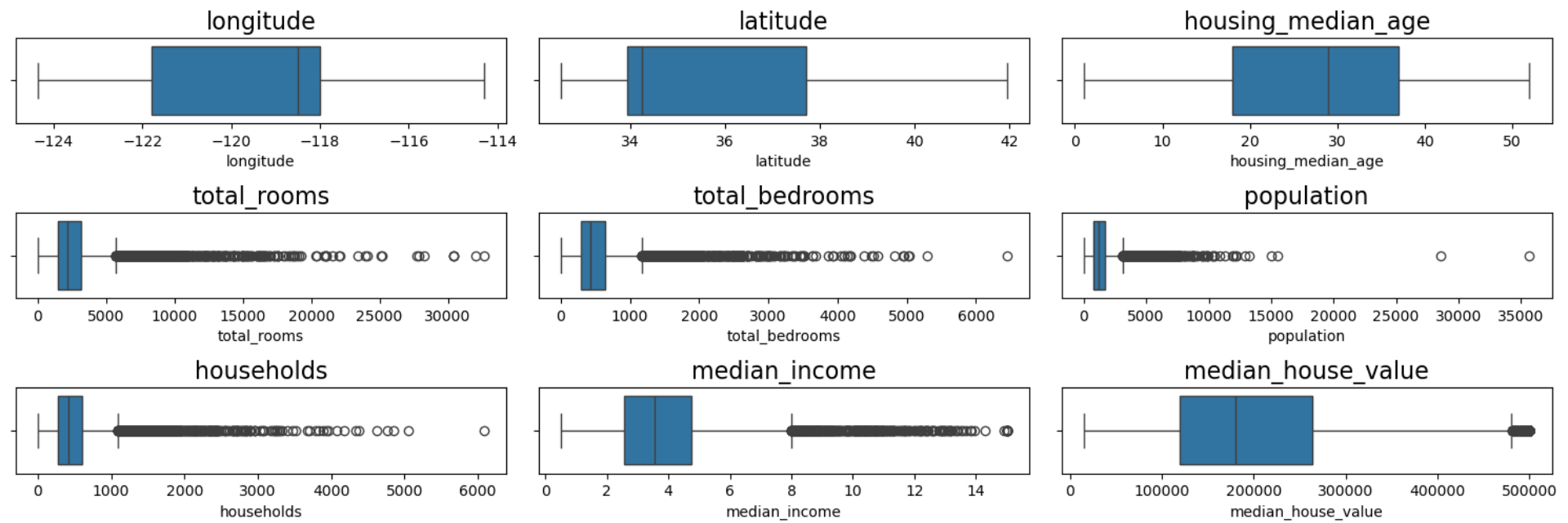
```
# Mengisi missing value dengan median
df['total_bedrooms'].fillna(df['total_bedrooms'].median(), inplace=True)
```

- Tidak ditemukan adanya data *duplicate*.
- Terdapat 137 *missing value* pada kolom total_bedrooms.
- Handling *missing value* dilakukan dengan median secara langsung tanpa SimpleImputer, karena:
  - Hanya terdapat 1 kolom *missing value dengan* jumlah relatif kecil.
  - Lebih cepat dan efisien.
  - Tidak perlu membuat pipeline preprocessing khusus.
  - Tetap sesuai *best practice* untuk kasus sederhana.

Say White

Preprocessing: Handling Outliers

- Sebagian besar outlier masih relevan dengan kondisi data dan tetap dipertahankan.
- Outlier dengan nilai sangat tinggi dihapus karena tidak mewakili kondisi umum, berpotensi menjadi noise, dan dapat mengganggu performa model.



