LAPORAN
PEMBAHASAN HASIL STUDI KASUS
DATA COMPETITION ISFEST 2023

Kelompok Bebas :

1. **Fahmi Agung Maulana**
2. **Gloria Natasya Irene Sidebang**
3. **Muhammad Ramdhan Fitra Hidayat**

INFORMATION SYSTEM FESTIVAL UMN

2023

BUSINESS UNDERSTANDING

A. Latar Belakang Permasalahan

Tempat tinggal adalah kebutuhan mendasar setiap makhluk hidup, khususnya manusia. Lokasi geografis, karakteristik fisik, fasilitas di sekitar, kondisi pasar, dan lain-lain adalah hal yang mungkin memengaruhi harga sebuah tempat tinggal. Ketersediaan data-data terkait harga rumah dan deskripsi terkait rumah tersebut pun dapat ditemui dengan mudah di aplikasi jual-beli *online*. Ketersediaan data dan urgensi terkait penentuan faktor-faktor yang memengaruhi harga rumah adalah hal yang melatar belakangi penelitian ini. Oleh karena itu, analisis data terkait harga rumah diperlukan.

B. Tujuan Penelitian

Tujuan dari analisis data ini adalah untuk menentukan faktor-faktor apa yang sangat memengaruhi harga sebuah rumah. Visualisasi data juga menjadi proses penting untuk membantu analisis.Selain itu, penelitian ini juga bertujuan untuk menentukan model *machine learning* terbaik dari beberapa model yang akan digunakan pada penelitian ini.

C. Kebutuhan Objek

Pada penelitian dibutuhkan data set terkait harga rumah dan deskripsi terkait rumah tersebut. Kami menggunakan data set yang telah disediakan dan dapat di akses di sini. Kami menggunakan bahasa pemrograman python dan beberapa *library* seperti Pandas, Numpy, Matplotlib, Seaborn, dan Sklearn serta platform visualisasi data Tableau untuk melakukan penelitian terhadap dataset.

D. Batasan dan Parameter Permasalahan

Pada penelitian ini data terkait harga rumah yang akan dianalisis hanya berlokasi pada area JABODETABEK, sebagaimana yang terdapat pada data set. Selanjutnya, analisis prediksi harga rumah hanya menggunakan atribut-atribut yang terdapat pada data set tanpa ada tambahan lain.

Parameter keberhasilan penelitian ini dapat diukur dari beberapa hal :

- Penentuan faktor-faktor yang memengaruhi harga rumah

Dari beberapa atribut yang tersedia pada data set akan ditentukan beberapa atribut yang paling memengaruhi harga rumah di area JABODETABEK.

- Penentuan model machine learning terbaik yang digunakan untuk menganalisis data

  Penentuan algoritma terbaik didasarkan pada nilai R-squared (R^2), Mean Squared Error (MSE), Root Mean Squared Error RMSE) untuk setiap model.

- Visualisasi data set

  penelitian ini diharapkan data menghasilkan *dashboard* yang lengkap dan informatif

E. Metode

Framework yang digunakan pada penelitian ini adalah *framework* CRISP-DM, yang terdiri dari tahapan *business understanding, data understanding, data preparation, prediction model and evaluation*, dan *conclusion and suggestion.*

Pada tahap prediction model dan evaluation digunakan beberapa model *multiple-regression,* yaitu Linear Regresi, Ridge Regresi, Lasso Regresi, KNN, Decision Tree, dan Random Forest, untuk memprediksi atribut 'price_in_rp' yang selanjutnya akan dipilih model terbaik berdasarkan performa nilai MSE, RMSE dan R^2 setiap model. Penentuan atribut terbaik akan dilakukan dengan berbagai percobaan dengan iterasi kombinasi atribut. Kombinasi atribut yang menghasilkan nilai performa model yang paling baik akan dipilih menjadi atribut yang memengaruhi harga rumah di JABODETABEK.

DATA UNDERSTANDING

A. Identifikasi Data

Data set yang digunakan pada analisis ini adalah data yang telah disediakan dan dapat diakses pada link ini. Data set yang digunakan terdiri dari 27 atribut dan 3553 baris. Angka 3553 baris terdapat 3553 buah rumah yang tercatat pada dataset. Sedangkan, atribut merupakan data-data detail terkait lokasi, harga, dan spesifikasi rumah tersebut