Say White

Preprocessing: Handling Outliers

- Outlier dengan nilai sangat tinggi dihapus karena tidak mewakili kondisi umum, berpotensi menjadi noise, dan dapat mengganggu performa model.

```
# Cek jumlah data outlier yang terlalu tinggi

print("Jumlah total_bedrooms > 5000 :", (df['total_bedrooms'] > 5000).sum())
print("Jumlah population > 25000    :", (df['population'] > 25000).sum())
print("Jumlah households > 5000     :", (df['households'] > 5000).sum())
```

```
Jumlah total_bedrooms > 5000 : 4
Jumlah population > 25000    : 2
Jumlah households > 5000     : 2
```

```
# Menghapus outliers

df = df[(df['total_bedrooms'] < 5000) & (df['population'] < 25000) & df['households'] < 5000 ]
```

Say White

Preprocessing: Handling Outliers Kolom Kategorikal

```
df['ocean_proximity'].value_counts()
```

```
ocean_proximity
<1H OCEAN     5998
INLAND        4561
NEAR OCEAN    1732
NEAR BAY      1477
ISLAND           2
Name: count, dtype: int64
```

```
# Drop data ISLAND
df=df.drop(df[df['ocean_proximity']=='ISLAND'].index)
```

- Kolom ocean_proximity merupakan satu-satunya kolom kategorikal dalam dataset.
- Ditemukan kategori ISLAND yang hanya memiliki 2 data (0.01% dari total data).
- Karena proporsinya sangat kecil, data ini tidak memberikan pengaruh signifikan terhadap model, berpotensi menjadi noise, dan bisa menyebabkan overfitting.
- Penghapusan dilakukan untuk menyeimbangkan distribusi kategori dan menjaga kestabilan serta efisiensi model.

Say White

Final Dataset After Preprocessing

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	ocean_proximity	median_house_value
0	-119.79	36.73	52.0	112.0	28.0	193.0	40.0	1.9750	INLAND	47500.0
1	-122.21	37.77	43.0	1017.0	328.0	836.0	277.0	2.2604	NEAR BAY	100000.0
2	-118.04	33.87	17.0	2358.0	396.0	1387.0	364.0	6.2990	<1H OCEAN	285800.0
3	-118.28	34.06	17.0	2518.0	1196.0	3051.0	1000.0	1.7199	<1H OCEAN	175000.0
4	-119.81	36.73	50.0	772.0	194.0	606.0	167.0	2.2206	INLAND	59200.0

```
<class 'pandas.core.frame.DataFrame'>
Index: 13768 entries, 0 to 14447
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           13768 non-null  float64
 1   latitude            13768 non-null  float64
 2   housing_median_age  13768 non-null  float64
 3   total_rooms         13768 non-null  float64
 4   total_bedrooms      13768 non-null  float64
 5   population          13768 non-null  float64
 6   households          13768 non-null  float64
 7   median_income       13768 non-null  float64
 8   ocean_proximity     13768 non-null  object
 9   median_house_value  13768 non-null  float64
dtypes: float64(9), object(1)
memory usage: 1.2+ MB
```

- Sebelum Preprocessing:
  - 14.448 baris data
  - 137 missing value pada kolom total_bedrooms
- Setelah Preprocessing:
  - 13768 baris data.
  - 0 missing value.
- Preprocessing dilakukan untuk memastikan kualitas data yang bersih, representatif, dan optimal bagi model.
- Data hasil preprocessing inilah yang akan digunakan dalam proses pemodelan machine learning.

03

Machine Learning Modeling

Say White

Define X & y and Data Splitting

- Variabel X: Seluruh kolom kecuali kolom target
- Variabel y: Kolom target (median_house_value)

```
X = df.drop(columns='median_house_value')
X.head(3)
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	ocean_proximity
0	-119.79	36.73	52.0	112.0	28.0	193.0	40.0	1.9750	INLAND
1	-122.21	37.77	43.0	1017.0	328.0	836.0	277.0	2.2604	NEAR BAY
2	-118.04	33.87	17.0	2358.0	396.0	1387.0	364.0	6.2990	<1H OCEAN

```
y = df['median_house_value']
y
```

```
0         47500.0
1        100000.0
2        285800.0
3        175000.0
4         59200.0
           ...
14443    144600.0
14444    159400.0
14445    289300.0
14446    484600.0
14447     69400.0
Name: median_house_value, Length: 13768, dtype: float64
```

- random_state = 0 untuk memastikan hasil pembagian data selalu konsisten.
- test_size = 0.2 artinya 80% data untuk training dan 20% data untuk testing.

```
X_train, X_test, y_train, y_test = train_test_split(
    X,
    y,
    test_size=0.2,
    random_state=0
)
```

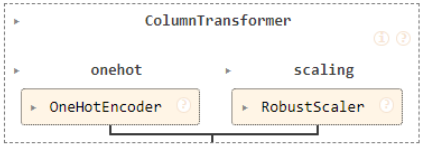
Say White

ColumnTransformer

```python
# Define Kolom kategorikal dan numerik
categorical_features = ['ocean_proximity']
numerical_features = ['total_bedrooms', 'longitude', 'latitude', 'housing_median_age',
                      'total_rooms', 'population', 'households', 'median_income']

# ColumnTransformer
transformer = ColumnTransformer([
    ('onehot', OneHotEncoder(), categorical_features),
    ('scaling', RobustScaler(), numerical_features)
])
```

```python
transformer
```



- **Encoding**: OneHotEncoder pada ocean_proximity.
- **Scaling**: RobustScaler pada seluruh kolom numerik.
- Semua proses encoding dan scaling dilakukan sekaligus dalam satu tahap preprocessing menggunakan **ColumnTransformer**.
- Bertujuan menyamakan skala data dan mengubah kategorikal ke numerik sebelum modelling.

```python
# Define Algo
linear = LinearRegression()
tree = DecisionTreeRegressor(random_state=0)
knn = KNeighborsRegressor()
rf = RandomForestRegressor(random_state=0)
xgb = XGBRegressor(random_state=0)

list_model = [linear, tree, knn, rf, xgb]

scorer = ["neg_root_mean_squared_error",
          "neg_mean_absolute_percentage_error",
          "neg_mean_absolute_error"]

# List Score RMSE
list_score_mean_rmse = []
list_score_std_rmse = []

# List Score MAE
list_score_mean_mae = []
list_score_std_mae = []

# List Score MAPE
list_score_mean_mape = []
list_score_std_mape = []

# Cross Validation loop
for model in list_model:
    pipe_model = Pipeline([
        ("Preprocessing", transformer),
        ("Modeling", model)])

    cv_score = cross_validate(
        estimator=pipe_model,
        X=X_train,
        y=y_train,
        cv=5,
        scoring=scorer,
        return_train_score=True)

    # Scoring RMSE
    list_score_mean_rmse.append(cv_score["test_neg_root_mean_squared_error"].mean())
    list_score_std_rmse.append(cv_score["test_neg_root_mean_squared_error"].std())

    # Scoring MAE
    list_score_mean_mae.append(cv_score["test_neg_mean_absolute_error"].mean())
    list_score_std_mae.append(cv_score["test_neg_mean_absolute_error"].std())