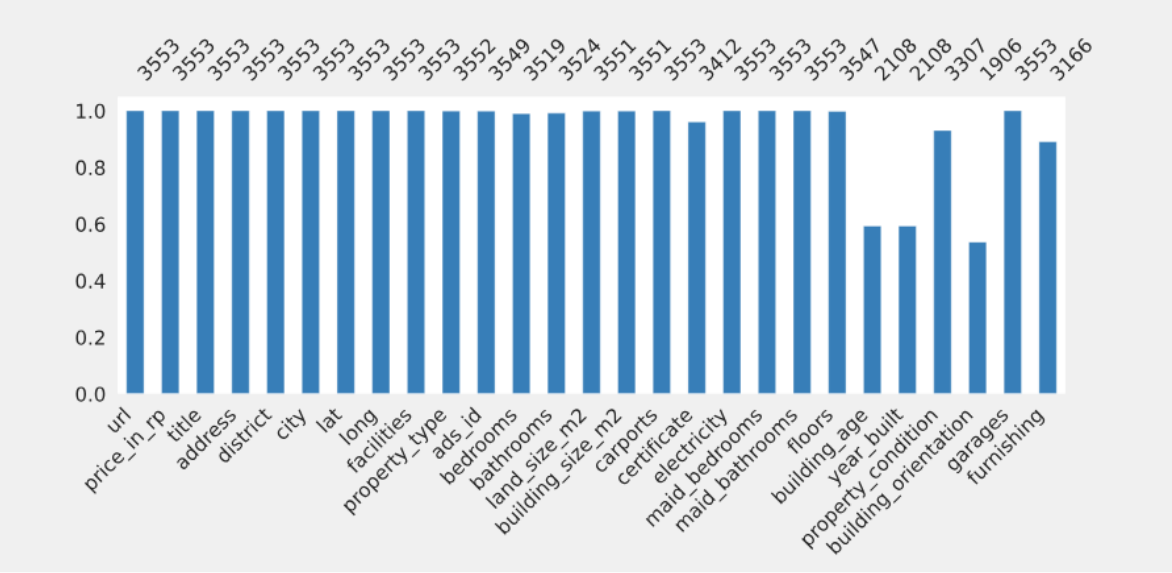
yang terdiri dari 14 atribut bertipe data kategorik dan 13 atribut bertipe data numerik dengan deskripsi sebagai berikut :

1. url : URL listing rumah (kategorik, nominal)
2. price_in_rp : Harga rumah dalam Rupiah (numerik , interval)
3. title : Nama rumah (kategorik, nominal)
4. address : Alamat rumah (kategorik , nominal)
5. district : Kecamatan (kategorik, nominal)
6. city : Kota (kategorik, nominal)
7. lat : Latitude (numerik, interval)
8. long : Longitude (numerik, interval)
9. facilities : List fasilitas yang ada di rumah (kategorik, nominal)
10. property_type : Tipe properti (kategorik, nominal)
11. ads_id : ID iklan (kategorik, nominal)
12. bedrooms : Jumlah kamar tidur (numerik, rasio)
13. bathrooms : Jumlah kamar mandi (numerik, rasio)
14. land_size_m2 : Luas tanah (numerik, rasio)
15. building_size_m2 : Luas bangunan (numerik, rasio)
16. carports : Jumlah tempat parkir mobil (numerik, rasio)
17. certificate : Sertifikat (kategorik, nominal)
18. electricity : Daya listrik rumah (kategorik, nominal)
19. maid_bedrooms : Jumlah kamar tidur ART (numerik, rasio)
20. maid_bathrooms : Jumlah kamar mandi ART (numerik, rasio)
21. floors : Tingkat bangunan (kategorik, nominal)
22. building_age : Usia bangunan (numerik, rasio)
23. year_built : Tahun bangunan didirikan (numerik, interval)
24. property_condition : Kondisi rumah (kategorik, nominal)
25. building_orientation : Orientasi bangunan (kategorik, nominal)
26. garages : Jumlah garasi (numerik, rasio)
27. furnishing : Kondisi furnishing di rumah (kategorik, nominal)

Atribut yang akan diprediksi pada penelitian ini adalah price_in_rp.

B. Kelengkapan Data

Pada penelitian ini kami menggunakan modul Pandas Profiling untuk mempermudah pengecekan data. Berikut adalah beberapa ringkasan yang dihasilkan dari pengecekan dataset :



A simple visualization of nullity by column.

Barplot di atas menggambarkan jumlah *missing value* untuk setiap atribut. Terlihat bahwa terdapat 13 atribut yang memiliki *missing value*, yaitu :

1. property_type (1 record)
2. ads_id (4 records)
3. bedrooms (34 records)
4. bathrooms (29 records)
5. land_size_m2 (2 records)
6. building_size_m2 (2 records)
7. certificate (141 records)
8. floors(6 records)
9. building_age(1445 records)
10. year_built(1445 records)
11. property_condition(246 records)
12. building_orientation(1647 records)

13. furnishing(387 records)

Sehingga presentasi total dari nilai *missing value* adalah 5,6% dari keseluruhan data. Penanganan terhadap nilai missing value akan dijelaskan lebih detail pada tahapan *data preparation*. Hasil lengkap dari pengecekan data dengan modul Pandas Profiling dapat dilihat di sini.

C. Evaluasi Kualitas Data

Dari 27 atribut, 14 atribut bertipe data kategorik dan 13 atribut bertipe data numerik, dan 3553 record, dataset memiliki missing value sebesar 5,6% dari keseluruhan dataset. Berdasarkan analisis missing value, dataset masih tergolong data yang bagus. Namun, diperlukan tahap pre-processing lanjutan sebelum masuk kepada tahap pemodelan.

DATA PREPARATION

A. Select Data

Pada penelitian ini, atribut yang tidak digunakan adalah 'url', 'title', 'property_type', 'address', 'district', 'facilities', 'building_age', 'year_built', 'building_orientation', ‘ads_id’.

‘ads_id’, 'url', 'title', dan ‘address' tidak digunakan karena merupakan atribut yang memiliki nilai unik untuk setiap record yang artinya tidak berpengaruh terhadap prediksi harga rumah.

B. Data Cleaning

a. Missing Value

Mengatasi missing value merupakan proses pengelolaan data yang dilakukan ketika terdapat nilai yang hilang atau tidak tersedia dalam suatu dataset. Tujuan utama dari handle missing value adalah untuk menjaga integritas dan kualitas data yang akan digunakan dalam analisis atau pemodelan.

Pertama, yang kami lakukan adalah memeriksa kolom mana saja yang terdapat missing value di dalamnya seperti berikut:

```
[263] data_encoded.isnull().sum()

price_in_rp                0
city                       0
lat                        0
long                       0
bedrooms                  34
bathrooms                 29
land_size_m2               2
building_size_m2           2
carports                   0
certificate              141
electricity                0
maid_bedrooms              0
maid_bathrooms             0
floors                     6
building_age            1445
year_built              1445
property_condition       246
building_orientation    1647
garages                    0
furnishing               387
dtype: int64
```

Terlihat bahwa kolom property_type, ads_id, bedrooms, bathrooms, land_size_m2, building_size_m2, certificate, floors, building_age, year_built, property_condition, building_orientation dan furnishing mengandung missing value

Selanjutnya, kami akan drop kolom yang memiliki banyak missing value, dan mengisi beberapa kolom yang memiliki missing value dengan jumlah yang lebih sedikit dengan data mediannya menggunakan simple imputer

```
[ ]  data_encoded = data_encoded.drop(['building_age', 'year_built', 'building_orientation'], axis=1)

[ ]  from sklearn.impute import SimpleImputer
     imputer = SimpleImputer(strategy='median')  # Menggunakan median untuk mengisi nilai hilang
     data_imputed = imputer.fit_transform(data_encoded)
     data_imputed = pd.DataFrame(data_imputed, columns=data_encoded.columns)
```

b. Data Redundancy

Data redundan mengacu pada adanya informasi yang berlebihan atau berulang dalam dataset yang digunakan untuk melatih model. Data redundan

dapat menjadi masalah karena dapat mengakibatkan peningkatan waktu komputasi, penggunaan sumber daya yang berlebihan, dan kemungkinan mempengaruhi kinerja model.

Kami menghapus data redundan dengan menggunakan method drop duplicates, seperti berikut:

```
[ ]  data_imputed.shape

     (3553, 17)

[ ]  final_data = data_imputed.drop_duplicates()
     final_data.shape

     (3186, 17)
```

c. Outlier

Outlier merujuk pada nilai yang jauh atau tidak biasa dibandingkan dengan sebagian besar data dalam suatu kumpulan data. Kami menggunakan metode iqr untuk menghapus outlier seperti berikut:

```
[ ]  #handling outlier
     def winsorize_column_iqr(df, column, multiplier):
         q1 = df[column].quantile(0.25)
         q3 = df[column].quantile(0.75)
         iqr = q3 - q1
         lower_limit = q1 - multiplier * iqr
         upper_limit = q3 + multiplier * iqr
         df[column] = np.where(df[column] < lower_limit, lower_limit, df[column])
         df[column] = np.where(df[column] > upper_limit, upper_limit, df[column])
         return df


     column_name = ['lat','long','bedrooms','land_size_m2','building_size_m2','carports','maid_bedrooms','maid_bathrooms','floors','garages']
     iqr_multiplier = 1.5

     for col in column_name:
       df = winsorize_column_iqr(df, col, iqr_multiplier)
```

```
[ ]  Q1 = df['price_in_rp'].quantile(0.25)
     Q3 = df['price_in_rp'].quantile(0.75)
     IQR = Q3 - Q1
     lower_bound = Q1 - 1.5 * IQR
     upper_bound = Q3 + 1.5 * IQR
     df = df[(df['price_in_rp'] >= lower_bound) & (df['price_in_rp'] <= upper_bound)]
```

d. Oversampling Data

Oversampling adalah salah satu teknik yang digunakan dalam pemrosesan data yang tidak seimbang (imbalanced data) di bidang machine learning.