    # Scoring MAPE
    list_score_mean_mape.append(cv_score["test_neg_mean_absolute_percentage_error"].mean())
    list_score_std_mape.append(cv_score["test_neg_mean_absolute_percentage_error"].std())
```

Say White

Cross-Validation

- **Cross-validation**: teknik untuk mengevaluasi performa model dengan membagi data menjadi beberapa bagian (fold), lalu melatih dan menguji model secara bergantian di setiap fold.
- **Tujuannya**: memastikan hasil evaluasi lebih akurat, stabil, dan tidak tergantung pada pembagian data tertentu.
- Menggunakan **5 model algoritma**,
  - 3 model regresi (LinearRegression, KNN Regressor, DecisionTree Regressor)
  - 2 model ensemble (RandomForest & XGBoost).
- Scoring menggunakan 3 evaluation metrics: **RMSE**, **MAE** dan **MAPE**.
- Menggunakan **cross_validate** dengan **cv=5**, artinya data dibagi menjadi 5 bagian (fold).
- Skor dari tiap fold dikumpulkan, lalu diambil **nilai rata-rata dan standar deviasi** untuk ketiga metrik evaluasi.

Say White

Cross-Validation Result

Model	Mean RMSE	Std RMSE	Mean MAE	Std MAE	Mean MAPE	Std MAPE
Linear Regression	-60838.573079	947.335860	-44866.326358	545.676989	-0.275312	0.002524
DecisionTree Regressor	65625.040312	1442.197736	-43740.095131	789.047809	-0.252250	0.010011
KNN Regressor	-57553.518529	727.167475	-40250.091982	577.469729	-0.233694	0.003927
RandomForest Regressor	-47229.693990	**1021.618369**	-31955.507006	548.070451	-0.188229	**0.004900**
XGBoost Regressor	**-45351.901084**	1022.033405	**-30597.154542**	**513.790203**	**-0.179145**	0.005576

- XGBoost menunjukkan performa terbaik dengan nilai rata-rata RMSE, MAE, dan MAPE paling kecil.
- Random Forest memiliki nilai standard deviation yang sedikit lebih rendah pada RMSE dan MAPE.
- Untuk memastikan model terbaik, dilakukan *benchmarking* di test set guna mengevaluasi akurasi prediksi pada data baru.
- Model terbaik akan dipilih berdasarkan performa paling akurat dan konsisten di data yang belum pernah dilihat.

Say White

Benchmarking RandomForest & XGBoost

Benchmarking dilakukan dengan menguji kedua model ke data test (test set) yang belum pernah dipelajari, dengan hasil sebagai berikut:

Model	RMSE	MAE	MAPE
RandomForest Regressor	45230.488688	30816.670505	0.179419
XGBoost Regressor	44461.165270	30375.300244	0.177037

- Model XGBoost menghasilkan rata-rata nilai error (RMSE, MAE, dan MAPE) yang lebih kecil dibanding Random Forest, yang menunjukkan akurasi prediksi yang lebih tinggi.
- XGBoost terpilih sebagai model terbaik untuk melakukan modeling pada project ini.

Say White

```python
# Define Hyperparameter
hyperparameter = {
    'Modeling__n_estimators': list(np.arange(100, 1001, 100)),
    'Modeling__max_depth': list(np.arange(3, 22)),
    'Modeling__learning_rate': list(np.arange(1, 100)/100),
    'Modeling__subsample': list(np.arange(2, 11)/10),
    'Modeling__colsample_bytree': list(np.arange(1, 11)/10),
    'Modeling__reg_alpha': list(np.logspace(-3, 1, 10))  }

# Define Model
xgb_model = XGBRegressor(random_state=0)

# Define Scorer
scorer = ["neg_root_mean_squared_error",
          "neg_mean_absolute_percentage_error",
          "neg_mean_absolute_error"]

# Define pipeline with preprocessing and the XGBoost model
pipe_model = Pipeline([
    ("Preprocessing", transformer),
    ("Modeling", xgb_model)])

# Setup RandomizedSearchCV with cross-validation
randomized_xgb = RandomizedSearchCV(
    estimator=pipe_model,
    param_distributions=hyperparameter,
    n_iter=100,
    scoring=scorer,
    refit="neg_root_mean_squared_error",
    cv=5,
    n_jobs=-1,
    verbose=1,
    random_state=0)
```

Hyperparameter Tuning

n_estimators	:	Jumlah pohon dalam model, dicoba dari **100 hingga 1000** dengan kelipatan **100**
max_depth	:	Kedalaman maksimal tiap pohon, dicoba dari **3 hingga 21**
learning_rate	:	Kecepatan pembelajaran model, dicoba dari **0.01 hingga 0.99** dengan interval **0.01**
subsample	:	Proporsi data latih yang digunakan tiap pohon, dicoba dari **0.2 hingga 1.0** dengan interval **0.1**.
colsample_tree	:	Proporsi fitur yang digunakan tiap pohon, dicoba dari **0.1 hingga 1.0** dengan interval **0.1**.
reg_alpha	:	Nilai regularisasi L1 untuk mencegah overfitting, dicoba dari **0.001 hingga 10** dalam skala logaritmik

Menggunakan RandomizedSearchCV dengan 100 kombinasi acak, 5-fold cross-validation, dan multi-metric scoring dengan RMSE sebagai indikator scoring utama.

RandomizedSearch dipilih karena lebih efisien dari GridSearch untuk jumlah kombinasi hyperparameter yang banyak.

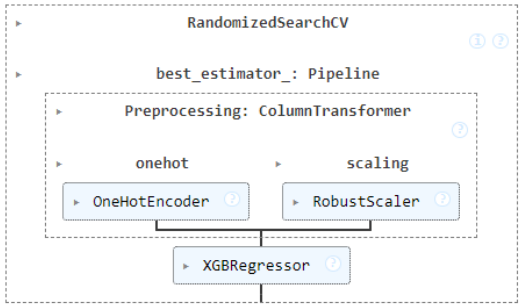
Say White

Fit the Model to Training Data

- Setelah dilakukan fitting ke data training, maka didapatkan Model, Best Score, dan Best Params sebagai berikut:

```
# Fit the model to the training data
randomized_xgb.fit(X_train, y_train)
```