Oversampling melibatkan peningkatan jumlah sampel pada kelas minoritas dalam dataset sehingga kelas minoritas memiliki jumlah sampel yang setara dengan kelas mayoritas.

Kami menerapkan oversampling menggunakan library imblearn seperti berikut:

```
[ ]  from imblearn.over_sampling import RandomOverSampler
     ros = RandomOverSampler(random_state=0)
     X_resampled, y_resampled = ros.fit_resample(X, y)
```

C. Transformation Data

a. Transformation Data Categorical

Dari data yang diberikan, diketahui tipe data dari setiap kolom sebagai berikut

```
[ ]  df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3553 entries, 0 to 3552
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   price_in_rp           3553 non-null   float64
 1   city                  3553 non-null   object
 2   lat                   3553 non-null   float64
 3   long                  3553 non-null   float64
 4   bedrooms              3519 non-null   float64
 5   bathrooms             3524 non-null   float64
 6   land_size_m2          3551 non-null   float64
 7   building_size_m2      3551 non-null   float64
 8   carports              3553 non-null   int64
 9   certificate           3412 non-null   object
 10  electricity           3553 non-null   object
 11  maid_bedrooms         3553 non-null   int64
 12  maid_bathrooms        3553 non-null   int64
 13  floors                3547 non-null   float64
 14  building_age          2108 non-null   float64
 15  year_built            2108 non-null   float64
 16  property_condition    3307 non-null   object
 17  building_orientation  1906 non-null   object
 18  garages               3553 non-null   int64
 19  furnishing            3166 non-null   object
dtypes: float64(10), int64(4), object(6)
memory usage: 555.3+ KB
```

Untuk mengatasi data kategorikal, kami menggunakan metode ordinal encoder yang akan mengubah tipe data object menjadi numerik dengan cara seperti berikut

```
[ ]  from sklearn.preprocessing import OrdinalEncoder

     # Make copy to avoid changing original data
     data_encoded = df.copy()

     # Apply ordinal encoder to each column with categorical data
     ordinal_encoder = OrdinalEncoder()
     data_encoded[object_cols] = ordinal_encoder.fit_transform(df[object_cols])
```

b. Transformation Data Skewed

Dari data yang ada, terdapat beberapa data yang skewed atau distribusinya tidak normal. Kolom-kolom yang memiliki data skewed tersebut yaitu kolom bedrooms, maid_bedrooms, dan garages. Untuk mengatasi hal tersebut dilakukan transformation data. Sebelum dilakukannya transformation data, terlebih dahulu dilakukan pengecekan jenis transformation yang cocok untuk kolom tersebut, Pengecekan dilakukan dengan kode berikut

```
#bedrooms skewed
bed_log = np.log(df['bedrooms'])
bed_log.skew()
```

```
0.01909163836882669
```

```
maidBed_sqrt = np.sqrt(df['maid_bedrooms'])
maidBed_sqrt.skew()
```

```
0.6199975506556292
```

```
bed_sqrt = np.sqrt(df['garages'])
bed_sqrt.skew()
```

```
0.4639276662561991
```

Setelah menemukan jenis transformation yang cocok, maka dilakukan transformation untuk masing-masing kolom dengan jenis yang telah ditentukan

```
df['bedrooms'] = np.sqrt(df['bedrooms'])
df['maid_bedrooms'] = np.sqrt(df['maid_bedrooms'])
df['garages'] = np.sqrt(df['garages'])
```

PREDICTION MODEL AND EVALUATION

A. Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) adalah proses analisis awal yang dilakukan untuk memahami dan menganalisis karakteristik dasar dari suatu dataset. Tujuan dari EDA adalah untuk mengeksplorasi data, mengidentifikasi pola, hubungan, dan anomali yang mungkin ada dalam data sebelum menjalankan model statistik atau membangun model machine learning.

```
[ ] df.shape

    (3553, 27)
```

Data ini terdiri dari 27 kolom dan 3553 entri. Kita dapat melihat semua 27 dimensi dari dataset kita dengan mencetak 5 entri pertama menggunakan kode berikut:

```
[ ] df.head(5)
```

	url	price_in_rp	title	address	district	city	lat	long	facilities	property_type	...
0	https://www.rumah123.com/properti/bekasi/hos11...	2.990000e+09	Rumah cantik Sumarecon Bekasi\nLingkungan asri...	Summarecon Bekasi, Bekasi	Summarecon Bekasi	Bekasi	-6.223945	106.986275	Tempat Jemuran, Jalur Telepon, Taman, Taman	rumah	...
1	https://www.rumah123.com/properti/bekasi/hos10...	1.270000e+09	Rumah Kekinian, Magenta Summarecon Bekasi	Summarecon Bekasi, Bekasi	Summarecon Bekasi	Bekasi	-6.223945	106.986275	Taman	rumah	...
2	https://www.rumah123.com/properti/bekasi/hos10...	1.950000e+09	Rumah Cantik 2 Lantai Cluster Bluebell Summare...	Summarecon Bekasi, Bekasi	Summarecon Bekasi	Bekasi	-6.223945	106.986275	Jogging Track, Kolam Renang, Masjid, Taman,...	rumah	...
3	https://www.rumah123.com/properti/bekasi/hos10...	3.300000e+09	Rumah Mewah 2Lantai L10x18 C di Cluster VERNON...	Summarecon Bekasi, Bekasi	Summarecon Bekasi	Bekasi	-6.223945	106.986275	Jalur Telepon, Jogging Track, Track Lari, K...	rumah	...

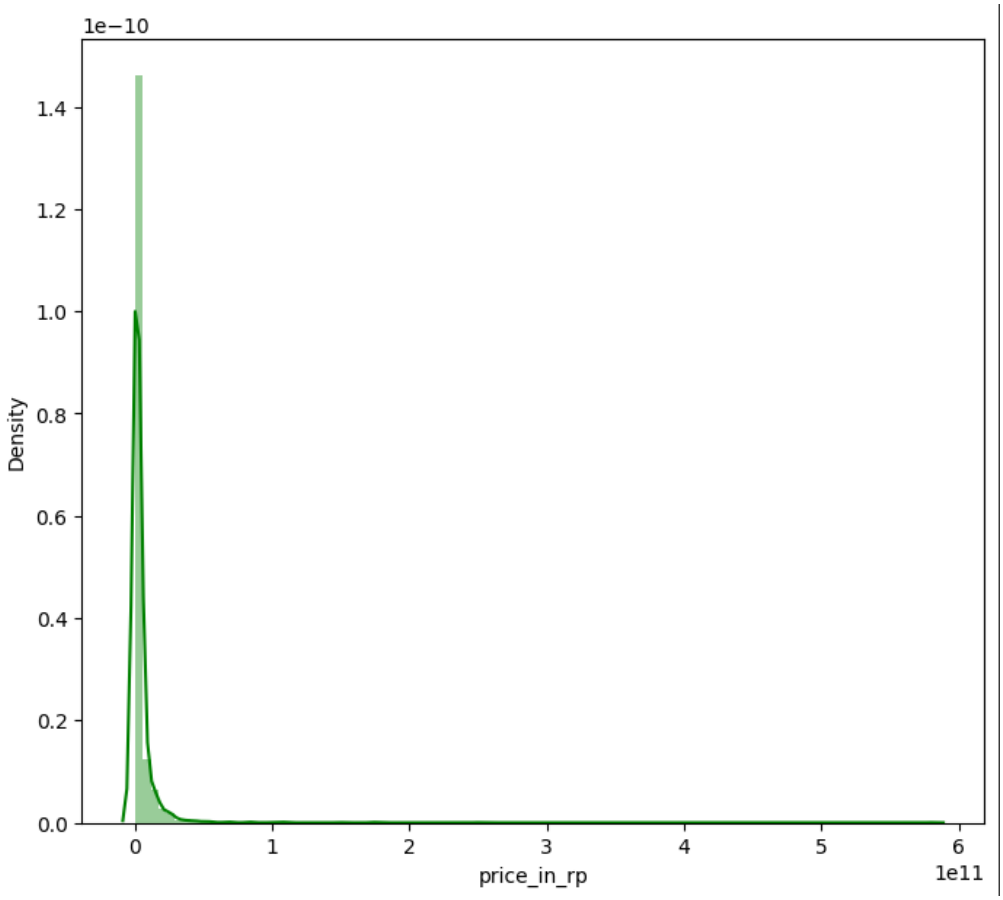
Penting untuk memeriksa tipe data dari setiap kolom yang ada dalam dataset, kami melakukannya dengan cara seperti gambar berikut:

```
[ ]  df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3553 entries, 0 to 3552
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   price_in_rp           3553 non-null   float64
 1   city                  3553 non-null   object 
 2   lat                   3553 non-null   float64
 3   long                  3553 non-null   float64
 4   bedrooms              3519 non-null   float64
 5   bathrooms             3524 non-null   float64
 6   land_size_m2          3551 non-null   float64
 7   building_size_m2      3551 non-null   float64
 8   carports              3553 non-null   int64  
 9   certificate           3412 non-null   object 
 10  electricity           3553 non-null   object 
 11  maid_bedrooms         3553 non-null   int64  
 12  maid_bathrooms        3553 non-null   int64  
 13  floors                3547 non-null   float64
 14  building_age          2108 non-null   float64
 15  year_built            2108 non-null   float64
 16  property_condition    3307 non-null   object 
 17  building_orientation  1906 non-null   object 
 18  garages               3553 non-null   int64  
 19  furnishing            3166 non-null   object 
dtypes: float64(10), int64(4), object(6)
memory usage: 555.3+ KB
```