```
Fitting 5 folds for each of 100 candidates, totalling 500 fits
```



Best_score: -43270.370758142395 (RMSE Score)
Best_params:

- **n_estimators** : 900
- **max_depth** : 4
- **learning_rate** : 0.11
- **subsample** : 0.7
- **colsample_tree** : 0.9
- **reg_alpha** : 0.007742636826811269

Say White

Predict to Test Set with Tuned Model

Sebelum Tuning

Model	RMSE	MAE	MAPE
XGBoost Regressor	44461.16527	30375.300244	0.177037

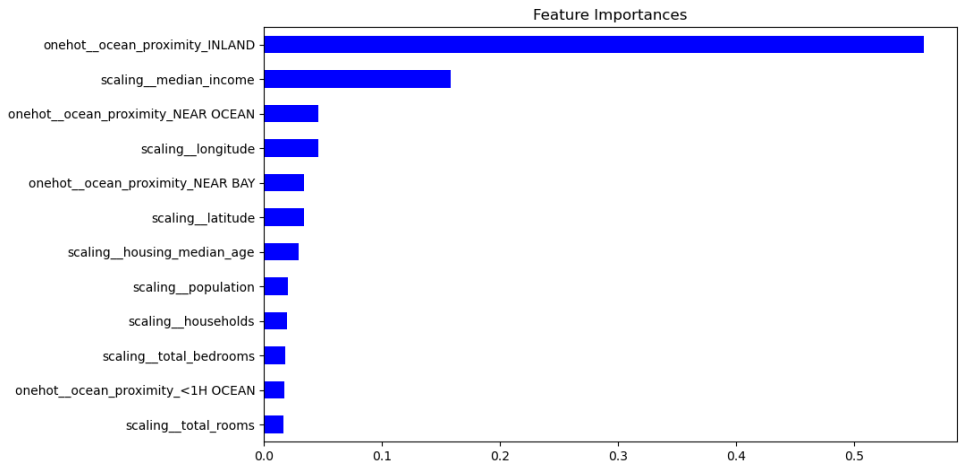
Setelah Tuning

Model	RMSE	MAE	MAPE
XGBoost Regressor	43560.707739	29474.336474	0.174652

- Berdasarkan perbandingan performa model XGBoost di atas, terlihat bahwa setelah dilakukan tuning, model mengalami penurunan nilai error pada ketiga metrik evaluasi (RMSE, MAE, dan MAPE).
- Hal ini menunjukkan bahwa tuning berhasil meningkatkan akurasi dan kinerja model XGBoost dalam memprediksi data, dengan penurunan yang signifikan pada nilai error di ketiga metrik evaluasi.
- Penurunan ini mencerminkan bahwa model lebih akurat dan stabil setelah dilakukan tuning, sehingga kinerja model semakin optimal.

Say White

Feature Importance



No.	Fitur	Importance
1	onehot__ocean_proximity_INLAND	0.559351
2	scaling__median_income	0.158754
3	onehot__ocean_proximity_NEAR OCEAN	0.046085
4	scaling__longitude	0.046009
5	onehot__ocean_proximity_NEAR BAY	0.034245
6	scaling__latitude	0.033781
7	scaling__housing_median_age	0.029425
8	scaling__population	0.020195
9	scaling__households	0.019407
10	scaling__total_bedrooms	0.018502

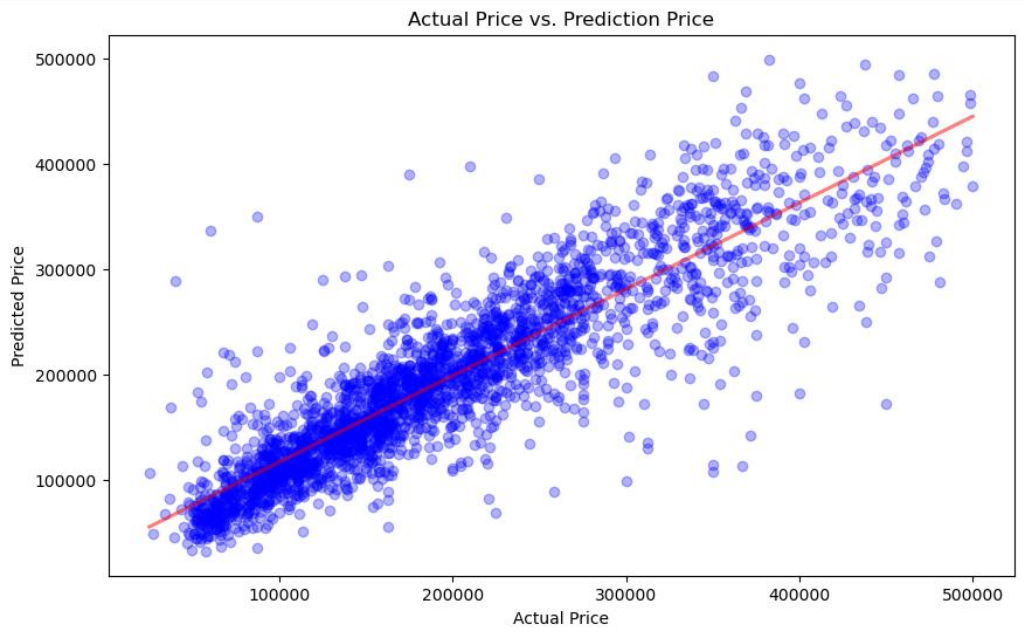