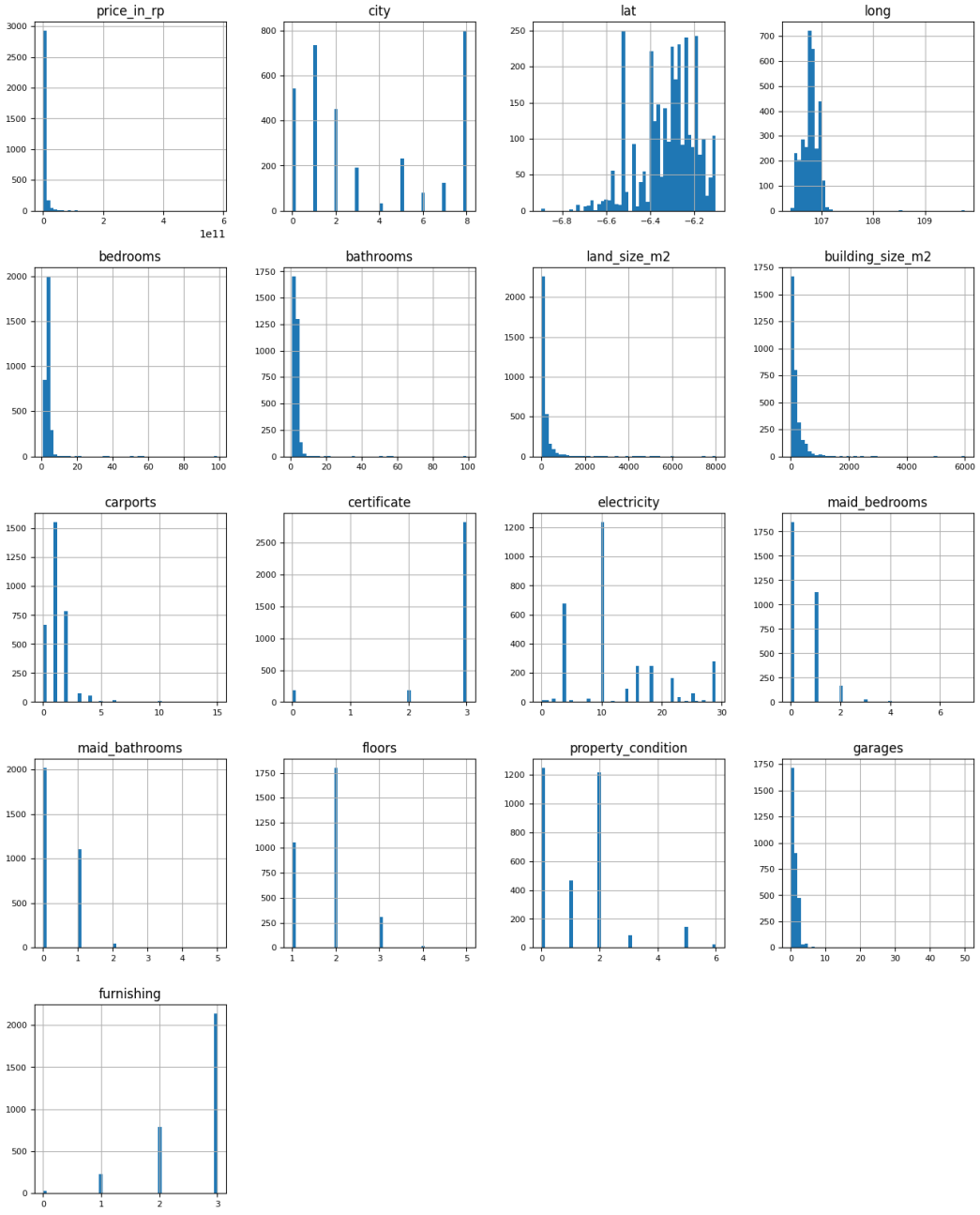
Dalam dataset yang diberikan, data target adalah harga rumah pada kolom price_in_rp, sehingga kita perlu mengetahui distribusi harga rumah tersebut. Distribusi harga rumah dapat dilihat dari gambar berikut

```
print(final_data['price_in_rp'].describe())
plt.figure(figsize=(8, 7))
sns.distplot(final_data['price_in_rp'], color='g', bins=100, hist_kws={'alpha': 0.4})
```



Selanjutnya penting juga untuk meninjau grafik histogram dari fitur-fitur yang akan digunakan pada training model untuk memberikan gambaran visual tentang distribusi data pada suatu variabel

```
final_data.hist(figsize=(16, 20), bins=50, xlabelsize=8, ylabelsize=8)
```



Setelah melihat visualisasi dari fitur-fitur tersebut, kita perlu melihat korelasi dari data-data tersebut dengan data target-nya, sehingga kita bisa menentukan data mana yang paling penting untuk dipilih

```
[ ]  # Hitung korelasi antara fitur-fitur dan target
     correlation = X.corrwith(y)

     # Urutkan berdasarkan nilai absolut korelasi
     sorted_correlation = correlation.abs().sort_values(ascending=False)

     # Tampilkan nama-nama fitur yang memiliki korelasi tertinggi dengan target
     important_features = sorted_correlation.index
     print("Fitur yang paling penting:")
     for feature in important_features:
         print(feature)
```

```
Fitur yang paling penting:
building_size_m2
land_size_m2
maid_bathrooms
electricity
maid_bedrooms
bedrooms
floors
carports
long
lat
garages
```

Kami juga membuat visualisasi menggunakan platform tableau yang membantu dalam analisis data ini, visualisasi lengkap dapat dilihat pada link ini.

B. Modeling & Evaluation

Pada analisis data harga rumah Jabodetabek, terdapat beberapa model yang digunakan antara lain Linear Regresi, Ridge Regresi, Lasso Regresi, KNN, Decision Tree, dan Random Forest. Pemilihan model-model tersebut didasarkan bahwa model tersebut merupakan model regresi yang biasa digunakan untuk memprediksi suatu hal.

Berdasarkan model-model yang telah untuk memprediksi harga rumah, model yang pilih merupakan model Random Forest. Model Random Forest adalah teknik regresi berbasis pohon keputusan untuk mengatasi masalah nonlinier. Dalam penerapannya, model random forest membuat beberapa pohon keputusan yang masing-masing membuat satu pohon menggunakan pengamatan dan prediksi yang berbeda dari hasil sampel yang dilatih dan dicari hasil terbaiknya.

Pemilihan model Random Forest ditetapkan berdasarkan nilai R2 Score, MAE, MSE yang mana nilai terbaik diraih oleh Random Forest. Berikut merupakan nilai yang didapat dari model-model yang digunakan

	Linear Regresi	Ridge Regresi	Lasso Regresi	KNN	Decision Tree	Random Forest
R2 Score	0.8218261331902194	0.821827154764736363	0.821826133190228787	0.990034495508955 2	0.95110 4116052 5426	0.994103876 3727752
MAE	480356743 .27564055	48033761 5.799035 85	48035674 3.275385 3	28470252. 52525252 5	2385989 83.0440 15	44312482.33 6501725
MSE	6.0252877 46793627e +17	6.025253 20031085 8e+17	6.025287 74679331 1e+17	3.3700246 38047138 e+16	1.65350 7193239 441e+17	1.993886201 2028804e+16

Dalam penggunaan model tersebut, data yang digunakan harus dibagi terlebih dahulu menjadi train dan test. Di sini, kami menggunakan test size berupa 0.2. Pembagian data menggunakan kode berikut

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Setelah dilakukan pembagian data, maka diperlukan mengimport library yang akan digunakan dalam pembuatan model dan import library untuk mengecek nilai r2 score, MAE, MSE, serta untuk validasi model yang dibuat.

```
from sklearn.metrics import r2_score,mean_absolute_error, mean_squared_error

from sklearn.linear_model import LinearRegression,Ridge,Lasso
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor,GradientBoostingRegressor,AdaBoostRegressor,ExtraTreesRegressor
from sklearn.svm import SVR
from xgboost import XGBRegressor

from numpy import absolute
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedKFold
```