- **ocean_proximity_INLAND** merupakan fitur yang memiliki pengaruh paling besar dalam prediksi harga properti, dengan nilai importance mencapai **0.56**.
- Fitur lainnya yang cukup penting adalah **median_income** dengan nilai importance **0.16**.
- Secara umum, **lokasi dan tingkat pendapatan** menjadi faktor paling dominan dalam menentukan harga properti menurut hasil model machine learning.

04

Model Visualization

Say White

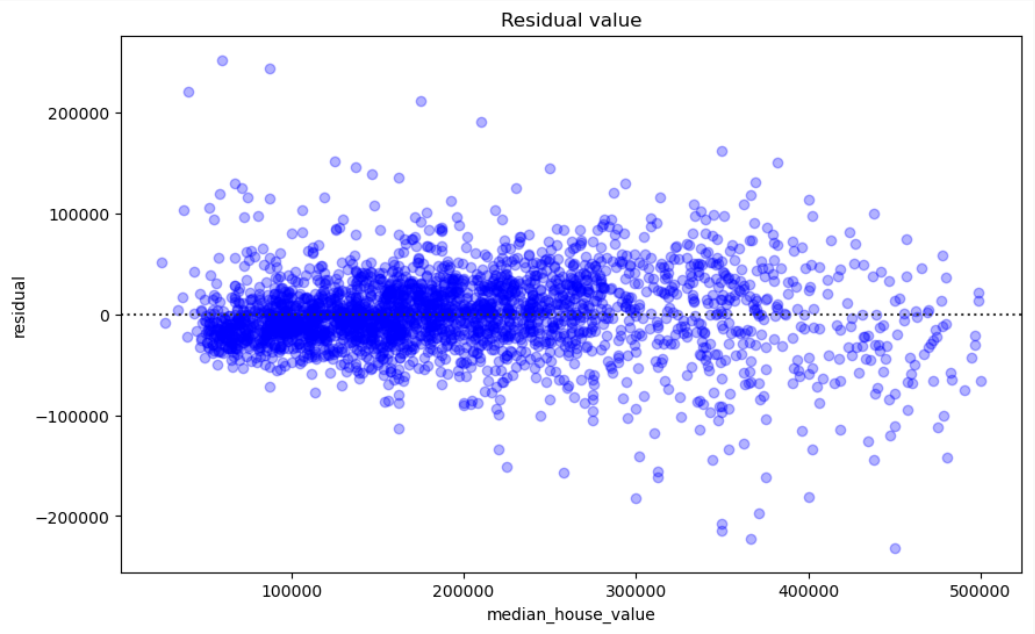
Scatterplot: Actual vs. Prediction Price



- **Garis regresi positif antara harga aktual dan harga prediksi,** artinya model dapat memprediksi harga rumah secara proporsional terhadap nilai aslinya.
- **Sebagian besar prediksi berada dekat dengan garis regresi,** menunjukkan model memiliki akurasi yang baik secara umum.
- Model cenderung lebih akurat dalam memprediksi harga rumah pada rentang rendah hingga sedang.
- **Terdapat beberapa outlier pada harga rumah yang sangat tinggi**, yang artinya model masih kesulitan dalam memprediksi harga rumah yang tinggi.

Say White

Residual Plot



- Sebagian besar prediksi model cukup akurat karena **residual tersebar dekat garis nol**.
- **Sebaran residual melebar pada harga rumah tinggi**. Menunjukkan prediksi model cenderung lebih bervariasi dan kurang stabil pada rumah yang mahal.
- **Tidak terlihat adanya pola tertentu pada residual.** Menunjukkan bahwa asumsi linearitas dan homoskedastisitas relatif terpenuhi untuk harga rumah rendah hingga menengah.

05

Conclusion & Recommendation

Say White

Conclusion