Pembentukan model regresi Random Forest dalam analisis ini menggunakan library berupa scikit-learn dengan kode sebagai berikut

```
model = RandomForestRegressor(n_estimators=100,
                              random_state=3,
                              max_samples=0.5,
                              max_features=0.75,
                              max_depth=15)
# Melatih model pada set pelatihan
model.fit(X_train, y_train)

# Membuat prediksi pada set pengujian
y_pred = model.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
print('MSE',mean_squared_error(y_test,y_pred))
```

```
R2 score 0.9941038763727752
MAE 44312482.336501725
MSE 1.9938862012028804e+16
```

Sedangkan, untuk pembentukan model regresi lainnya adalah sebagai berikut

Linear Regression

```
model = LinearRegression()
# Melatih model pada set pelatihan
model.fit(X_train, y_train)

# Membuat prediksi pada set pengujian
y_pred = model.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
print('MSE',mean_squared_error(y_test,y_pred))
```

```
R2 score 0.8218261331902194
MAE 480356743.27564055
MSE 6.025287746793627e+17
```

Ridge Regression

```
model = Ridge(alpha=10)
# Melatih model pada set pelatihan
model.fit(X_train, y_train)

# Membuat prediksi pada set pengujian
y_pred = model.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
print('MSE',mean_squared_error(y_test,y_pred))
```

```
R2 score 0.8218271547647363
MAE 480337615.79903585
MSE 6.025253200310858e+17
```

Lasso Regression

```
model = Lasso(alpha=0.001)
# Melatih model pada set pelatihan
model.fit(X_train, y_train)

# Membuat prediksi pada set pengujian
y_pred = model.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
print('MSE',mean_squared_error(y_test,y_pred))
```

```
R2 score 0.8254132696388272
MAE 480485075.8626162
MSE 5.903981914031136e+17
```

KNN

```
model = KNeighborsRegressor(n_neighbors=3)
# Melatih model pada set pelatihan
model.fit(X_train, y_train)

# Membuat prediksi pada set pengujian
y_pred = model.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
print('MSE',mean_squared_error(y_test,y_pred))
```

```
R2 score 0.9905118881030175
MAE 30012070.707070712
MSE 3.208585264309764e+16
```

Desicion Tree

```
model = DecisionTreeRegressor(max_depth=8)
# Melatih model pada set pelatihan
model.fit(X_train, y_train)

# Membuat prediksi pada set pengujian
y_pred = model.predict(X_test)

print('R2 score',r2_score(y_test,y_pred))
print('MAE',mean_absolute_error(y_test,y_pred))
print('MSE',mean_squared_error(y_test,y_pred))
```

```
R2 score 0.9497950344519025
MAE 235001591.994157
MSE 1.6977762741608906e+17
```

Setelah dilakukannya pembentukan model dan membandingkan hasil masing-masing model, maka dilakukannya evaluasi untuk mengecek seberapa optimal hasil yang diperoleh terhadap model yang digunakan.

Evaluasi yang digunakan dalam analisis ini adalah K-Fold Cross Validation. K-fold cross-validation adalah jenis uji cross-validation yang dapat digunakan untuk mengevaluasi kinerja proses dari metode algoritmik dengan memisahkan sampel data acak dan mengelompokkan data hingga nilai K-fold. Hasil dari Cross Validation adalah sebagai berikut

	Model	RME	RMS	R2
0	Linear Regression	4.797761e+08	5.804952e+17	0.829184
1	Ridge Regression	4.797617e+08	5.804953e+17	0.829184
2	Lasso Regresion	4.797761e+08	5.804952e+17	0.829184
3	KNN	2.935740e+07	3.119692e+16	0.990799
4	Desicion Tree	2.384413e+08	1.712274e+17	0.949659
5	Random Forest	4.390240e+07	1.876441e+16	0.994471

Berdasarkan hasil Cross Validation, maka didapatkan bahwa model yang terbaik merupakan model Random Forest. Dengan menggunakan model ini, akan mempermudah dalam mencapai tujuan dan kriteria keberhasilan bisnis dikarenakan model tersebut menghasilkan hasil yang lebih baik daripada model lainnya.

CONCLUSION AND SUGGESTION

Berdasarkan hasil analisis yang telah dilakukan terhadap data house price jabodetabek, model yang memiliki performansi terbaik adalah model Random Forest dengan 9 atribut, yaitu 'district', 'city', 'facilities', 'certificate', 'property_condition', 'building_size_m2', 'land_size_m2', 'maid_bathrooms', dan 'electricity'. Dengan menggunakan model Random Forest dapat memberikan kemudahan dalam pemanfaatan dan pengimplemetasiannya dalam bisnis yang dilakukan. Dengan begitu, masalah-masalah atau kasus nyata yang ada dalam bisnis yang dilakukan dapat teratasi.