- Model terbaik: XGBoost merupakan model terbaik berdasarkan hasil cross-validation dan benchmarking, dengan nilai error (RMSE, MAE, dan MAPE) yang lebih rendah.
- **Hasil metrics evaluasi:**
    - RMSE : 43.560
    - MAE : 29.474
    - MAPE : 0.17

    Berdasarkan nilai MAPE dan MAE, dapat disimpulkan bahwa rata-rata prediksi model akan menyimpang sekitar 17% ($29.474) dari harga aktual, jika digunakan pada data baru.
- **Visualisasi (Actual vs Predicted):** Model mampu memprediksi harga rumah dengan baik pada rentang harga menengah ke bawah, namun masih kesulitan dalam memprediksi rumah harga tinggi .
- **Feature Immportance:** Fitur ocean_proximity_INLAND memberikan kontribusi terbesar dalam prediksi harga rumah (56%) .

Say White

Recommendation

Untuk Model Machine Learning:

- **Lakukan hyperparameter tuning dengan GridSearch** untuk meningkatkan performa model secara optimal.
- **Terapkan feature selection secara iteratif** untuk memilih fitur terbaik dan menyederhanakan model.
- **Lakukan evaluasi dan retraining model secara berkala,** minimal 6 bulan sekali agar tetap relevan dengan dinamika pasar properti.

Untuk Dataset:

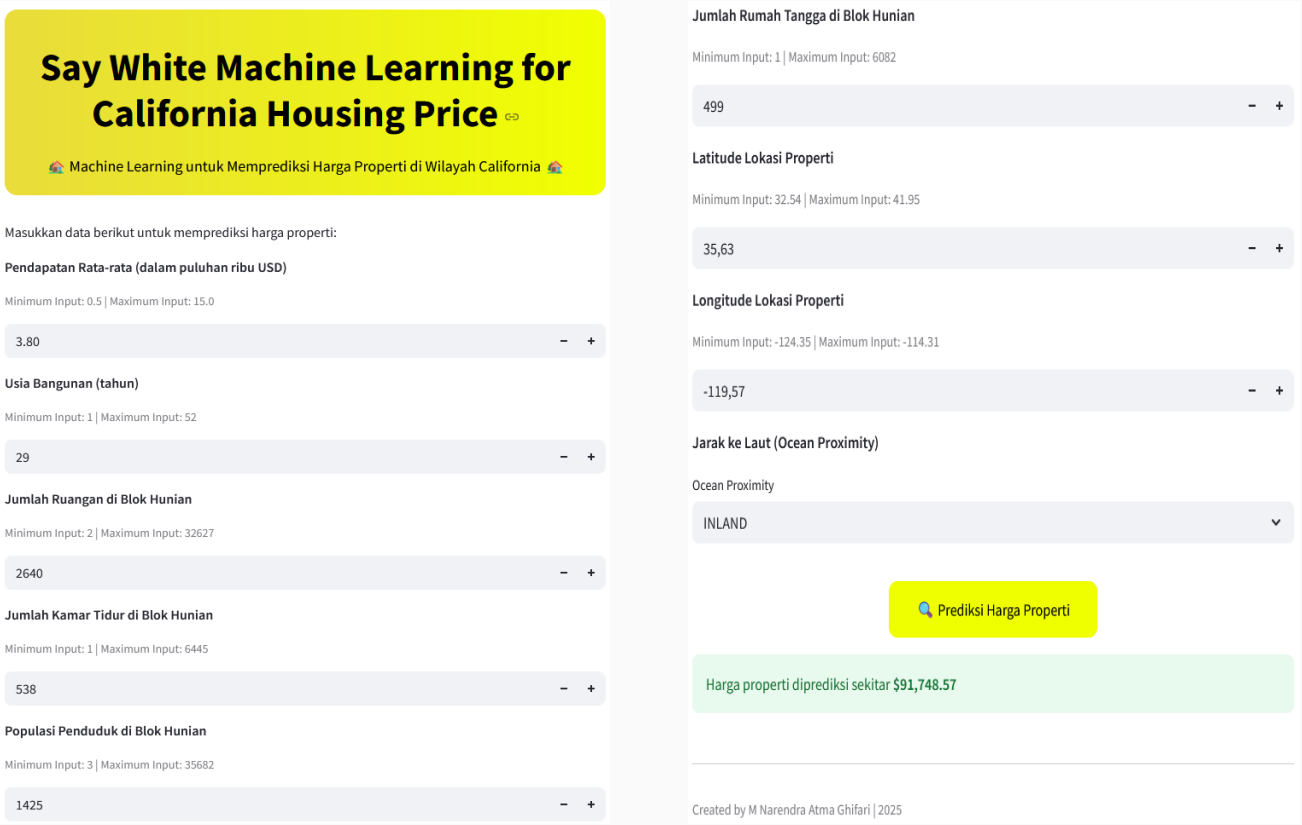
- **Update dataset dengan data terbaru** untuk mengakomodasi inflasi dan perubahan tren pasar.
- **Penambahan Feature baru** yang berpotensi mempengaruhi harga properti secara signifikan (luas tanah, luas bangunan, dll) untuk meningkatkan akurasi prediksi.

06

Model Deployment

Streamlit

Say White



[Access Streamlit]

Thanks!

Feel free to reach me out:

mnaghifari@gmail.com.com
github.com/mnaghifari